



UNIVERSIDAD
DE MÁLAGA



E.T.S.
INGENIERÍA
INFORMÁTICA



UNIVERSIDAD
DE MÁLAGA



E.T.S.
INGENIERÍA
INFORMÁTICA



UNIVERSIDAD
DE MÁLAGA



E.T.S.
INGENIERÍA
INFORMÁTICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADUADO EN INGENIERÍA DE SOFTWARE

2TRAIN – Análisis deportivo de la ciudad de Málaga

2TRAIN – Sports analysis of Malaga

Realizado por
Alejandro Ruiz Lara

Tutorizado por
Antonio Vallecillo Moreno
Dolores Burgueño Caballero

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, FEBRERO DE 2018

Fecha defensa:

Fdo. El/la Secretario/a del Tribunal



UNIVERSIDAD
DE MÁLAGA



E.T.S.
INGENIERÍA
INFORMÁTICA

Resumen

El avance del concepto que se entiende por web, sus aplicaciones y toda la capacidad informativa que ofrece crece a un ritmo exponencial. Este crecimiento centra su fundamento en aumentar el bienestar de sus usuarios y hacer cada vez más fácil la toma de decisiones. Participando de este enfoque, la necesidad de inventar y crear nuevo contenido cambia de concepto. Es la capacidad de agrupar adecuadamente estas cantidades ingentes de información la que empieza a proponer nuevas soluciones a este ámbito.

“2Train – Análisis deportivo de la ciudad de Málaga” es una aplicación web multiplataforma capaz de agrupar y procesar flujos de datos abiertos procedentes del Ayuntamiento de Málaga, que tienen temática deportiva y formatos aptos para la geolocalización. Este procesamiento se ejerce de distintas formas y conforme al objetivo general de la aplicación: facilitar la capacidad de decisión de un usuario a la hora de practicar cualquier actividad deportiva en la ciudad de Málaga.

Esta característica se otorga ofreciendo al usuario la capacidad de visualizar todas las instalaciones o áreas deportivas de la ciudad sobre un mapa. A la vez, se le permite filtrar dichos lugares en relación a su ubicación o la distinta temática de dichos centros deportivos. Por último, una lista seleccionable de deportes provoca un análisis de tendencia en Twitter en un radio significativo al contexto de la ciudad.

Un uso funcional y coherente de distintos flujos de información que incorporan al proyecto en la búsqueda tecnológica del incremento de bienestar ciudadano.

Palabras clave

Datos abiertos, Cartografía, Análisis de tendencia, Aplicación web, CartoDB, Twitter.

Abstract

The progress of the Web, its applications and the information it offers grows exponentially. One of the goals of this new interconnected world is to improve the wellness of its users, and to provide easier tools to support decision making. The need for creating new content changes its concept, and new solutions are starting to proliferate, based on coherently assembling and analyzing large sets of information flows.

“2Train – Análisis deportivo de la ciudad de Málaga” is a multiplatform web application capable of grouping and processing open data flows from the City of Malaga, which have sport themes and formats suitable for geolocation. This processing is carried out in different ways and according to the general objective of the application: to facilitate the decision-making capacity of users who want to practice sport activities in the city of Malaga.

The system is able to visualize in a single map all the sport areas and sport centers of Malaga. It allows users to filter these places by location, sport type, or facility. In addition, users can choose sports from a list and perform trend analysis in Twitter about the perceived interest of people in the vicinity to play these sports.

Keywords

Open data, Cartography, Trend analysis, Web application, CartoDB, Twitter

Índice

Introducción	1
1.1 OBJETIVO	1
1.2 CONTENIDO Y ESTRUCTURA DE DOCUMENTO	2
Motivación de proyecto	5
2.1 OBJETO	6
2.2 ANTECEDENTES	6
2.3 DESCRIPCIÓN DE SITUACIÓN ACTUAL	7
2.4 METODOLOGÍA Y FASES DE TRABAJO	8
Requisitos iniciales	11
Alcance	13
4.1 IDENTIFICACIÓN DE LAS PARTES	13
4.2 DEFINICIÓN	14
4.3 PLAN DE GESTIÓN DEL ALCANCE	16
4.4 SUPOSICIONES PRELIMINARES	17
4.5 DEPENDENCIAS, RESTRICCIONES Y LIMITACIONES	18
4.6 CRITERIOS DE ACEPTACIÓN	19
4.7 ENTORNO DE DESPLIEGUE	21
4.8 ENTORNO PARA LA IMPLEMENTACIÓN DEL SISTEMA ACTUAL	21
4.9 APLICACIONES COLABORADORAS	22
4.10 IDENTIFICACIÓN DE INTERESADOS. PERFILES DE USUARIO	23
Análisis y especificación de requisitos	25
5.1 REQUISITOS FUNCIONALES	25
5.2 REQUISITOS NO FUNCIONALES	38
5.2.1 REQUISITOS DE ASPECTO	38
5.2.2 REQUISITOS DE FACILIDAD DE USO Y APRENDIZAJE	39
5.2.3 REQUISITOS OPERACIONALES	40
5.2.4 REQUISITOS DE MANTENIMIENTO Y PORTABILIDAD	42
5.2.5 REQUISITOS LEGALES Y CULTURALES	43
5.3 REQUISITOS DE DOCUMENTACIÓN	44
5.3.1 MANUAL DE USUARIO	44
5.3.2 AYUDA EN LÍNEA	45
Análisis de riesgos	47
6.1 DEFINICIÓN DE RIESGOS	48
6.2 PLAN DE GESTIÓN DE RIESGOS	49

Modelado y diseño del sistema	55
7.1 CASOS DE USO	56
7.2 MODELO DEL DOMINIO	58
7.3 MODELADO ESTRUCTURAL: DIAGRAMA DE CLASES	60
7.4 MODELO DE NAVEGACIÓN	63
7.5 MODELO DE FLUJO DE PROCESOS	66
7.6 DISEÑO DE INTERFAZ	70
7.7 MODELADO DE COMPORTAMIENTO	72
7.7.1 DIAGRAMAS DE SECUENCIA	72
7.7.2 DIAGRAMAS DE MÁQUINAS DE ESTADO	76
Implementación	77
8.1 TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS	77
8.2 CARTOGRAFÍA WEB. DATOS ABIERTOS	80
8.3 BASE DE DATOS	84
8.4 ANÁLISIS DE TENDENCIA	87
8.5 PROGRAMACIÓN CONCURRENTE	89
8.6 FRAMEWORKS	91
8.6.1 FRONT-END	91
8.6.2 BACK-END	92
8.7 DISEÑO WEB	95
8.8 CLOUD COMPUTING. DESPLIEGUE	96
Mantenimiento y pruebas	99
9.1 CONSIDERACIONES PREVIAS	100
9.2 TESTS EXPERIMENTALES	102
9.2.1 PARTICIPANTES SELECCIONADOS	108
9.2.2 RESULTADOS OBTENIDOS	110
9.2.3 CONCLUSIONES	112
9.3 MANTENIMIENTO DE LA APLICACIÓN	113
Conclusiones	115
10.1 OBJETIVOS CUMPLIDOS	115
10.2 DIFICULTADES ENCONTRADAS	117
10.3 LÍNEAS FUTURAS	120
10.4 VÍDEO-PITCH	120
Referencias	123
Planificación temporal	127
Manual de usuario	137
Normas aplicadas	143

1

Introducción

Se inicia en este primer capítulo la definición del proyecto que motiva el trabajo de fin de grado que es objeto de estudio.

Antecediendo a la estructura de documento que se expone, se explica a continuación el objetivo que tiene esta memoria y cómo se desglosa hasta el más profundo detalle el proyecto “2Train – Análisis deportivo de la ciudad de Málaga”.

1.1 OBJETIVO

Con esta memoria de proyecto se pretende exponer al lector tres aspectos fundamentales en el desempeño de cualquier proyecto software.

En primer lugar, se esclarece el objeto del proyecto con la fundamentación de su idea general, los beneficios que se pretenden obtener con su realización y el modo en que este se acometerá.

Seguidamente, se estudia el trabajo realizado para lograr su consecución. Entendiendo como trabajo realizado la secuencia de pasos seguidos para alcanzar su conclusión. Es decir, las etapas que se han debido recorrer para dar dichos pasos, la planificación sobre la que se ha fundamentado la consecución de cada una de ellas y todo el material adicional que sustente dichos conceptos.

Finalmente, se exponen las conclusiones derivadas del desempeño del proyecto, tanto a nivel de producto tecnológico presente y futuro, como a nivel formativo.

1.2 CONTENIDO Y ESTRUCTURA DE DOCUMENTO

Para acometer los objetivos se ha seguido una estructura secuencial y enfocada al hilo temporal sobre el que se desarrolla el proyecto en cuestión.

Esta memoria se ha desarrollado siguiendo el esquema descrito en la norma española “UNE-157801: Proyectos Sistemas de Información”. Esta norma, desarrollada por AENOR, establece formalmente los requisitos de entrega indispensables para todo proyecto relacionado con sistemas de la información.

Capítulo 2: Motivación del proyecto

El capítulo 2 define la idea de negocio que se desarrolla en el proyecto, la cual está fundamentada sobre los problemas de composición de fuentes de información coherente que se han tenido en cuenta cabo y cómo se acopla el sistema al estado actual. Además, en este capítulo se definen y desarrolla la metodología y fases de trabajo elegidas para acometer el proyecto.

Capítulo 3: Requisitos iniciales

El capítulo 3 muestra las características principales, de alto nivel y más relevantes que el sistema debe ser capaz de cubrir.

Capítulo 4: Alcance

Puesta en contexto del enunciado del negocio, atendiendo a parámetros indispensables para poder acometer su desarrollo tecnológico. Define todos los aspectos necesarios para sostener el plan de acción que precederá a la implementación del sistema.

Capítulo 5: Estudio de alternativas y viabilidad

Análisis competitivo de la idea en desarrollo frente a las ya existentes en el mercado. También conlleva la puesta en común de aspectos del alcance definidos que puedan ser sustituidos por otros que mejoren las características del producto.

Capítulo 6: Análisis y especificación de requisitos

Este capítulo desarrolla la primera etapa de la metodología de trabajo software usada. En él, se extraen los requisitos funcionales y no funcionales del sistema; además de los necesarios de documentación como sistema software.

Capítulo 7: Análisis de riesgos

Exposición de los riesgos que pueden afectar al sistema, cómo se manejarán y medidas recomendadas y planificadas de mitigación y/o contingencia.

Capítulo 8: Modelado y diseño del sistema

Definición de los modelos desarrollados previa implementación del sistema web. Se expone el significado que dichos modelos dan al dominio y se explican las decisiones de diseño tomadas para aplicarlos. Fundamentado en casos de uso, modelo estructural y de comportamiento, definición de procesos, navegación web, diseño de interfaz y todos sus diagramas asociados.

Capítulo 9: Implementación

Especificación de las tecnologías y herramientas usadas para la implementación del sistema, analizando las implicaciones de cada una de ellas y el modo en que han sido aplicadas a la adaptación al sistema.

Capítulo 10: Mantenimiento y pruebas

Definición de las medidas consideradas para examinar el resultado obtenido, su vía de acción y las conclusiones derivadas del mismo. Además, se expondrá el plan de mantenimiento del sistema.

Capítulo 11: Conclusiones

Exposición de las consideraciones relevantes tras el desarrollo del sistema a nivel de negocio y formativo; además de las dificultades encontradas durante el mismo y posible trabajo futuro.

2

Motivación de proyecto

Este proyecto web nace de la puesta en común de ideas entre su autor, los tutores de Trabajo de Fin de Grado (TFG) y los distintos flujos de datos ofertados por el Ayuntamiento de Málaga.

La necesidad de alcanzar dichos conjuntos de información proviene de la motivación inicial del proyecto: ofertar una aplicación web capaz de proporcionar procesamiento de eventos complejos. Así, se pretende dar un producto funcional, que no solo conforme aspectos formativos, sino un producto con entidad propia.

Sin embargo, el análisis de la información proporcionada por la institución pública ha acabado derivando el enfoque concreto del proyecto a un nivel analítico centrado en la cartografía web actual, las posibilidades deportivas locales y el análisis de tendencia de sus ciudadanos.

2.1 OBJETO

“2Train – Análisis deportivo de la ciudad de Málaga” adquiere su formalización como aplicación web multiplataforma con el objeto de otorgar distintos beneficios, muy concretos, a cualquier ciudadano de la localidad, o visitante de ella, que pretenda consultar información con la que tomar decisiones respecto a su práctica deportiva en tiempo real.

De entre estos beneficios que caracterizan al sistema destaca como principal la visualización por parte del usuario de los distintos centros deportivos de la ciudad, clasificados por diversos criterios acordes a la información incluida en la aplicación y que se especifica más adelante. Se complementa este objetivo con la geolocalización del usuario en tiempo real, permitiéndole filtrar por distancia a él dichas ubicaciones.

Además de ayudar al usuario a seleccionar el centro deportivo más adecuado para sus intereses o deseos particulares, el sistema también pretende facilitar la decisión de qué deporte realizar. Esto lo consigue a través de la oferta de cuatro tipos de deportes que seleccionar. Su selección activa en background un proceso de análisis de tendencia sobre la plataforma Twitter, refiriéndose a usuarios cercanos a él y buscando tuits relacionados con los deportes elegidos. De este modo, el sistema hace más sólido su objeto de operación a la vez que incorpora el procesamiento de flujos de datos que le dio origen.

2.2 ANTECEDENTES

Es evidente que el sector de la tecnología web vive un momento de crecimiento exponencial, cuya explosión ha cambiado la vida de todos sus usuarios en términos que aún cuesta reconocer.

En los últimos veinte años, la capacidad de decisión de un usuario respecto a cualquier tema que afecte a su vida cotidiana ha variado desde un rango de días, a un rango de pocos segundos. Tanto es así, que hace poco más de diez años, cualquier persona de a pie solo podía ejercer la

práctica deportiva que se plantease de forma espontánea por aquellos lugares que conocía de antemano o que otras personas le habían recomendado (Luján Mora, 2014). No era viable ser turista en una ciudad nueva y tener la confianza suficiente como para practicar cualquier actividad deportiva. La falta de información acentuaba las limitaciones presenciales del usuario.

Sin embargo, con las tecnologías emergentes y la capacidad de relocalizar grandes cantidades de información en tiempo real, se ha querido llevar al olvido antiguas sensaciones como la descrita. Si bien, se ha llegado al punto adecuado en cuanto a capacidad informativa, se seguía manifestando una falta de congregación coherente de información para solventar este problema concreto.

El Ayuntamiento de Málaga dispone de una página web donde consultar una gran variedad de datos, pero sin embargo estos no pueden ser consultados cómodamente por ciudadanos no técnicos sin haber sido procesador con anterioridad. Atendiendo a la ingente cantidad de datos que proporciona el Ayuntamiento en cuestiones deportivas, se podía llegar a páginas web independientes donde consultar los puntos de recogida de “MálagaBici” o zonas de musculación al aire libre. Sin embargo, la funcionalidad seguía sin abarcar el objeto del proyecto en cuestión. Seguía faltando un sistema que agrupase dichos flujos de información de forma adecuada para proporcionar las capacidades anteriormente descritas a los miles de usuarios de la web interesados en el deporte.

2.3 DESCRIPCIÓN DE SITUACIÓN ACTUAL

Una vez analizado el contexto antecedente a la necesidad encontrada y los objetivos fundamentales para desarrollarla, cabe destacar la situación actual en la que se encuentra el sistema.

El sistema ha de atravesar todas las fases de desarrollo que caracterizan a la metodología iterativa incremental utilizada. Ello alcanza la consecución de un primer prototipo funcional, que engloba todas las

funcionalidades descritas y que es objetivo de este desarrollo académico. Este objetivo se define como tal, tras establecer el cumplimiento de lo que en este proyecto se entiende por éxito.

El sistema ha sido puesto a prueba con diversos usuarios que han valorado su funcionalidad y abordará de forma gratuita el objetivo docente al que aspira a la finalización de su proceso de producción. Siempre ayudado por el CEMI del Ayuntamiento de Málaga.

2.4 METODOLOGÍA Y FASES DE TRABAJO

Para acometer el desarrollo de este proyecto se ha usado una metodología ágil, que conlleva un proceso iterativo e incremental adaptada a un trabajo software académico. No se ciñe expresamente a la definición de metodologías conocidas en desarrollo software, tales como: Scrum o Extreme Programming, pero sigue su patrón de uso. De tal modo, se ha desarrollado el producto conforme a las normativas de desempeño de productos software actuales, cuyo trabajo involucra a un grupo inferior a diez trabajadores. (Iglesias Fraga, 2016)

Los roles a destacar en esta metodología adaptada al desempeño de un trabajo software académico son:

- **Jefe de Proyecto:** Cargo desempeñado por el alumno y autor del proyecto en cuestión. Capacidad para tomar las decisiones finales sobre el proyecto.
- **Ejecutor:** Cargo también desempeñado por el autor del proyecto. Conlleva la implementación de cada una de las fases de desarrollo para todo período de duración.
- **Cliente:** Cargo desempeñado por los tutores de proyecto, los cuales interactúan con el prototipo y evalúan periódicamente el ajuste del resultado al valor esperado. Alto índice de probabilidad de aumentar su satisfacción al participar en la mejora iterativa e incremental del producto.

- **Asesor:** Cargo desempeñado por los tutores de proyecto sobre el que se presentan disponibles en todo momento del período en desarrollo para facilitar información desconocida o poco accesible al ejecutor.

Dicha metodología ha implicado la realización de las fases que se describen a continuación, en períodos de dos semanas. Cada período se ha ceñido al desarrollo de una funcionalidad general del sistema hasta alcanzar la completitud de todas ellas.

Antes de comenzar a aplicar la metodología, es necesario establecer un período inicial y no iterativo de definición de las funcionalidades a implementar y estimación de períodos a cubrir hasta la conclusión del proyecto. Este proceso se identifica con la planificación temporal estimada que se ha llevado a cabo haciendo uso de la herramienta “MS Project” para facilitar su gestión. Esto sustenta el proceso de desarrollo sobre una base de planificación sólida, que descarta incertidumbre. Dichas funcionalidades definidas actúan a modo de actividades del proyecto y forman un listado de control con identificadores únicos asignados a las mismas. Sobre él se agrupan los períodos previstos para la ejecución del proyecto.

Una vez completadas, se prosigue con iguales períodos para concretar dichas funcionalidades y llevar al sistema a un estado de prototipo final con la viabilidad que se esperaba.

Durante cada período se han implementado las fases de desarrollo software expuestas en la Tabla 1 que se muestra a continuación:

Tabla 1: *Fases de desarrollo de la metodología aplicada*

FASE	DESCRIPCIÓN
Especificación de requisitos	Definición de las características exactas a cubrir por la función ejecutada, tanto a nivel funcional, como no funcional.
Modelado y diseño	Incorporación a los distintos modelos del sistema de los componentes necesarios para abordar la funcionalidad. Especificación paralela de las decisiones de diseño que conllevarse.
Implementación	Ejecución literal de la funcionalidad diseñada.
Pruebas	Evaluación experimental del prototipo generado tras cada iteración por parte de posibles usuarios reales de la aplicación. Esto implica formulación y análisis de tests experimentales a partir de ciertos criterios de éxito predefinidos.
Planificación	Replanteamiento del siguiente período de trabajo, incorporando posible modificaciones al período anterior.

3

Requisitos iniciales

En un primer acercamiento a las capacidades que el sistema debe cubrir, se definieron los siguientes requisitos iniciales que tuvieron en cuenta para definir sus características básicas y, sobre todo, para poder satisfacer la necesidad de producción de un primer prototipo.

Trabajando aún en un alto nivel de abstracción, los requisitos iniciales basan su utilidad en la perspectiva funcional del proyecto y en las características básicas que intervienen en sus requisitos no funcionales y de desarrollo. (Durán Toro, 2005)

Estos requisitos se muestran en la Tabla 2, expuesta a continuación:

Tabla 2: *Requisitos iniciales*

REQUISITOS	DESCRIPCIÓN
Entorno de desarrollo apropiado	Adquirir un IDE con capacidades de implementación web y pruebas de ejecución funcional.

Sitio web de despliegue	Tener acceso a un servicio web que proporcione alojamiento y ejecución de aplicaciones webs, tales como Cloud Computing.
Acceso a herramientas complementarias	Estudio y disposición de credenciales a las herramientas y servicios web que proveerán información funcional para la implementación del sistema.
Acceso a flujos de información	Identificación de las mejores fuentes de datos abiertos necesarias y estudio de los formatos adecuados.
Documentación de tecnologías aplicadas	Análisis y comprensión de las distintas tecnologías de back-end y front-end y de los distintos lenguajes aplicados al desarrollo.
Ofrecer ubicación cartográfica de centros deportivos malagueños	Ubicar visualmente y de forma clara todas las oportunidades deportivas de Málaga posibles.
Proveer distintas capacidades de filtrado	Capacidad de interacción a nivel de filtrado de centros deportivos y tipos de deporte según distintos criterios.
Visión de tendencia deportiva en tiempo real	Procesamiento de flujos de datos en tiempo real que muestre un análisis de tendencia deportiva.

4

Alcance

El alcance del proyecto se define como los límites que se establecen en lo relacionado al producto que se construye y las personas, organizaciones, otros productos y tecnología que tienen una relación directa con el sistema.

4.1 IDENTIFICACIÓN DE LAS PARTES

Título del Proyecto: Análisis ambiental para actividades deportivas en la ciudad de Málaga, usando procesamiento de eventos complejos.

Código identificador: 001

Cliente: Tutor de Trabajo de Fin de Grado a modo de cliente solicitante del Proyecto Software.

- Nombre: Antonio Vallecillo Moreno.
- Dirección: Bulevar Louis Pasteur, 35. 29071 Málaga.
- Teléfono: 952 13 97 24.

Suministrador: Alumno elaborador del Trabajo de Fin de Grado.

- Nombre: Alejandro Ruiz Lara
- Dirección: C/Carlomagno, 1. 29601 Marbella (Málaga)
- Teléfono: 626 84 30 44

Málaga, a 10 de Octubre de 2017

Cliente

Suministrador

4.2 DEFINICIÓN

Concretando el concepto al proyecto en desarrollo, el alcance viene dado por la facilitación al usuario en la toma de decisión respecto a dónde realizar una actividad deportiva en la ciudad de Málaga e incluso qué deporte es más recomendable en un momento determinado.

El cumplimiento de este objetivo global se ve dividido en tres campos fundamentales que componen los límites a los que se ciñe el sistema.

- Permitir la ubicación cartográfica en tiempo real de los diferentes centros deportivos de la ciudad.
- Ofrecer la capacidad de filtrado sobre dichos centros y áreas con respecto a la ubicación del usuario y/o la temática de esas ubicaciones.
- Permitirle conocer la preferencia a realizar ciertos deportes en tiempo real de sus conciudadanos a través de análisis de tendencia.

Teniendo claro el marco en que se encaja el alcance de este proyecto, a continuación se limitan las características tratadas con mayor grado de detalle.

La ubicación geográfica de los centros y áreas deportivos de la ciudad provienen de archivos de datos abiertos en formato CSV, GeoJSON o SHP. Como cualquier fuente de datos externa, su contenido es gestionado y actualizado a beneplácito del proveedor, en este caso el Ayuntamiento

de Málaga. No obstante, otorga a la aplicación “2Train – Análisis deportivo de la ciudad de Málaga” el número suficiente de regiones para proporcionar un resultado funcional.

Las cualidades de filtrado vienen establecidas por las distintas funciones que establece la base de datos (PostgreSQL), la cual subyace a la herramienta especializada en tratamiento de datos abiertos que se ha utilizado y que se definirá más adelante en esta memoria. La buena predisposición de los flujos de datos usados y de la herramienta en su generalidad ha hecho muy complejo la definición de dichas cualidades sin extralimitaciones que excediesen demasiado el concepto global y académico de este proyecto.

La selección progresiva de puntos geométricos y geolocalizables tal y como se hiciese con cualquier sentencia SQL, sobre una base de datos estáticas y el uso de funciones especializadas para el tratamiento de los datos almacenados han fundamentado la interacción del usuario que se propone. Esta y otras implementaciones se derivan al apartado de implementación de esta memoria.

Por último, el análisis de tendencias en la población es un ámbito de gran magnitud que ha sido necesario limitar cuidadosamente bajo los criterios del sistema propuesto. A pesar de no ser la red social con mayor flujo de tráfico, se ha decido examinar el tráfico de Twitter sobre otras redes sociales, dada la utilidad de la información textual, constante y en tiempo real que se genera. Además, otras cuestiones técnicas referidas más adelante han hecho que la herramienta utilizada se haga ideal para acometer el punto final que completa el objetivo de la aplicación.

Muchos frentes abiertos y posibles para lograr el objetivo bien marcado de facilitar la decisión deportiva a cualquier usuario. De dichas posibilidades se han seleccionado los más ilustrativas respecto a nuevas tecnologías a aprender durante el proyecto, y las que más posibilidades ofrecían para incrementar el valor del sistema en su continuo refinamiento.

4.3 PLAN DE GESTIÓN DEL ALCANCE

En el plan de gestión del alcance del proyecto se plasman de forma atómica los criterios que se siguen para la elaboración, aceptación y/o actualización de distintos procesos referentes a la definición del alcance del proyecto (Managers, 2014).

Al tratarse de un proyecto individual y delimitado por un límite horario de trabajo independiente, dicha definición de procesos se hace de forma resumida y tentativa. Habiendo implementado el plan que se detalla a continuación:

- **Proceso para definir en detalle el Alcance del Proyecto:** Partiendo de una estructura jerárquica, Padre-Hijo, se enuncia el alcance de este proyecto partiendo del objetivo global y refinándolo en subobjetivos que ayudarán a facilitar la capacidad de decisión que se pretende cubrir.
- **Proceso para la definición de la Estructura de Desglose del Trabajo (EDT):** A partir de la definición del alcance, se extraen los entregables asociados a cada sub-objetivo definido. Esos entregables componen el nivel superior de la EDT, la cual se descompone en tareas atómicas con las que acometer el proyecto. Esta descomposición se realiza de forma esquemática y compone la base para la planificación temporal del proyecto.
- **Proceso para la aceptación de los entregables del proyecto:** Este TFG se compone de dos entregables generales: la aplicación web objetivo y la presente memoria. Respecto a la aplicación web, se genera un entregable cada dos semanas en el que se implementa de forma incremental una de sus funcionalidades. Estas entregas son aceptadas si cumple el alcance establecido para la funcionalidad detallada. En el caso de la memoria, se entrega una versión de ella cada dos capítulos completados. Estos son aceptados si cumplen unos niveles mínimos de lenguaje, sin excederse, y si tratan los temas esperados por su apartado.

- **Proceso para definir cambios sobre el sistema:** Este proceso se desarrolla conforme lo hace el proyecto, dada la metodología de trabajo seleccionada. Las solicitudes de cambio las establece el cliente al ejecutor tras el análisis de cada entregable al fin de su plazo. Estos cambios pueden ser o bien la integración de nueva funcionalidad en el sistema, o la sustitución de aspectos poco funcionales antes definidos por otros nuevos. En cualquier caso, los cambios se ven plasmados tanto en la implementación como, de forma paralela, en cada una de sus etapas de desarrollo: requisitos, modelado y pruebas.

4.4 SUPOSICIONES PRELIMINARES

A continuación se enumeran todos aquellos aspectos que se dan por conocidos o supuestos en la definición del problema y en el planteamiento de la solución.

Es muy importante documentar estas suposiciones preliminares, ya que ejercen un alto valor en el nivel de mantenimiento y pruebas. Es muy probable que un fallo del sistema o una ausencia de funcionalidad sea provocada por una de estas suposiciones.

- Los datos abiertos con los que se trabaja son los más fiables ofrecidos por parte del Ayuntamiento de Málaga.
- Los datos abiertos con los que se trabaja serán actualizados asiduamente para plasmar cambios en centros deportivos.
- Twitter producirá un flujo de tuits superior a trescientos por minuto durante el procesamiento necesario.
- Todo usuario interesado en la aplicación se encontrará en Málaga o querrá operar con una ubicación en la ciudad de Málaga.

- Todo usuario interesado en la aplicación dispondrá de conexión a internet en su acceso.
- Los datos abiertos con los que se trabaja no serán eliminados sin sustitución en su repositorio de origen.
- Todo usuario interesado en la aplicación dispondrá de capacidad visual para operar sobre el mapa.

4.5 DEPENDENCIAS, RESTRICCIONES Y LIMITACIONES

Las dependencias de un proyecto vienen establecidas por la vinculación que tiene alguno de sus aspectos, herramientas utilizadas o funcionalidades, con respecto a otros recursos ajenos a su control.

Conociendo esto, es evidente la correspondencia que se va a establecer entre cada uno de los tres sub-objetivos definidos en el alcance y las herramientas y datos usados para acometerlos. Son los siguientes:

- La capacidad informativa del mapa depende en un alto grado de la gestión del archivo de datos abiertos que haga el Ayuntamiento en su origen. De ser desechado, las ubicaciones e información cartográfica del mapa dejarían de actualizarse. Los datos ofrecidos dejarían de variar.
- El análisis de tendencia deportiva depende enteramente de Twitter. De este modo, si en el radio requerido de ubicación se produce un número de tuits muy inferior al previsto por la aplicación, el resultado ofrecido en el análisis no sería significativo y el radio se amplía en un grado de magnitud.
- La interactividad ofrecida entre usuario y mapa para su filtrado depende en gran medida de la API ofrecida por la herramienta cartográfica utilizada. Si la biblioteca en cuestión deja de actualizarse o se desecha, la funcionalidad de filtrado debería actualizarse junto con ella.

- El flujo de usuarios simultáneos que acepta la aplicación depende de la capacidad contratada en su sitio de alojamiento web. El trato ofrecido por el Platform as a Service (PaaS) en el que se desplegará impondrá los límites de tráfico.

Por otra parte, no solo son las dependencias quienes marcan los impedimentos de la aplicación. También es muy importante destacar las restricciones y limitaciones que ofrece el sistema a cualquier usuario que pretenda usarla. Estas limitaciones, a diferencia de las dependencias, se enumeran en el contexto de funcionalidad del sistema.

- No se podrán filtrar centros deportivos bajo más criterios temáticos que los preestablecidos: general, musculación, natación, playas, vóley playa, carriles bici, puntos de recogida “MálagaBici” y zonas verdes.
- No se podrán filtrar centros deportivos por una distancia superior a veinte kilómetros del usuario.
- El análisis de tendencia solo se efectuará sobre una selección deportiva preestablecida, la cual el usuario puede confeccionar a su gusto.
- La operación continua con la aplicación requiere una conexión a la red permanente.
- Problemas visuales en la percepción de color por parte del usuario harán inservibles algunas capacidades de filtrado.

4.6 CRITERIOS DE ACEPTACIÓN

Para llegar a un punto en el que se consensue la aceptación del proyecto como tal, se habrán tenido que cumplir una serie de criterios definidos en este apartado.

El éxito del proyecto no se mide en términos del alcance de la finalización del desarrollo del mismo, sino que se deben tener en cuenta muchos otros criterios que conformen dicha finalización.

Este proyecto entiende el éxito como la capacidad de completar el sistema dentro de los límites de alcance, tiempo, costo, calidad, recursos y riesgo acordados, estableciendo al menos un criterio de aceptación para cada objetivo marcado en el alcance del mismo.

Para alcanzar el éxito del proyecto se deben superar los siguientes criterios:

- Finalizar todo entregable en un plazo inferior al día límite de entrega establecido por la Escuela Técnica Superior de Ingeniería Informática de Málaga.
- Finalizar el sistema en un plazo no superior al 5% respecto a la planificación temporal estimada adjunta.
- Ofrecer la ubicación cartográfica de al menos tres tipos de centros deportivos diferentes de Málaga.
- Ofrecer al menos tres capacidades de filtrado diferentes sobre el conjunto general de centros y áreas deportivas.
- Mostrar sobre el mapa la geolocalización del usuario en tiempo real.
- Realizar el análisis de tendencia sobre un flujo de datos significativo respecto a la ciudad de Málaga.
- Ofrecer interactividad sobre el nivel de procesamiento de la aplicación para controlar la gestión de los recursos móviles.
- Permitir el cambio manual de ubicación a tener en cuenta como localización del usuario.
- Actuar sobre las peticiones del usuario con un tiempo de respuesta menor a cinco segundos.

- Permitir al usuario alcanzar su objetivo en menos de tres clicks.

4.7 ENTORNO DE DESPLIEGUE

El estudio del entorno de despliegue del sistema se caracteriza por la definición de este como producto de entidad propia e independiente. No forma parte de ningún otro producto, ni es un sistema pensado para ejercer como librería o servicio auxiliar para otros sistemas que puedan estar interesados.

El objetivo fundamental en este ámbito para el proyecto es otorgar la mayor sencillez posible en la capacidad de localización e interacción del usuario con el sistema. Por lo tanto, “2Train – Análisis deportivo de la ciudad de Málaga” es una aplicación web multiplataforma. Es decir, adquiere el carácter web que hace universalmente accesible a cualquier aplicación a través de internet y se diseña para poder ser accedida desde cualquier dispositivo. Este aspecto es realmente interesante cuando el objetivo general de facilitar la toma de decisión deportiva de sus usuarios puede ser requerido de forma espontánea. Cualquier usuario puede necesitar acceder a la aplicación sin tener un ordenador a mano.

Este aspecto y el auge de los dispositivos móviles, que hace ya más de cinco años que superaron en número a los ordenadores de sobremesa, hace que el carácter multiplataforma gane relevancia en el proyecto.

4.8 ENTORNO PARA LA IMPLEMENTACIÓN DEL SISTEMA ACTUAL

Desde el punto de vista de la implementación propia, el sistema está pensado para sacar el máximo partido posible a las tecnologías que facilitan hoy día el trabajo en la red.

Desde una perspectiva tradicional, la aplicación web está pensada para ser desplegada en cualquier servidor de aplicaciones web con soporte para el uso del lenguaje Java como base del Framework

Spring y para permitir el intercambio de mensajes entre distintos servicios web de manera fluida.

Sin embargo, se aborda este objetivo sin la necesidad de alquilar o comprar un servidor externo que configurar y mantener. El auge de las tecnologías Cloud ha hecho que el despliegue tecnológico del proyecto se realice bajo el concepto Platform as a Service (PaaS). La parte de desarrollo del proyecto solo se preocupa de la implementación y funcionalidad del sistema, mientras que los aspectos de seguridad, configuración y hardware queda en manos de la empresa elegida para desplegar el código fuente.

Este aspecto deja a un lado el “hosting” e integra la aplicación en el mundo del “Cloud Computing”. Se comienza con una cuenta gratuita de nivel de servicio escaso pero suficiente para el objetivo primario no comercial del proyecto. Además, dado que es la empresa externa quien se encarga del mantenimiento del servidor, el nivel de servicio y flujo de tráfico permitido es escalable según la necesidad del sistema en tiempo real.

Un entorno para la implementación flexible y correlacionado con el ámbito web, multiplataforma y actual que caracteriza al sistema.

4.9 APLICACIONES COLABORADORAS

En principio, el proyecto no cuenta con una colaboración empresarial directa. Existen aplicaciones que el sistema utiliza para ofrecer su servicio. Los dos ejemplos de ello son Twitter y CartoDB. Sin embargo, su carácter de API gratuita hace que la relación entre ellas y este sistema adquiera el grado de colaboración.

La utilidad prevista para este producto como sistema asociado a las tecnologías del Ayuntamiento de Málaga implica una previsión de colaboración entre la utilidad y facilidad de uso informativo de esta aplicación y otros recursos más sofisticados de aplicaciones privadas del consistorio. El uso de datos abiertos del sistema puede

verse mejorado por la adquisición de otros flujos de datos de mayor utilidad proveídos por aplicaciones como: EMT Málaga (Aparcamientos), Audioguía oficial del Ayuntamiento de Málaga (museos), etc.

4.10 IDENTIFICACIÓN DE INTERESADOS. PERFILES DE USUARIO

Los interesados a destacar en el desarrollo de este proyecto vienen dados por la metodología de trabajo utilizada. Un interesado en el proyecto es toda persona u organización que se ve afectada por el resultado producido en el desarrollo del sistema. (García, 2014)

Como se definió con anterioridad, el proyecto va a contar con los perfiles de: Jefe de Proyecto, Ejecutor, Cliente y Asesor. Para no caer en redundancia en la descripción de cada rol, ahora se van a definir desde el punto de vista de su interés principal en el proyecto y las responsabilidades que ejercen en él, atendiendo al ámbito general del proyecto y no desde el punto de vista de la metodología de trabajo.

Esta información se muestra en la Tabla 3 que se expone a continuación:

Tabla 3: *Perfiles de interesados*

INTERESADO	INTERÉS PRINCIPAL	RESPONSABILIDADES
Jefe de proyecto	Sacar el mayor beneficio posible al equipo de trabajo para encarar futuros proyectos y obtener beneficios.	<ul style="list-style-type: none"> - Desarrollo del proyecto frente al cliente. - Obtener buen desarrollo por parte del ejecutor. - Extraer la información necesaria del cliente.

Ejecutor	Ganar valor profesional al concluir con éxito sus tareas.	- Funcionamiento correcto y coherente respecto a sus tareas asignadas.
Cliente	Alcanzar las funcionalidades que espera de la forma más rápida y simple posible.	- Confiar en el equipo adecuado y otorgar la información correcta respecto a sus intereses.
Asesor	Ofrecer información relevante y de ayuda para obtener un mayor beneficio.	- Buena ejecución de cualquier tarea relacionada con su ámbito de conocimiento.

5

Análisis y especificación de requisitos

Se define en este apartado la especificación de requisitos del proyecto. Estos quedan completados con el análisis de requisitos funcionales, no funcionales y los respectivos a la documentación.

El apartado que se desarrolla constituye la primera fase específica de la metodología de desarrollo que se ha utilizado. Por tanto, dichos requisitos son definidos de forma incremental junto con el avance de las iteraciones del proyecto.

5.1 REQUISITOS FUNCIONALES

Estos son los requisitos enfocados a especificar de la forma más clara y concisa posible todas las capacidades que debe proveer el sistema para abordar las expectativas de funcionalidad desarrolladas por el autor y cualquier usuario potencial.

La definición de este tipo de requisitos se ve fuertemente ligada a la fase de recolección de casos de uso del sistema. Esto se debe a que los casos de uso definen las capacidades de actuación que el usuario tiene sobre el sistema, y dichas características se ven fuertemente identificadas por los requisitos funcionales. Por lo tanto, aquí se describen dichas características funcionales y además se refinan hasta el nivel más atómico y funcional necesario. Los casos de uso se definen de forma paralela y se exponen en la sección de modelado y diseño del sistema.

Para involucrar el análisis de riesgos con la especificación de requisitos, se realiza una medida de priorización para cada uno de ellos sobre una escala independiente realizada para el proyecto: Muy baja, Baja, Alta, Muy alta. (Halad, Ridaou, & Kaplan, 2004)

- **Visualizar sobre un mapa los distintos centros deportivos de la ciudad de Málaga.**

IDENTIFICADOR

RF-001

DESCRIPCIÓN

Característica que permite al usuario tener una vista sencilla e intuitiva en pantalla que muestre el mapa aéreo de la ciudad de Málaga con distintas marcas que ubiquen sus centros deportivos.

PRIORIZACIÓN

Este requisito funcional envuelve el objetivo más ambicioso del proyecto. Una fuente sencilla en la que mostrar el procesamiento de datos abiertos que da origen a la aplicación es esencial. Por lo tanto, su relevancia en el sistema es considerada como: "Muy alta".

REQUISITOS DERIVADOS

De esta característica se pueden derivar los requisitos no funcionales mostrados en la Tabla 4 expuesta a continuación:

Tabla 4: *Requisitos derivados de RF-001*

IDENTIFICADOR	CAUSA
RNF-002	El mapa se debe mostrar con un aspecto amplio, fácil de visualizar y con la opción más genérica activada por defecto.
RNF-005	El mapa no debe tardar en crearse más de lo que cualquier usuario esté dispuesto a esperar en la carga de una página.

SUBFUNCIONALIDADES

No procede destacar funcionalidades derivadas de este requisito.

ENTRADAS

Activación de la aplicación por parte del Usuario.

PROCESOS

Cuando el usuario abre la aplicación, el sistema carga su página principal, que es la única que contiene, y hace una petición por defecto a la base de datos PostgreSQL de CartoDB de un archivo GeoJSON con las posiciones de los centros deportivos y zonas verdes de la ciudad de Málaga.

SALIDAS

El mapa generado por la API “carto.js” a partir de la selección de datos obtenida en formato GeoJSON.

- **Permitir capacidad de filtrado sobre los centros deportivos mostrados según su temática.**

IDENTIFICADOR

RF-002

DESCRIPCIÓN

Característica que permite al usuario modificar la visión actual del mapa, obteniendo las ubicaciones de distintos puntos deportivos de la ciudad de Málaga.

PRIORIZACIÓN

Dado que la visualización del mapa engloba por defecto la ubicación de centros deportivos de carácter general y zonas verdes, esta característica complementa de forma satisfactoria la interacción del usuario. Sin embargo, su importancia no haría inservible completamente la aplicación. Su prioridad es “Alta”.

REQUISITOS DERIVADOS

De esta característica se pueden derivar los requisitos no funcionales mostrados en la Tabla 5 expuesta a continuación:

Tabla 5: *Requisitos no funcionales derivados de RF-002*

IDENTIFICADOR	CAUSA
RNF-003	Los botones de filtrado deben ser indicativos de su función.
RNF-005	El mapa no debe tardar en recargarse más de lo que un usuario esté dispuesto a esperar.
RNF-008	Una recarga del filtrado del mapa no debe hacer caer el sistema.

SUBFUNCIONALIDADES

Tabla 6: *Subfuncionalidades de RF-002*

IDENTIFICADOR	DESCRIPCIÓN
2.1	Filtrar las ubicaciones, mostrando los carriles bici y los puntos de recogida de bicis ("MálagaBici").
2.2	Filtrar las ubicaciones, mostrando los gimnasios y zonas de musculación al aire libre.
2.3	Filtrar las ubicaciones, mostrando las piscinas, playas y accesos de vóley playa.
2.4	Filtrar las ubicaciones, mostrando los centros deportivos generales y las zonas verdes con independencia de ser la característica por defecto.
2.5	Mostrar los niveles de ruido esperados por las distintas zonas de la ciudad de Málaga.

ENTRADAS

Activación o desactivación de cualquiera de los botones destinados a este propósito en la interfaz.

PROCESOS

Obtener esta orden por parte del usuario hace que el sistema modifique la sentencia SQL que mantiene la muestra del mapa hasta ese momento, obteniendo así un conjunto distinto de cartografía a mostrar.

SALIDAS

El mapa generado por la API “carto.js” a partir de la selección de datos obtenida en formato GeoJSON.

- **Tener visible la ubicación del usuario en el momento de activación de la aplicación.**

IDENTIFICADOR

RF-003

DESCRIPCIÓN

El usuario podrá ubicarse a sí mismo en el mapa de forma automática una vez inicia la aplicación. Esto además le servirá para que el sistema pueda tenerla en cuenta a la hora de analizar la tendencia a su alrededor y ofrecer un filtrado por distancia a él.

PRIORIZACIÓN

Este requisito funcional no es el más relevante de la aplicación. Sin embargo, el mostrar la ubicación actual del usuario sobre el mapa facilita su capacidad de decisión y ofrece posibilidades de ampliación de funcionalidad, como el análisis de tendencia. Su prioridad es considerada “Alta”.

REQUISITOS DERIVADOS

De esta característica se pueden derivar los requisitos no funcionales mostrados en la Tabla 5 expuesta a continuación:

Tabla 7: *Requisitos no funcionales derivados de RF-003*

IDENTIFICADOR	CAUSA
RNF-005	La carga de la posición del usuario debe hacerse de forma simultánea al resto del mapa.
RNF-006	No se usará la posición del usuario de forma externa a su sesión.
RNF-007	La posición del usuario tendrá una precisión válida en los límites del sistema.
RNF-008	El acceso erróneo a la posición del usuario no provocará el fallo del sistema.

SUBFUNCIONALIDADES

Tabla 8: *Subfuncionalidades de RF-003*

IDENTIFICADOR	DESCRIPCIÓN
3.1	A través de un campo que reciba el número de kilómetros máximo al que el usuario quiere ver los centros deportivos seleccionados, el sistema filtra la muestra del mapa actual con la selección que se encuentre a una distancia igual o inferior a la expresada.
3.2	Ofrecer un centro de localización sobre el que gestionar el análisis de tendencia en caso de que el usuario lo requiera.

ENTRADAS

Activación de la aplicación por parte del Usuario.

PROCESOS

Cuando el usuario abre la aplicación, el sistema accede de forma automática a su ubicación a través del navegador con el que esté operando. Esta posición es marcada a través de la modificación SQL de un parámetro reservado en la capa de ubicación del mapa mostrado.

SALIDAS

El mapa generado por la API “carto.js” con el punto de la ubicación del usuario marcado de forma distinguible sobre el resto.

- **Modificar la ubicación tentativa del usuario de forma manual.**

IDENTIFICADOR

RF-004

DESCRIPCIÓN

El usuario puede modificar el punto que muestra su ubicación sobre el mapa introduciendo una nueva dirección de forma manual.

PRIORIZACIÓN

Este requisito funcional no es demasiado relevante. Amplía la capacidad de operación de la aplicación al permitir analizar tendencias y filtrar por distancia a ubicaciones en las que el usuario no está en ese momento, pero con su posición inicial podría ser suficiente para mantener el nivel de operación del sistema. Su prioridad es “Baja”.

REQUISITOS DERIVADOS

De esta característica se pueden derivar los requisitos no funcionales mostrados en la Tabla 9 expuesta a continuación:

Tabla 9: *Requisitos no funcionales derivados de RF-004*

IDENTIFICADOR	CAUSA
RNF-003	La introducción de la nueva dirección no debe llegar a los tres pasos.
RNF-004	Debe quedar claro dónde y cómo se ejecuta el cambio de dirección del usuario
RNF-005	La carga de la posición del usuario no debe exceder el límite de espera del usuario
RNF-006	No se usará la posición del usuario de forma externa a su sesión.
RNF-007	La posición del usuario tendrá una precisión válida en los límites del sistema.
RNF-008	El acceso erróneo a la posición del usuario no provocará el fallo del sistema.

SUBFUNCIONALIDADES

No procede destacar funcionalidades derivadas de este requisito.

ENTRADAS

Activación de la aplicación por parte del Usuario y pulsación del botón que hace visible el campo de modificación de ubicación.

PROCESOS

Una vez el usuario tiene visible el campo de texto para cambiar la dirección de su localización y lo rellena, el sistema se hace cliente de un servicio web que le devuelve las coordenadas de dicha localización.

Con ellas, se modifica el campo SQL de la base de datos de CartoDB dedicado a la capa de ubicación.

SALIDAS

El mapa que se mostraba anteriormente pero con el punto de la ubicación del usuario movido de lugar.

- **Exponer los horarios de amanecer y anochecer diarios de la ciudad de Málaga.**

IDENTIFICADOR

RF-005

DESCRIPCIÓN

El usuario puede ver en la interfaz la hora de amanecer y anochecer en la ciudad y así obtener mayor información para tomar su decisión deportiva.

PRIORIZACIÓN

Este requisito funcional no tiene relevancia en el sistema. Dado que las fuentes de información en este ámbito son infinitas, se ha añadido por ampliar la capacidad de decisión que intenta solventar el sistema. No tiene operativa interna respecto al objetivo fundamental del proyecto. Su prioridad es “Muy baja”.

REQUISITOS DERIVADOS

De esta característica se pueden derivar los requisitos no funcionales mostrados en la Tabla 10 expuesta a continuación:

Tabla 10: *Requisitos no funcionales derivados de RF-005*

IDENTIFICADOR	CAUSA
RNF-005	El tiempo de espera por la aparición de las horas de amanecer y anochecer no puede exceder a la carga de la página.
RNF-008	Un error en la carga del horario no debe afectar al resto de funcionalidad.

SUBFUNCIONALIDADES

No procede destacar funcionalidades derivadas de este requisito.

ENTRADAS

Activación de la aplicación por parte del Usuario.

PROCESOS

Cuando el usuario abre la aplicación, un proceso interno hace de cliente a un servicio web REST externo que, pasándole las coordenadas de la ciudad, devuelve un JSON con los horarios de amanecer y anochecer.

SALIDAS

Exposición en pantalla de la información devuelta por el JSON, limitada a las horas de anochecer y amanecer en un formato legible para usuarios no técnicos.

- **Conocer la tendencia actual alrededor del usuario a realizar ciertos deportes.**

IDENTIFICADOR

RF-006

DESCRIPCIÓN

El usuario puede seleccionar uno o varios deportes de una lista preestablecida y conocer el porcentaje de usuarios de Twitter que están tratando temas relacionados con ellos en un radio perimetrado a su ubicación.

PRIORIZACIÓN

Después de la capacidad de visualizar la ubicación de centros y áreas deportivas, esta es la funcionalidad que promueve el sistema. El análisis de la tendencia de otras personas a realizar el deporte sobre el que se quiere decidir el usuario facilita en gran medida su decisión. Por lo tanto, la prioridad es “Alta”.

REQUISITOS DERIVADOS

De esta característica se pueden derivar los requisitos no funcionales mostrados en la Tabla 11 expuesta a continuación:

Tabla 11: *Requisitos no funcionales derivados de RF-006*

IDENTIFICADOR	CAUSA
RNF-003	El inicio, modificación o detenimiento del análisis no debe exceder los tres pasos.
RNF-004	Debe quedar claro dónde y cómo se ejecuta análisis de tendencia y su resultado.

RNF-008	El detenimiento inesperado del análisis podrá ser reanudado y no conlleva caída del sistema.
RNF-015	El uso de la información de tuits y sus usuarios no excederá lo permitido por la API pública de Twitter.
RNF-016	Solo se tendrán en cuenta en el análisis, los tuits con base cultural similar a la de la ciudad de origen del sistema.

SUBFUNCIONALIDADES

Tabla 12: *Subfuncionalidades de RF-006*

IDENTIFICADOR	DESCRIPCIÓN
6.1	Gestionar el análisis de tendencia. Ser capaz de modificar la selección de deportes con la que se inicia el análisis de tendencia.
6.2	Detener el análisis de tendencia. El nivel de procesamiento interno que realiza dicho análisis puede consumir recursos de los dispositivos. Es aconsejable poder detenerlo cuando el usuario lo requiera.

ENTRADAS

Activación de la aplicación por parte del Usuario y pulsación de uno o más deportes a analizar como tendencia del listado preestablecido.

PROCESOS

El listado de deportes seleccionados se pasa al controlador y este genera una hebra en background que consulta los tuits más recientes de Twitter en un radio cercano a la ubicación del usuario.

SALIDAS

Se devuelve un valor indicativo del número de tuits con coincidencias de palabras claves relacionadas con los deportes seleccionados, con el que se muestra una barra de progreso en la interfaz.

5.2 REQUISITOS NO FUNCIONALES

Los requisitos no funcionales son aquellas necesidades del sistema que no tienen relación con la funcionalidad explícita que tiene que dar.

En este caso, las características funcionales que se deben ofrecer y el sector de público al que van dirigidas provocan una necesidades operacionales, de rendimiento, legales, de usabilidad... que deben ser expuestas para la correcta formación del sistema.

5.2.1 REQUISITOS DE ASPECTO

- **Interfaz:**

IDENTIFICADOR

RNF-001

DESCRIPCIÓN

El nivel de uso del sistema, en cuanto a la espontaneidad que se prevé y la inmediatez que puede requerir, hace necesario un aspecto intuitivo, de componentes simples y claros.

- **Estilo del producto:**

IDENTIFICADOR

RNF-002

DESCRIPCIÓN

El producto debe ofrecer las funcionalidades necesarias desde el menor número posible de interacciones requeridas para alcanzarlas. La simpleza debe ser paralela a lo sencillo de la idea de deporte que se tiene.

5.2.2 REQUISITOS DE FACILIDAD DE USO Y APRENDIZAJE

- **Facilidad de uso:**

IDENTIFICADOR

RNF-003

DESCRIPCIÓN

Todos los componentes mostrados no deben requerir más de un paso para operar sobre la interfaz tal y como el usuario espera.

- **Facilidad de aprendizaje:**

IDENTIFICADOR

RNF-004

DESCRIPCIÓN

Todo elemento mostrado en pantalla debe tener un aspecto y/o título intuitivo respecto a su operación.

5.2.3 REQUISITOS OPERACIONALES

- **Requisitos de velocidad.**

IDENTIFICADOR

RNF-005

DESCRIPCIÓN

El sistema debe gestionar cualquier petición del usuario en menos de tres segundos. (Doom, 2016)

- **Requisitos de seguridad crítica.**

IDENTIFICADOR

RNF-006

DESCRIPCIÓN

El sistema debe garantizar que la transacción de la ubicación del usuario se limitará a aparecer en su propio mapa y no será utilizada en ubicaciones externas.

- **Requisitos de precisión.**

IDENTIFICADOR

RNF-007

DESCRIPCIÓN

El sistema ubicará los centros con un radio de error inferior a los cien metros.

Además, no contendrá componentes que proporcionen más de una funcionalidad diferente.

- **Requisitos de fiabilidad y disponibilidad.**

IDENTIFICADOR

RNF-008

DESCRIPCIÓN

Como aplicación web, el sistema deberá ser accesible en todo momento que reciba una petición de acceso.

No debe dejar de dar servicio en una métrica superior a las tres recargas erróneas. Se propone su replicación cuando alcance niveles de acceso superiores a los docentes.

- **Requisitos de escalabilidad.**

IDENTIFICADOR

RNF-009

DESCRIPCIÓN

El sistema debe estar servido en un entorno tecnológico que permita escalar su disponibilidad de forma simultánea y flexible a los accesos que se le hacen.

- **Entorno físico.**

IDENTIFICADOR

RNF-010

DESCRIPCIÓN

Aunque no sea conocida, el sistema debe contar con una ubicación física de despliegue con la que operar en caso de modificaciones o ampliaciones de lenguaje.

- **Entorno tecnológico.**

IDENTIFICADOR

RNF-011

DESCRIPCIÓN

El sistema debe garantizar poder ser accedido por cualquier dispositivo tecnológico con conexión a internet.

- **Soporte.**

IDENTIFICADOR

RNF-012

DESCRIPCIÓN

El sistema debe garantizar al menos un botón de ayuda que muestre un mensaje con la funcionalidad de alguna característica poco intuitiva.

5.2.4 REQUISITOS DE MANTENIMIENTO Y PORTABILIDAD

- **Requisitos de mantenimiento:**

IDENTIFICADOR

RNF-013

DESCRIPCIÓN

El sistema debe ofrecer un código modular y legible, para poder ser mantenido por distintos técnicos en su proceso de vida.

- **Requisitos de portabilidad**

IDENTIFICADOR

RNF-014

DESCRIPCIÓN

El sistema debe garantizar que su tecnología de implementación y su entorno de despliegue no condicionará el acceso de ningún dispositivo con conexión a la red.

5.2.5 REQUISITOS LEGALES Y CULTURALES

- **Requisitos legales.**

IDENTIFICADOR

RNF-015

DESCRIPCIÓN

El sistema garantizará el uso exclusivo de la información de usuario proporcionada por la API gratuita de Twitter, sin su almacenamiento ni búsqueda paralela de información personal.

Además, no mostrará el contenido de sus tuits a modo de información propia.

Con ello se prevé la integridad de la legislación de propiedad de datos y la ley de propiedad intelectual.

- **Requisitos culturales.**

IDENTIFICADOR

RNF-016

DESCRIPCIÓN

El sistema mostrará la información en el idioma predominante en su país de origen, español.

Además, usará el sistema numérico actual de la Unión Europea. El sistema arábigo.

5.3 REQUISITOS DE DOCUMENTACIÓN

Este apartado de la especificación de requisitos del sistema implica la descripción de la documentación que es necesaria para el correcto despliegue del proyecto, en caso de ser comercializado como producto de entidad propia.

Dado el carácter de aplicación web con el que se desarrolla este proyecto, los documentos necesarios para proveer un buen servicio multiplataforma excluyen los ámbitos de instalación y configuración. Poniendo mayor énfasis en las ayudas de utilización.

5.3.1 MANUAL DE USUARIO

Este documento se identifica con una guía que el usuario del sistema pueda consultar en caso de no saber a simple vista cómo interactuar con él, o de ser la primera vez que se dispone a utilizarlo.

Este es un documento claro y conciso que describe de una forma didáctica y, por tanto tutorial, todos aquellos conocimientos que un usuario cualquiera del sistema debe tener en cuenta para sacarle el máximo rendimiento posible.

Como eje fundamental del documento se describen todas aquellas operaciones que el usuario puede realizar sobre el sistema, usando el vocabulario adecuado para su entendimiento en niveles no técnicos. Las descripciones sobre dichas operaciones han de ser breves para fomentar la capacidad de atención del usuario. También se deben especificar todos los términos que intervienen en cada operación con los términos explícitos con que estos elementos aparecen en las distintas interfaces de la aplicación.

Dado el carácter claro y conciso del documento, su longitud no debe excederse. Se podría considerar que en una descripción bien espaciada y diferenciada entre las distintas funcionalidades, no se debe exceder de

una página por conjunto de funcionalidades de una interfaz de la aplicación.

Una vez especificados los conceptos fundamentales que marcan la pauta a seguir en el Manual de Usuario, cabe destacar la necesidad de un índice sencillo en la primera parte del mismo que indique el nombre de las distintas funcionalidades del sistema que se pueden consultar y la página en la que se encuentran. Es muy importante no cargarlo de múltiples subapartados o poca claridad en la especificación del lugar al que acudir. Su uso normal se caracteriza por accesos rápidos a conceptos concretos. (Alegsa, 2010)

Para concluir esta especificación, es imprescindible definir la creación de un glosario que describa todos aquellos términos que puedan llevar a confusión o establecer ambigüedad con otros términos del manual. Se ubicará en la parte final del mismo.

Como formato de uso del manual se define el formato “pdf” ya que es universal a cualquier sistema operativo y facilita su impresión, siempre maquetada en DIN-A4.

5.3.2 AYUDA EN LÍNEA

El uso de aplicaciones web conlleva un índice de pérdida de personalización en los productos que deben ser cubiertos con un buen servicio. Este es el cometido de la ayuda en línea.

En caso de ser comercializado, el sistema debe proveer un sistema al cual cualquier usuario pueda acceder para resolver conflictos o dudas en su interacción con el producto.

El desarrollo de este elemento puede convertirse en un proyecto en sí mismo, pero su disponibilidad equilibra el índice de trabajo adicional y el nivel de satisfacción de los usuarios.

El primer prototipo que se desarrolla no contempla la creación de este servicio, pero en caso futuro esta se mostrará sobre la interfaz como un botón de nombre “ayuda” en la esquina superior derecha. Deberá ser discreto y poco perceptible al usuario en su navegación común, pero visible para cuando este sea requerido.

Con pulsar dicho botón se abrirá una pantalla sobre la aplicación, pero sin salir de ella, con una barra de texto en la que el usuario podrá especificar su problema. Esta información, una vez introducida, le llevará a la página web del de la empresa encargada del mantenimiento de la aplicación y lo conectará con un técnico.

6

Análisis de riesgos

Se identifican los riesgos del proyecto como cualquier evento o circunstancia cuya probabilidad de ocurrencia es incierta pero que, en caso de aparecer, tiene un efecto significativo sobre los objetivos del proyecto (Oliveira, 2017).

Atendiendo a este concepto, es necesario identificar con el mayor grado de certeza posible cuáles son estos posibles eventos a tener en cuenta y así poder elaborar una estrategia acertada para su tratamiento.

Según la probabilidad de que estos riesgos ocurran, el grado de negatividad que aporte al proyecto y la prioridad del objetivo al que afecte se opta por un tratamiento de aceptación, una estrategia de mitigación o la aplicación de un plan de contingencia.

El objetivo final de la aplicación reside en proporcionar un funcionamiento que cumpla sus criterios funcionales y no funcionales. Por lo tanto, la identificación de riesgos y la definición de su tratamiento se hace indispensable y colabora en la extracción de incertidumbre a la hora de cumplir dichos criterios.

6.1 DEFINICIÓN DE RIESGOS

Dentro de la definición de los riesgos concretos a tratar y teniendo en cuenta el carácter no comercial del proyecto, estos se subdividen en dos categorías:

- **Riesgos del proyecto:** Los riesgos del proyecto identifican los problemas potenciales de presupuesto, planificación temporal, personal (asignación y organización), recursos, clientes, requisitos y su impacto sobre el presente proyecto.
- **Riesgos técnicos:** Los riesgos técnicos identifican problemas potenciales de diseño, implementación, de interfaz y de mantenimiento. Además, la incertidumbre técnica, el uso de técnicas anticuadas y las tecnologías demasiado avanzadas son también causa de nacimiento de este tipo de eventos. Los riesgos técnicos están presentes en los problemas que son más difíciles de resolver de lo que se pensaba.

La prioridad de los objetivos funcionales del proyecto es un factor fundamental a la hora de establecer una estrategia u otra frente a un riesgo. Como ya se definió previamente a los requisitos funcionales, la escala de prioridad de características de este proyecto es: Muy baja, Baja, Alta, Muy alta.

Además, el siguiente factor que complementa a la prioridad en la elección del plan de acción con el que tratar un riesgo es su probabilidad de ocurrencia. En este proyecto se escoge la siguiente escala de probabilidad: Muy baja, Probable, Muy alta.

Una vez aclaradas las escalas de probabilidad de ocurrencia, de prioridad y los tipos de riesgos sobre los que se va a concretar su gestión en este proyecto, caben destacar tres conceptos que caracteriza el plan de acción a llevar a cabo.

- **Aceptación:** Es la estrategia de gestión a llevar a cabo cuando se considera que el riesgo tratado tiene una probabilidad de ocurrencia “Muy baja” y la prioridad no es “Muy alta”.
- **Contingencia:** Plan de acción caracterizado por el intento de mantener acotado el efecto del riesgo en vigor. Se opta por esta estrategia siempre que su ocurrencia sea “Probable” o inferior y la prioridad del objetivo afectado no sea “Muy alta”.
- **Mitigación:** Este tipo de plan de acción se encarga de resolver el efecto negativo producido por el riesgo en vigor durante la ejecución del proyecto. Su capacidad de resolución es debido a que afecta a objetivos de prioridad “Muy alta” y su probabilidad de ocurrencia es “Probable” o “Muy alta”.

6.2 PLAN DE GESTIÓN DE RIESGOS

- **Riesgos del proyecto:**

Tabla 13: *Gestión de riesgos del proyecto*

RIESGO	IMPACTO	PROBABILIDAD	PRIORIDAD	ESTRATEGIA
Presupuesto	Aumento del coste de proyecto esperado.	Muy baja	Alta	Aceptación: Se continúa con el desarrollo y se comunica el efecto al promotor.
Planificación temporal	Aumento del tiempo de desarrollo.	Muy baja	Muy alta	Mitigación: Se vuelve a planificar la temporización del proyecto y se acuerdan nuevos plazos.

Personal	Disminución de personal inesperada (renuncias, bajas, etc.).	Baja	Alta	Contingencia: Se aumenta en una hora el tiempo de trabajo del personal restante.
Recursos	Faltan recursos de desarrollo.	Probable	Muy alta	Mitigación: Se incorpora nuevo personal con experiencia.
Impacto del producto	No aceptación del sistema final.	Probable	Muy alta	Mitigación: Se elaboran nuevas etapas iterativas y se vuelve a planificar la temporización del proyecto.

Para elaborar esta gestión de riesgos se ha considerado que la experiencia en tareas de estimación del equipo de proyecto es elevada y, por lo tanto, su probabilidad de fallo no es alta.

- **Riesgos técnicos:**

Tabla 14: *Gestión de riesgos técnicos*

RIESGO	IMPACTO	PROBABILIDAD	PRIORIDAD	ESTRATEGIA
Diseño	Un objetivo no se puede implementar por fallos en consideraciones previas.	Muy baja	Muy alta	Mitigación: En la siguiente iteración de desarrollo se vuelve a diseñar la característica
Implementación	Para cierto camino de ejecución, una funcionalidad falla.	Muy alta	Alta	Contingencia: Se proponen vías alternativas de ejecución y se espera para su mitigación en tiempo de mantenimiento.
Incompatibilidad	La aplicación no ejecuta correctamente en ciertos navegadores	Muy baja	Alta	Contingencia: Se abre un enlace a navegadores que permiten una ejecución correcta del sistema.

Uso excesivo de recursos	El procesamiento de la aplicación hace inservible el desarrollo de otras tareas en el dispositivo, o su consumo de batería es muy alto.	Probable	Muy alta	Mitigación: Se planifican nuevas iteraciones de desarrollo en las que se ponga como objetivo la mejora de la eficiencia de procesamiento.
Sin conexión a la red	El dispositivo pierde la conexión a la red durante su navegación.	Probable	Alta	Contingencia: Se muestra un mensaje de aviso y se detiene el procesamiento interno de la aplicación, manteniendo la interfaz congelada.
Mantenimiento	El servidor deja de aceptar modificaciones de código.	Muy baja	Muy alta	Mitigación: Se remodela la compilación del proyecto y se busca su portabilidad a nuevos servidores.

Fallo en geolocalización	El sistema no puede acceder a la posición del usuario.	Probable	Alta	Contingencia: Se avisa al usuario y se ofrece la posibilidad de insertar su ubicación manualmente.
Caída de la aplicación	El sistema deja de funcionar inesperadamente.	Muy baja	Muy alta	Mitigación: Cuando se detecta un fallo de servicio, la aplicación se reinicia automáticamente.



7

Modelado y diseño del sistema

Tras una exposición de todos los conceptos teóricos necesarios para comprender la motivación del proyecto, su alcance y el conjunto de requisitos a tener en cuenta para su desarrollo, se procede al modelado y diseño del mismo.

Tal y como el resto del proyecto, esta fase se desarrolla de forma incremental; exponiendo los distintos tipos de modelos tenidos en cuenta, con un grado mayor de detalle cada vez. Finalmente, el resultado mostrado se identifica con el diseño final del sistema “2Train – Análisis deportivo de la ciudad de Málaga”.

Para llevar a cabo el desarrollo de esta fase, se ha decidido modelar el proyecto desde dos puntos de vista complementarios: Modelo del dominio y Modelado del sistema. Cabe destacar que para el diseño del primer apartado mencionado se aplica el lenguaje de modelado UML, ya que aporta las características suficientes para completarlo. Sin embargo, para el modelado general del sistema se hace uso de un perfil para UML que extiende el lenguaje: MagicUWE (Maximillian University Munich, 2010).

MagicUWE es una extensión para modelado web basada en UML. Puesto que el sistema que se desarrolla es una aplicación web multiplataforma, se ha decidido diseñarla con las herramientas apropiadas que extienden sus posibilidades de modelado hacia un entorno web. Por ello, a parte del modelado general que permite UML, se complementa el diseño con modelos específicos, como el de navegación y procesos (integrados en un mismo diagrama), el modelado tentativo de interfaz del sistema y los diagramas de flujos de procesos asociados a los procesos definidos en la navegación.

7.1 CASOS DE USO

Con el diseño de casos de uso del sistema comienza el modelado en desarrollo.

Este apartado es ideal para dar comienzo a esta fase, ya que el diagrama de casos de uso representa la forma en la cual un usuario (Actor) opera con el sistema en desarrollo.

En el contexto del modelado en UML y por tanto de este proyecto, un caso de uso es una tarea específica que se realiza tras una orden de algún agente externo con el que está relacionado, sea desde una petición de un actor o bien desde la invocación desde otro caso de uso. En definitiva, estos se van a identificar con las características funcionales del sistema.

El diagrama que modela este aspecto se muestra a continuación, en la Figura 1:

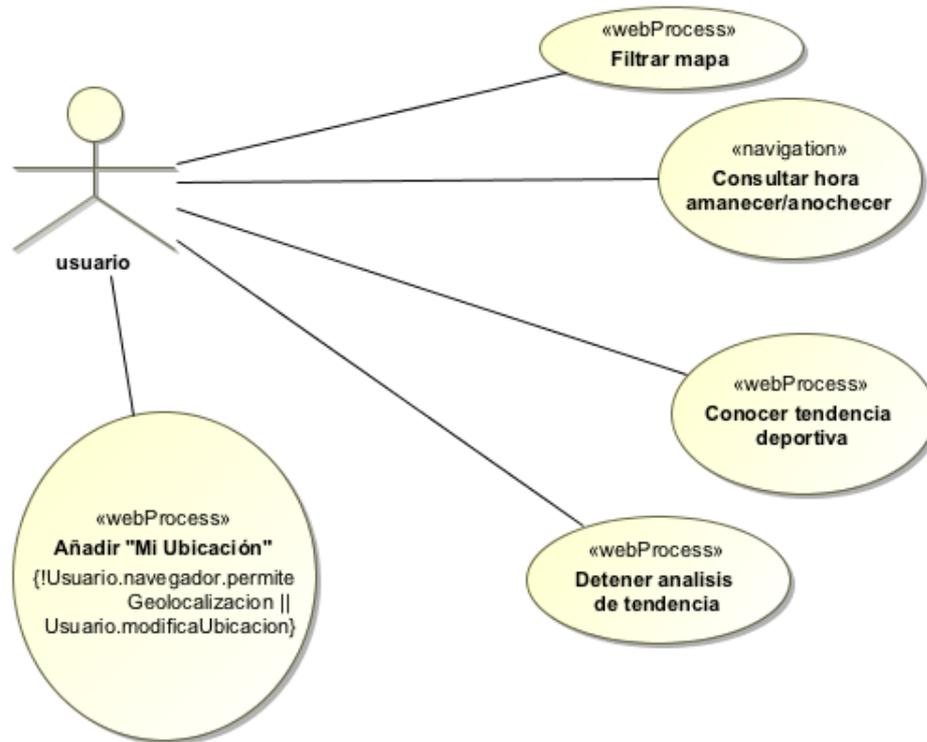


Figura 1: Diagrama de casos de uso

Dada la simplicidad de concepto del sistema, los casos de uso solo contemplan la interacción con el único actor externo: el usuario. Este será capaz de filtrar el mapa que caracteriza la interfaz según diferentes criterios de temática o distancia, consultar la hora de amanecer y anocheecer de la ciudad de Málaga, seleccionar deportes de los que conocer su tendencia a ser tratados en Twitter, detener el procesado correspondiente al análisis de tendencia y añadir ubicaciones al sistema de forma automática y manual.

Como se ha especificado al comienzo, se identifica cada caso de uso con las capacidades del usuario en el sistema y con los requisitos funcionales que harán útil el cumplimiento de su objetivo global: facilitar la decisión a la hora de decidir la ubicación y el deporte a realizar en la ciudad de Málaga.

Dada la extensión que propone MagicUWE sobre UML, se aprecian dos estereotipos diferentes en los casos de uso:

- **Navigation:** Identifica a las acciones de usuario que conllevan una muestra por pantalla. Buen ejemplo de ello es “Consultar hora de amanecer/anocheecer”.
- **Web process:** Identifica a los casos de uso cuya interacción con el usuario conlleva procesamiento interno del sistema. Son ejemplos de ello: “Añadir “Mi ubicación”” o “Detener análisis de tendencia”. Estos casos de uso identifican a los procesos a tener en cuenta en el sistema.

La única restricción adicional para la ejecución de un caso de uso es la impuesta a “Añadir “Mi ubicación””, la cual será ejecutada si el navegador no permite la geolocalización automática del usuario o si este quiere cambiar su ubicación sobre el mapa.

7.2 MODELO DEL DOMINIO

El modelo del dominio del sistema intenta modelar el contexto de funcionamiento del sistema en términos de sus entidades, objetos del mundo real con los que interactúa y acontecimientos más relevantes.

Para mostrarlo, la Figura 1 expone un diagrama estructural en el que se identifican las clases que ejecutan los sucesos más importantes en el sistema, los objetos con los que estas entidades interactúan y que existen en el dominio y en el proyecto como negocio y el conjunto de sucesos que se dan entre ellos. Estos sucesos vienen dados como relaciones entre entidades y objetos, nombradas de forma intuitiva respecto al evento que desarrollan.

Para diferenciar una entidad de un objeto, estos últimos adquieren la regla de nombramiento: “:<nombreObjeto>”, mientras que una clase del sistema se nombra: “<NombreClase>”. Esto se muestra en la Figura 2 expuesta a continuación:

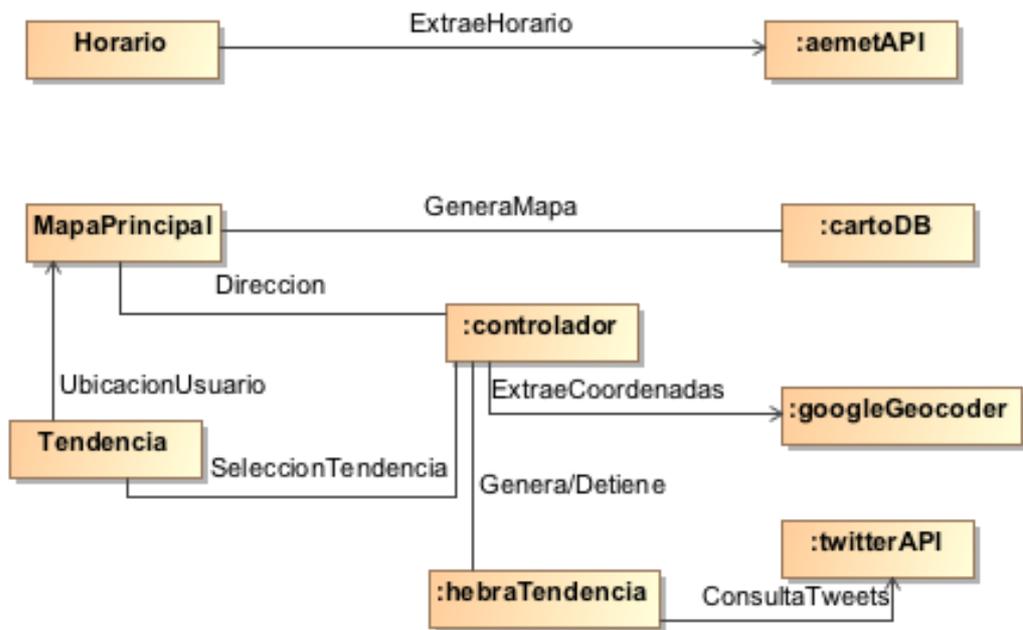


Figura 2: Modelo del dominio

Como se puede observar, se identifican como objetos y entidades del sistema a nueve elementos.

La entidad “Horario” es la más desacoplada del contexto, ya que solo interactúa con la instancia de la API de AEMET para extraer el horario de amanecer y anochecer de la ciudad.

“Mapa principal” es la entidad del sistema que opera sobre el mapa mostrado. Esta se comunica con la instancia de la herramienta cartográfica “Carto” para extraer el mapa que requiere. También se comunica con la instancia operativa del controlador en ese momento para llevar a cabo del proceso de geocodificación de una dirección. A su vez, dicha instancia del controlador se comunica con una instancia de la librería de Google que le ofrece este servicio de direcciones y coordenadas.

Por último, se define el dominio de la acción que más opera sobre el contexto que le rodea. La entidad “Tendencia” extrae de “MapaPrincipal” la ubicación del usuario y comunica al controlador la selección deportiva del usuario sobre la cual calcular la tendencia. Es el controlador quien

crea una instancia en segundo plano que opere el procesamiento sobre la API de Twitter y calcule el análisis.

7.3 MODELADO ESTRUCTURAL: DIAGRAMA DE CLASES

El modelado estructural del sistema se identifica con su diagrama de clases. En este se muestran las entidades que operan en el sistema para llevar a cabo los procesos necesarios, los atributos necesarios para completarlos, las relaciones existentes entre ellas para que exista la comunicación necesaria para la invocación de procesos y las operaciones que los componen.

Estas operaciones son invocadas directamente por el usuario al operar sobre una clase o por otra clase del sistema que tenga relación con la poseedora de la operación. Por lo tanto, aunque se diseña el modelado estructural, esta identificación de las relaciones entre entidades unida a la exposición de sus operaciones acercan el diseño a su modelo de comportamiento.

Esto se muestra a continuación el diagrama de clases de la Figura 3:

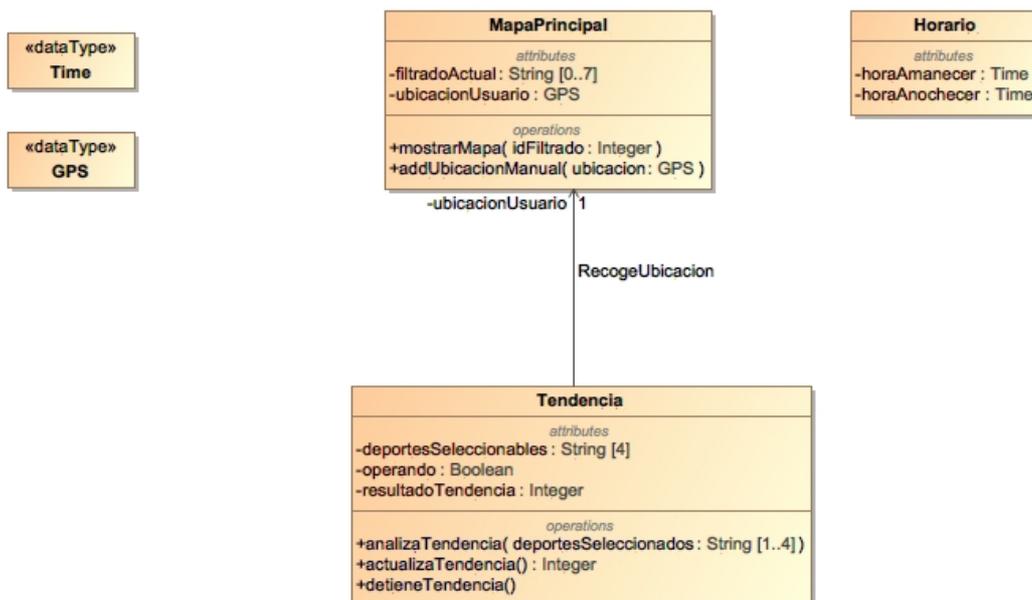


Figura 3: *Diagrama de clases*

Conceptualmente y a nivel de interacción el sistema propuesto es simple. Por ello, el modelado estructural viene caracterizado por tres clases: “Mapa principal”, “Tendencia” y “Horario”. Cada una de ellas se identifica con la consecución de uno de los tres objetivos prioritarios del sistema.

- **MapaPrincipal:** Esta clase es la encargada de la exposición y gestión de todo lo concerniente a la geolocalización y tratamiento del mapa.

Tiene dos atributos: “filtradoActual” y “ubicacionUsuario”. Con el segundo atributo se tiene o bien la ubicación automática que se recoge y muestra del usuario o la ubicación manual editada por él mismo. Para facilitar su gestión, se ha definido un nuevo tipo de dato “GPS” que conlleva el tratamiento de sus coordenadas y el formato a comunicar a la API “carto.js”. El tipo del primer atributo se corresponde con un array de String. Dichos valores String contienen las sentencias SQL preestablecidas, una por cada capa del mapa, con las que se ejecuta cada uno de todos los posibles filtrados sobre el sistema. De forma derivada, estos pueden componerse al ejecutar más de un filtrado simultáneamente. Estas sentencias hacen la petición a la base de datos de Carto y la posterior modificación del mapa se haga de forma más simple.

Se completa la definición de esta clase tan relevante en el sistema con la especificación de sus operaciones:

- “mostrarMapa(id:Integer)”: Es la operación a la que se invoca cuando un usuario selecciona un tipo de filtrado u otro. Esta selección le pasa a la operación un identificador de tipo Integer para seleccionar la sentencia SQL (atributo “filtradoActual”) a tratar con Carto.
- “añadirUbicacionManual(ubicación:GPS)”: Esta operación se ejecuta automáticamente al recibir la ubicación actual del usuario o cuando este la modifica manualmente. Recibe las coordenadas de ubicación desde su ordenador o desde un

método privado que obtiene las coordenadas a partir de una dirección postal sobre un servicio web externo. Con dichas coordenadas el sistema opera sobre la capa de ubicación del usuario del mapa a través de “carto.js”.

- **Tendencia:** Esta clase conlleva el segundo objetivo más relevante del sistema. Será la encargada de gestionar el análisis de tendencia deportiva sobre Twitter que indique el usuario.

Su atributo “deportesSeleccionables” almacena el nombre de los deportes que el usuario selecciona para conocer su tendencia. Este atributo es vacío al principio y puede contener hasta los cuatro deportes preestablecidos en la interfaz.

El atributo “operando” es un booleano que le indica al sistema si el usuario desea que el procesamiento se esté llevando a cabo o si quiere que este se detenga.

Por último, el atributo “resultadoTendencia” es un Integer que muestra el porcentaje de tuits analizados con coincidencia de palabras clave respecto a la selección de deportes actual.

Sus operaciones mantienen y ejecutan el contenido de sus atributos. Estas son:

- “analizaTendencia(deportesSeleccionados)”: Inicia el análisis de tendencia al crear una hebra en segundo plano que consulta un flujo de trescientos tuits actuales por minuto en un radio de cien kilómetros alrededor de la posición actual del usuario y en comparación con las palabras claves asignadas al listado de deportes seleccionados. También modifica la selección deportiva si la hebra ya estaba operando.
- “actualizaTendencia()”: Esta operación se realiza de forma periódica y automática cada treinta segundos, y comunica al objeto cliente el valor de tendencia a mostrar. Este valor, junto

con otras variables se mantienen de forma concurrente entre la hebra principal y la hebra en segundo plano.

- “**detieneTendencia()**”: Esta operación cierra el ciclo de operación de la hebra que analiza la tendencia en Twitter. Puede ser invocada si se deseleccionan todos los deportes de la lista o también si se pulsa el botón “Stop”. Como puede ser reanudada, esta detención mantiene la hebra en una espera no activa que no consuma recursos del sistema.
- **Horario**: Esta clase únicamente mantiene almacenados los horarios de amanecer y anochecer de la ciudad de Málaga, los cuales recoge al inicio de la aplicación desde un servicio web externo. Para mantener sus atributos, que serán horas, se define el tipo de dato “Time”.

Esta última clase se mantiene de forma independiente a las demás. Sin embargo, aunque el sistema no concede relaciones complejas tales como herencias o agregación, existe una relación unidireccional entre “Tendencia” y “MapaPrincipal”. Esta se da única y exclusivamente con la única instancia de MapaPrincipal, ya que la aplicación no instanciará más de un objeto de esta clase. La relación se ha denominado “RecogeUbicacion” ya que es la relación que permite a la “Tendencia” obtener la ubicación actual del usuario a través del atributo “ubicaciónUsuario” de “MapaPrincipal”, y así poder realizar su análisis en el radio adecuado respecto a la posición del usuario.

7.4 MODELO DE NAVEGACIÓN

El diagrama de navegación es una incorporación que hace MagicUWE al modelado de UML. Se ocupa de la estructuración del contenido en nodos y enlaces para mostrar la forma en que un usuario procede sobre el sistema. Se identifica como contenido las clases tenidas en cuenta como modelado estructural, los procesos que estas llevan a cabo para alcanzar sus objetivos y las relaciones que se da entre ellos.

En este caso, las clases definidas se aplican sobre el estereotipo “navigationNode” y los procesos con el estereotipo “processClass”. Por último, el diagrama establece las relaciones existentes entre dichos componentes, alterando sus nodos en términos de cómo este se estructura en la navegación web real de la aplicación. Estas relaciones tienen dos cometidos diferentes: “ProcessLink” relaciona un proceso con cualquier otro componente y “NavigationLink” relaciona nodos y componentes distintos de procesos.

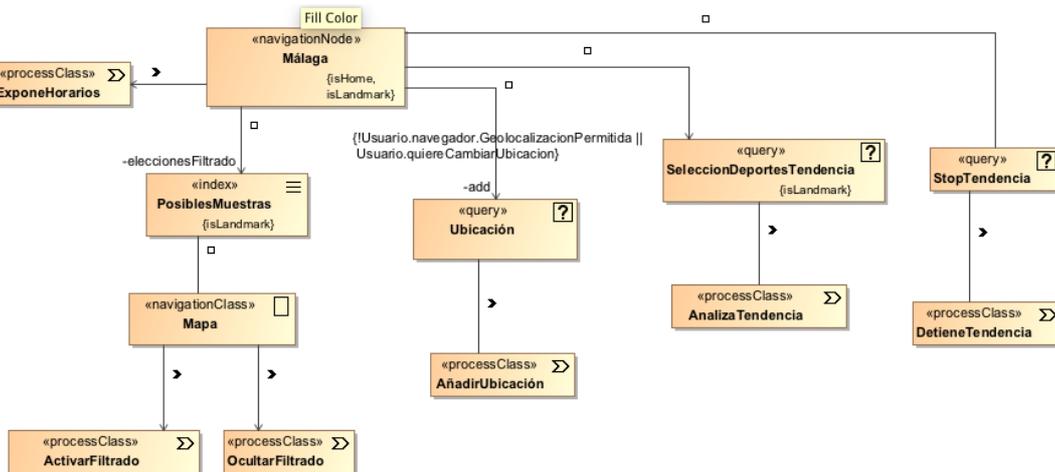


Figura 4: Diagrama de navegación

Los procesos se identifican con los objetivos a cumplir por la aplicación. Más detalladamente, estos procesos son paralelos a los casos de uso del sistema.

El punto central de la navegación de la aplicación “2Train – Análisis deportivo de la ciudad de Málaga” es el nodo de navegación que se identifica con el inicio de la aplicación: “Málaga”. Es el componente que envuelve al resto de contenido que compone el sistema. Dado que se quiere mantener un acceso simple e interacciones rápidas, la aplicación está diseñada para tener un único punto de entrada y no presentar distintas interfaces, ni recargas. Por ello, este componente genérico es “Home” y “Landmark”. Es decir, es el punto de inicio de la aplicación y es accesible en todo momento. Los otros componentes “Landmark” son “Posibles muestras” y “Selección deportes tendencia”. Ambos son accesibles en todo momento de la navegación.

Este inicio de aplicación tiene una dirección directa (processLink) con el proceso que expone los horarios de amanecer y anochecer.

Antes de continuar, explicar que un componente “index” muestra un índice con múltiples selecciones posibles, mientras que un “query” expone un campo que requiere información o acción del usuario para desencadenar un proceso.

Una vez cargada la página, las relaciones ofrecen distintas posibilidades:

- **Índice “Posibles muestras”:** Muestra el componente de navegación que permite al usuario seleccionar uno o más botones de filtrados sobre el mapa. Este índice interactúa directamente con el nodo “Mapa”, el cual ejecutará el proceso activar u ocultar filtrado, dependiendo del estado del botón pulsado.
- **Query “Ubicación”:** Este componente propone un campo de texto en el que introducir una dirección con la que cambiar la ubicación del usuario. Es accedida si dicho usuario quiere cambiar su ubicación, o su navegador no permite la geolocalización. Evidentemente, interactúa a través de otro “processLink” con el proceso “Añadir ubicación”. Este campo solo será accesible si el usuario pulsa un botón que permita su visualización, previamente.
- **Query “Selección deportes tendencia”:** Como query, el componente requiere la introducción de contenido. En este caso se trata de la exposición de un listado con las opciones: “Correr”, “Ciclismo”, “Nadar” y “Otros”, de las cuales el usuario puede seleccionar las que desee. Esta selección desencadena el proceso de análisis de tendencia.
- **Query “Stop tendencia”:** Este componente es un botón que si es pulsado detiene el proceso de análisis de tendencia. Si se pulsa sin estar operativo el análisis no tiene efecto alguno.

7.5 MODELO DE FLUJO DE PROCESOS

Como complemento a todo el diseño del sistema, se muestra el contenido de los procesos que se han mencionado. Se profundiza en cómo se realiza cada proceso y la secuencia de pasos que llevan a cabo para operar.

Aquí se definen nodos de estereotipo “userAction” y “systemAction”. El primero es una acción llevada a cabo por el usuario y el segundo una acción llevada a cabo por el sistema.

- **Activar filtrado:**

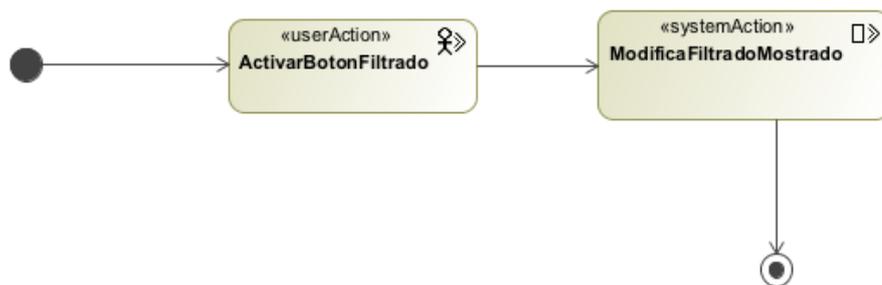


Figura 5: Diagrama de flujo – Activar filtrado

Cuando un usuario decide implementar un nuevo filtrado sobre el mapa, debe activar el botón adecuado al filtrado que quiere realizar y ello desencadena una operación de entidad del sistema que modifica el filtrado a mostrar.

- **Ocultar filtrado:**

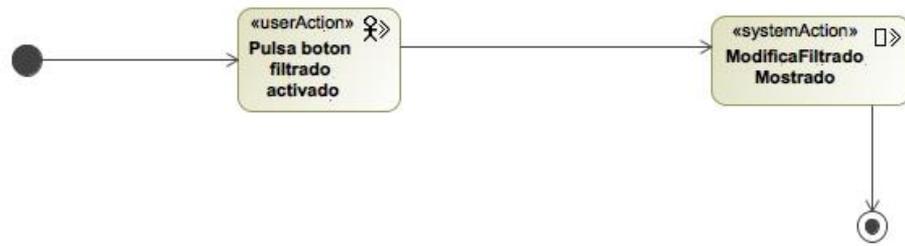


Figura 6: Diagrama de flujo – Ocultar filtrado

Al igual que el proceso anterior, el usuario debe pulsar el botón del filtrado activado que desee ocultar y ello hace que el sistema modifique el filtrado mostrado.

- **Expone horarios:**

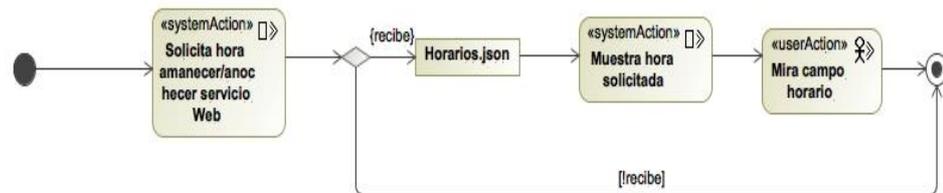


Figura 7: Diagrama de flujo – Expone horarios

Este proceso se desarrolla únicamente a nivel de sistema. El inicio de la aplicación solicita la información necesaria a un servicio web externo. Si recibe la el archivo “.json” esperado, el sistema trata los datos y los muestra en el campo correspondiente. El usuario puede mirar el horario. Si no recibe dicha información, el proceso termina sin que el usuario puede ver el horario. Una nueva recarga de la página reiniciaría el proceso.

- Añadir “Mi ubicación”:

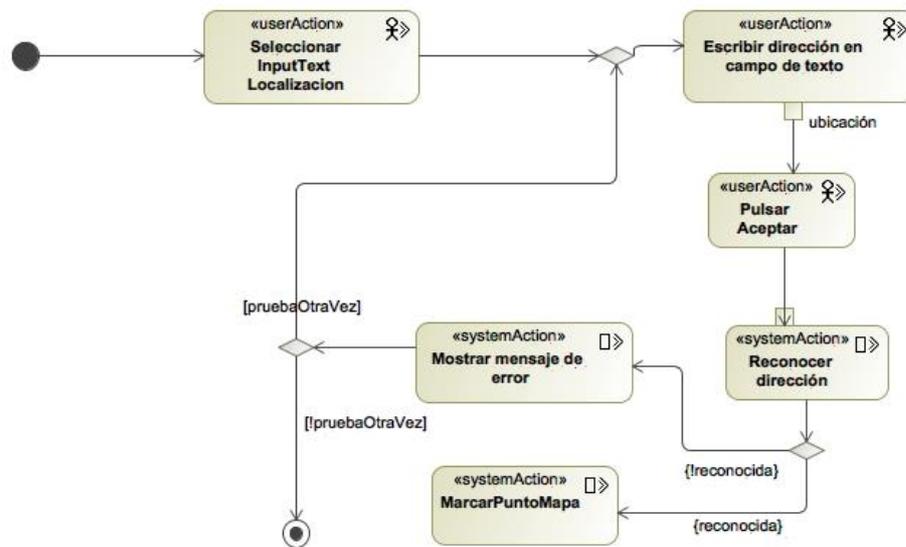


Figura 8: Diagrama de flujo – Añadir “Mi ubicación”

Si un usuario decide modificar su ubicación manualmente, debe seleccionar el campo de texto de dicho apartado y escribir la dirección a la que quiere migrar su posición sobre el mapa. Cuando pulsa aceptar, el sistema reconoce la dirección introducida. Si la dirección es válida el sistema obtiene las coordenadas de la misma y marca el punto sobre el mapa. Si la dirección no es reconocida, se muestra un mensaje de error y, dependiendo de si el usuario quiere volver a intentarlo, termina el proceso o vuelve a introducir la dirección.

- **Analiza tendencia:**

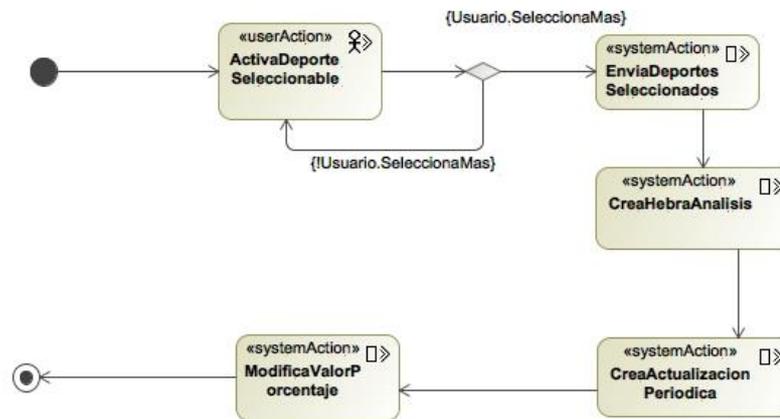


Figura 9: Diagrama de flujo – Analiza tendencia

El proceso se inicia y el usuario activa uno o más deportes de la lista seleccionable. Ello, hace que el sistema envíe la selección al controlador y este crea la hebra en segundo plano. Además, se crea una tarea periódica que consulta el valor de tendencia producido cada treinta segundos y el propio sistema modifica la barra de progreso con dicho valor para mostrársela al usuario.

- **Detiene tendencia:**

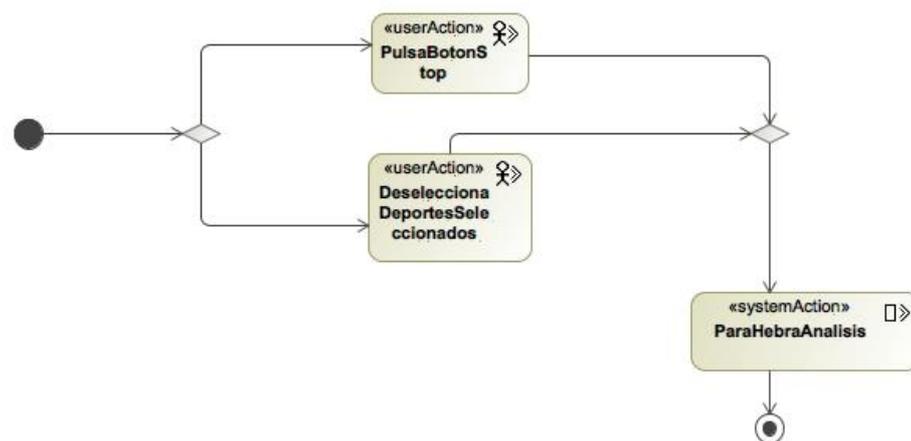


Figura 10: Diagrama de flujo – Detiene tendencia

La detención del proceso de análisis puede darse de dos modos. Pulsando el botón “Stop” que detiene el proceso y deselectiona los deportes automáticamente, o deselectionando todos los deportes seleccionados hasta el momento.

Con la definición esquemática de los pasos a seguir para implementar cada proceso que caracteriza el sistema, queda modelado el flujo de acción del mismo.

7.6 DISEÑO DE INTERFAZ

El diseño de la interfaz completa el modelado de cualquier aplicación web a través de MagicUWE. Es innegable la importancia de una buena interfaz en el seguimiento de un sistema web. Un buen diseño de las interfaces que componen una aplicación puede facilitar su éxito en más de un 45% de las ocasiones (Rodríguez Domínguez, 2018).

Por ello, UML en combinación con MagicUWE permite la definición tentativa de la distribución de los distintos componentes a incorporar sobre la interfaz de la aplicación. Para ello se debe ser coherente al diagrama de navegación que ya se ha realizado.

La Figura 11 expuesta a continuación representa este modelo:

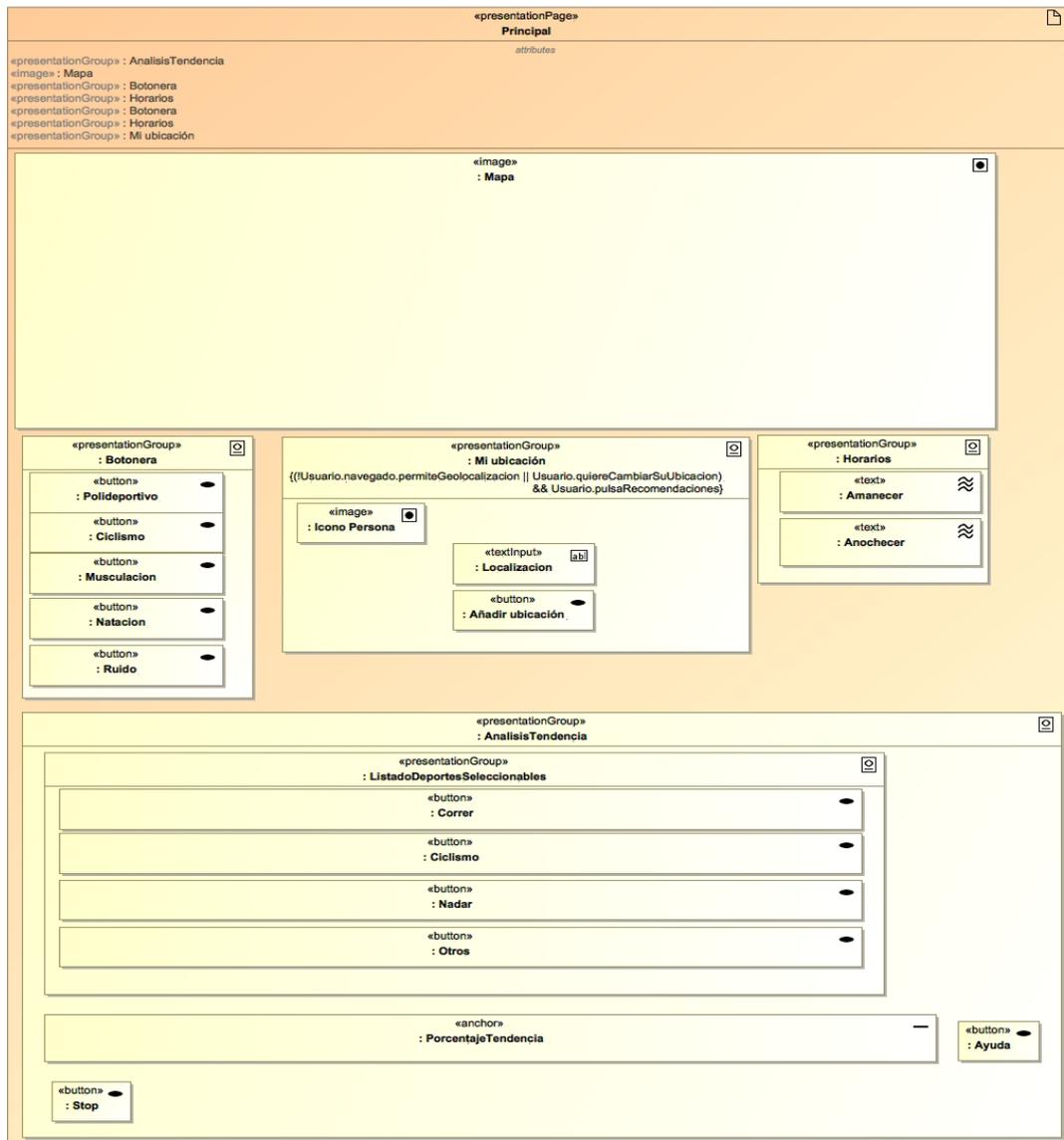


Figura 11: Diagrama de presentación

Dado que el sistema “2Train – Análisis deportivo de la ciudad de Málaga” tiene el objetivo de mostrar una interacción lo más simplificada posible para el usuario, con componentes concretos y acciones bien definidas, se ha optado por la definición de una sola interfaz.

Dicha interfaz mantiene de forma accesible todos los componentes de selección identificados durante este proceso de diseño. Además, ninguna interacción con los componentes de la misma conllevará recarga de la página. Una vez que la aplicación se inicia, las interacciones del usuario con los diferentes botones alcanzarán sus objetivos pero con procesos asíncronos que no ralenticen la conexión de la aplicación.

Presidiendo la interfaz se muestra el mapa de ancho completo. Conformando el núcleo prioritario de la aplicación.

Bajo él, se coloca el índice de filtrado que, a modo de diseño posee una secuencia de botones que permite filtrar el contenido del mapa según distintos criterios. Además, va acompañado del panel de modificación de ubicación, el cual contiene el campo de texto de dirección y el botón de aceptación descritos en su flujo de proceso. También en esa sección se muestra un panel informativo con las horas de amanecer y anochecer de la ciudad de Málaga.

Por último, se mantiene la zona mitad inferior de la interfaz web con el panel dedicado al análisis de tendencia. Está compuesta por una lista seleccionable de deportes de fácil acceso y diseño amplio e intuitivo.

Se pretende informar de la tendencia analizada a través de una barra de progreso ubicada bajo dicha lista de selección y acompañada por un botón de ayuda que clarifique la acción calculada por la tendencia y un botón de stop que pueda detener el proceso de forma inmediata.

7.7 MODELADO DE COMPORTAMIENTO

Aunque el comportamiento del sistema ya se ha tratado con anterioridad, cabe complementarlo con la exposición de los dos componentes básicos de este apartado: Diagramas de secuencia y diagramas de estado.

7.7.1 DIAGRAMAS DE SECUENCIA

Un diagrama de secuencia muestra el intercambio de mensajes en el proceso de interacción entre el usuario y los distintos componentes del sistema para llevar a cabo un proceso.

Si se ha decidido completar el comportamiento del sistema con este tipo de diagrama es por las palabras claves que definen a estos diagramas: “Intercambio de mensajes”. De forma esquemática, se van a exponer los diagramas de secuencia para los tres objetivos fundamentales del sistema.

Se van a reconocer a los componentes: Usuario, Mapa (cliente de la aplicación), “cartoDB” (servicio cartográfico usado), Controlador (servidor del sistema), “GoogleGeocoder” (servicio de geocodificación de direcciones utilizado), Twitter y “Hebra análisis”, estos dos últimos la hebra en segundo plano que se genera para analizar tuits en tiempo real.

- **Visión del mapa:**

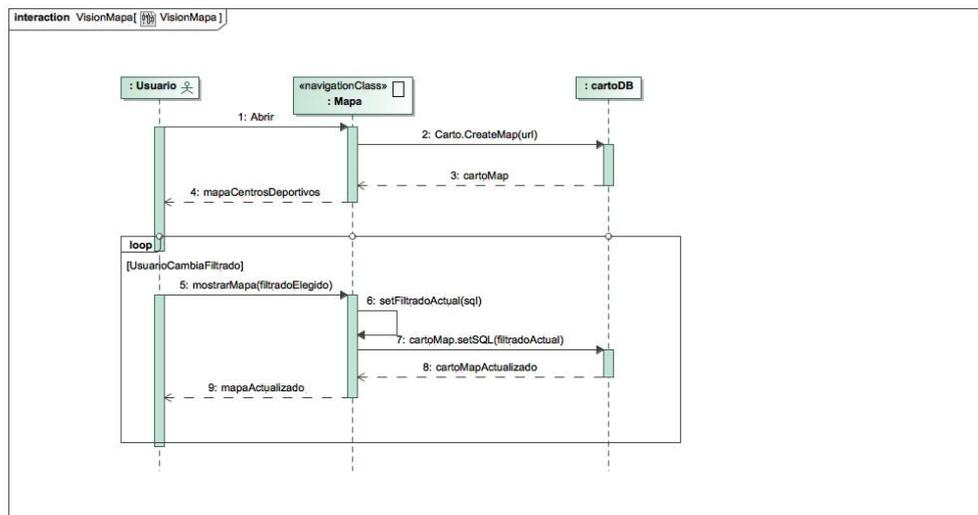


Figura 12: Diagrama de secuencia – Visión del mapa

Un usuario abre la aplicación y automáticamente se pide a la herramienta cartográfica utilizada “Carto” la generación del mapa correspondiente a la cuenta asociada (URL). Ello devuelve el mapa creado por defecto en el perfil de la herramienta y se expone en la aplicación.

Mientras el usuario quiera modificar los filtrados, se invocará a la operación “mostrarMapa” del mapa, esta cambiará el filtrado a nivel SQL en sí misma y hará de nuevo una petición a “Carto” con la que modificará el mapa mostrado.

- **Modificar ubicación de usuario:**

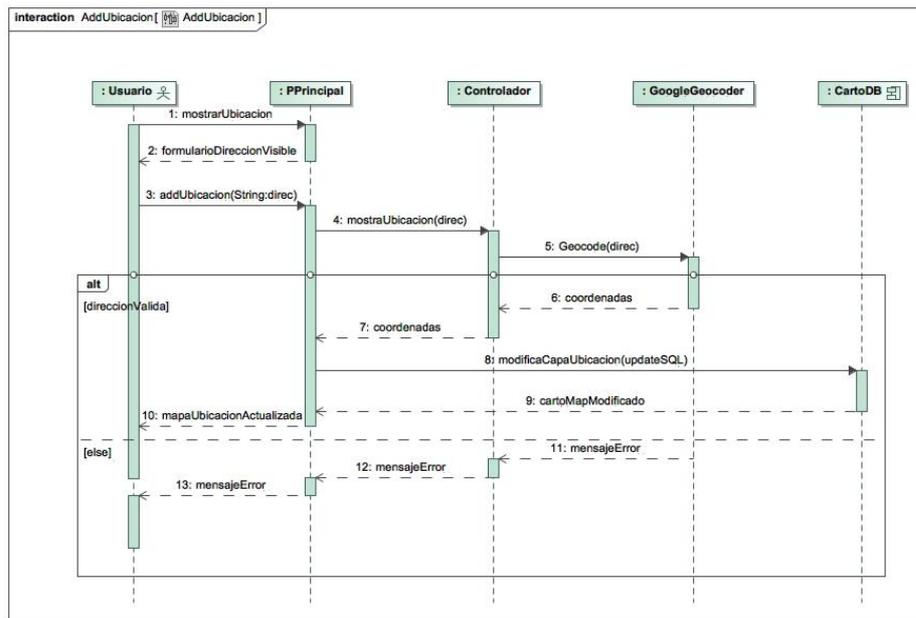


Figura 13: Diagrama de secuencia – Modificar ubicación de usuario

El usuario solicita que se muestre el formulario de ubicación a través de un botón y la página se lo muestra. Con la introducción de la dirección esta pasa del cliente al controlador. El controlador hace “geocoding” con ella para obtener las coordenadas asociadas. Si la dirección devuelve unas coordenadas correctas, estas vuelven a la página y se solicita a “Carto” la modificación de la capa de ubicación del usuario con las nuevas coordenadas”. Se devuelve el mapa con la ubicación modificada.

En caso de que no devolviese las coordenadas, sino una función de error, este mensaje de error devuelto por el servicio de Google se transmite hasta ser mostrado al usuario.

- **Analizar tendencia:**

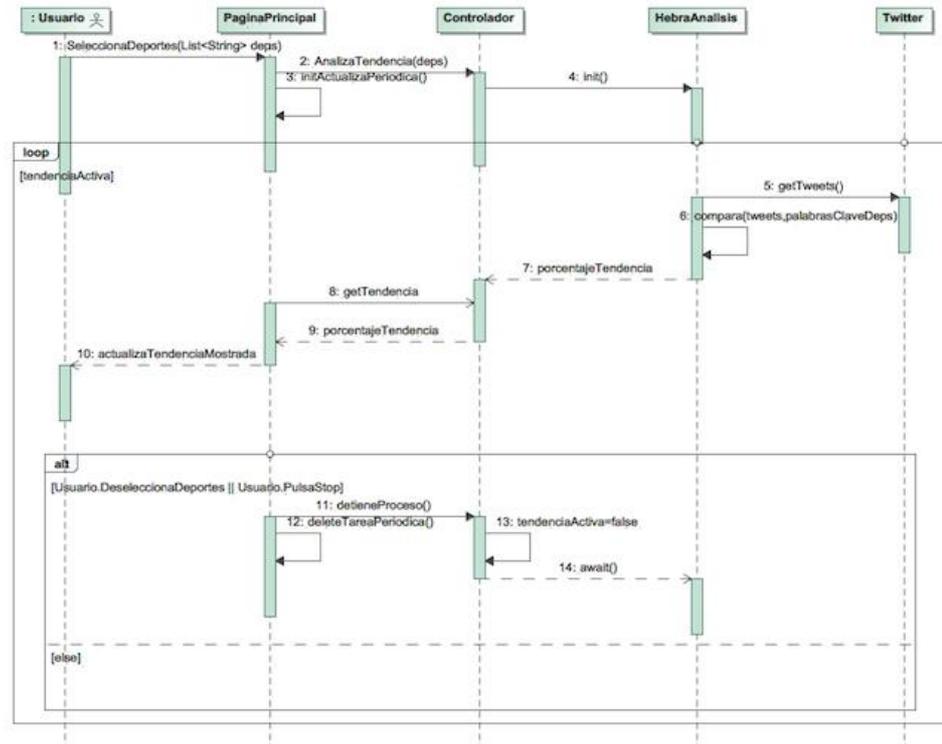


Figura 14: Diagrama de secuencia – Analizar tendencia

Este proceso se inicia con la selección de deportes en la interfaz por parte del usuario. Ello hace que el cliente pase la selección al controlador y este inicie la hebra de análisis en segundo plano. Además, el cliente inicia la tarea periódica que consulta el valor de tendencia generado.

Mientras el análisis está activo, la hebra recoge tuits periódicamente a través de Twitter API y los compara con un conjunto de palabras claves que tiene almacenadas referentes a los deportes seleccionados en ese momento. En cada interacción devuelve un valor de tendencia al controlador y la tarea periódica del cliente pide y recoge dicho valor para actualizar la vista de la barra de progreso que llega al usuario.

Si el usuario detiene el proceso. El cliente avisa al controlador de la acción y detiene su tarea periódica. El controlador modifica una

variable de consulta que usa la hebra de forma concurrente para mantenerse activa o no, y esta queda parada en un monitor.

7.7.2 DIAGRAMAS DE MÁQUINAS DE ESTADO

Dada la simplicidad que se le otorga a la aplicación en desarrollo, no se mantiene un flujo de procesos y clases representativo para la generación de diversas máquinas de estado. Sin embargo, sí es interesante analizar este tipo de modelado de comportamiento en el proyecto y, sobre todo, sobre el proceso de análisis de tendencia que lo compone.

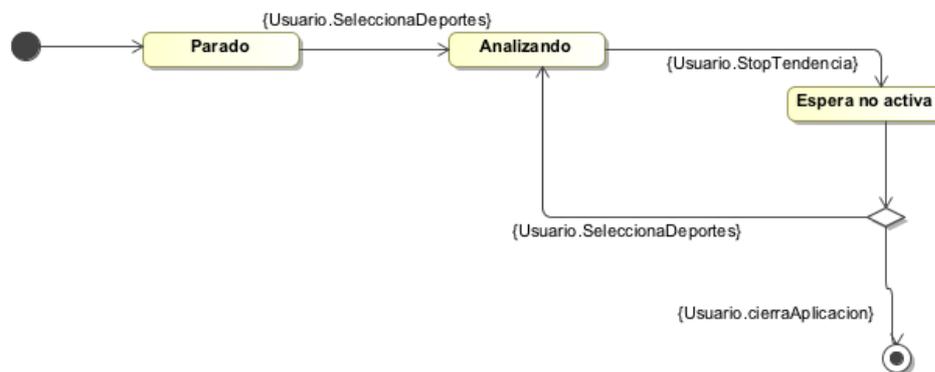


Figura 15: *Diagrama de estados – Analizar tendencia*

El análisis de tendencia del sistema es el único componente que atraviesa distintos estados a lo largo de su ciclo de vida. Por ello, se define su comportamiento para los estados de "Parado", "Analizando" y "Espera no activa".

Una vez se inicia la aplicación, el análisis permanece "Parado". Si el usuario hace alguna selección deportiva para este apartado, da comienzo al proceso y pasa a estado "Analizando". Por último, si el usuario decide detener el proceso de análisis, este pasa a un estado de "Espera no activa" del que se podrá salir de dos formas: Haciendo una nueva selección de deportes que lo devuelva al estado "Analizando" o cerrando la aplicación y, por tanto, destruyendo el componente.

8

Implementación

A continuación se describen todos los detalles de implementación tenidos en cuenta para desarrollar este proyecto.

8.1 TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS

En este apartado se numeran y definen resumidamente todas las tecnologías y herramientas usadas, a nivel de implementación, para la consecución del proyecto. Posteriormente, dichas herramientas se explican con mayor grado de detalle en el apartado de implementación que le corresponde.

- **NetBeans:** Entorno de desarrollo web centrado en el lenguaje Java y utilizado para la programación del proyecto elegido.
- **Maven:** Tecnología de gestión de proyectos Java en la que las librerías enlazadas son aplicadas desde un repositorio web propio. Un proyecto generado sobre dicha tecnología se caracteriza por contener sus recursos especificados en el archivo “pom.xml”. Gracias a su versatilidad de enlace, resuelve todas las dependencias existentes en los recursos utilizados en tiempo de ejecución. Tipo de proyecto web que conforma el proyecto. (Garzas, 2014)

- **Servicios REST:** Tipo de servicios web que ejecuta sus peticiones a través de la interfaz HTTP. Usa este protocolo a modo de API y provee la información requerida a través de las operaciones predefinidas: GET, POST, HEAD o PUT. Además, devuelve los resultados en formatos de intercambio ligeros, como JSON, o de etiquetado estándar, como XML (Luque Baena, Arquitecturas REST, 2017).
- **JSON:** Es un formato de intercambio ligero cuyo objetivo es reducir la sobrecarga de la comunicación debida al uso de etiquetas en los lenguajes de marcado. Representa una colección de pares nombre/valor que van internos en una lista ordenada de valores. En este proyecto, este formato se usa para toda petición realizada a servicios web REST.
- **CartoDB:** Herramienta de cartografía web en crecimiento exponencial y con dedicación sobresaliente a la gestión de datos abiertos. Herramienta cartográfica utilizada.
- **PostgreSQL:** Base de datos utilizada por la herramienta “CartoDB” en el almacenamiento de datos cartográficos de sus usuarios registrados.
- **Javascript:** Lenguaje de programación interpretado, imperativo y débilmente tipado que facilita el tratamiento de páginas web dinámicas, gracias a su modificación de contenido en tiempo de ejecución. Es ejecutado en el lado cliente. Lenguaje utilizado sobre la interfaz web (Luque Baena, Javascript, 2009).
- **JQuery:** Framework de Javascript que facilita diversas funcionalidades del lenguaje, tales como: acceso a los elementos del DOM, agregar interacción a la tecnología Ajax, etc. Framework utilizado en este proyecto sobre el javascript dedicado a realizar peticiones Ajax.
- **Ajax:** Solución para una comunicación asíncrona entre un navegador y un servidor web.

- **Spring Framework:** Framework de Java que abstrae muchas de las funcionalidades ofrecidas por el lenguaje a un grado menor de detalle y proporciona directivas para la ejecución de procesamiento web simplificado. Framework utilizado en este proyecto para reducir la complejidad y ampliar el nivel de abstracción del lenguaje Java.
- **Java:** Lenguaje de programación de propósito general, orientado a objetos y concurrente usado como base para el back-end del proyecto.
- **Programación concurrente:** Tipo de programación orientada a diversas hebras y procesos, que se encarga de asegurar la correcta sincronización y protección de los datos al ser accedidos por dichas hebras y procesos de forma paralela.
- **Twitter API:** Librería de código abierto ofrecida por la red social Twitter para permitir interactuar a aplicaciones externas con su plataforma.
- **Google Maps API:** Librería ofrecida por la compañía Google que permite la integración de servicios cartográficos y de geocodificación a aplicaciones externas. Herramienta utilizada para el proceso de geocodificación de direcciones de forma dinámica.
- **Bootstrap:** Framework orientado al diseño web originalmente creado por Twitter que permite la creación de interfaces web a través de componentes CSS y Javascript. Tecnología utilizada en este proyecto para implementar un diseño web “responsive”.
- **Platform as a Service (PaaS):** Tecnología de nivel intermedio del Cloud Computing que se encarga de facilitar el despliegue de aplicaciones en servidores propios. En este nivel de tecnología, el cliente solo se ocupa de ofrecer un código fuente adaptado a las restricciones de la compañía Cloud elegida. No se requieren elementos de configuración de servidor, tales como: seguridad, rendimiento o capacidad de tráfico permitida. Opción de despliegue

elegida para publicar el proyecto en desarrollo. Concretamente se hace uso de los servidores “Heroku”.

- **GitHub:** Es una plataforma de desarrollo colaborativo de software que permite alojar proyectos utilizando el sistema de control de versiones Git. Su uso en base a la versión final de este proyecto lo hace de código abierto.
- **CDN:** Tecnología que redistribuye localmente el contenido especificado en cualquiera de sus servidores operativos, con la información adicional necesaria para hacerlo ejecutable. El sistema de servidores acerca el contenido a los distintos dispositivos solicitantes, Este aspecto ha sido usado para obtener los recursos (imágenes y archivos) utilizados por el sistema en su ejecución (OVH, 2015).

Antes de dar comienzo a la definición exhaustiva de los procesos de implementación seguidos, cabe destacar que el sistema en desarrollo es un proyecto Maven, Java Web. Aprovecha todas las capacidades ya definidas de dicha tecnología para enlazar las librerías utilizadas de: Spring framework y twitter4j sin sobrecargar su contenido y resolviendo todas sus dependencias de forma automática.

8.2 CARTOGRAFÍA WEB. DATOS ABIERTOS

Este apartado constituye el eje fundamental que da origen al sistema “2Train – Análisis deportivo de la ciudad de Málaga”.

Desde un principio, el objetivo de esta aplicación web es representar una vista aérea de la ciudad de Málaga que permita a un usuario visualizar las distintas posibilidades deportivas que esta ofrece. Por lo tanto, se hace indispensable el uso de tecnologías asociadas a la cartografía web.

En primer lugar, se ha involucrado a la tecnología cartográfica con más usuarios a nivel mundial que, como es usual, pertenece a la compañía Google: Google Maps. La API ofrecida por esta herramienta proporciona

un acceso simple, a través de código Javascript, a las operaciones base de todo mapa web: creación del mapa, ubicación de marcadores con posibilidad de modificación dinámica de los mismos y edición del centro del mapa.

Sin embargo, el objetivo de este proyecto ha alcanzado un punto más ambicioso que el de la facilidad de creación del mapa requerido. Para poder ofrecer al usuario todas las posibilidades deportivas de la ciudad de Málaga se deben considerar diversas fuentes de datos abiertos con las que procesar las ubicaciones a operar sobre el mapa. Estas fuentes de datos abiertos son suministradas por la página web oficial del CEMI del Ayuntamiento de Málaga. (Bueno Vallejo, 2015)

Para entender el propósito de procesamiento de dichos datos, cabe explicar qué entiende este proyecto por datos abiertos: Conjunto de datos ofrecidos de forma gratuita y pública a través de alguna plataforma web y que se agrupan en archivos de datos de temática común y formato apto para su gestión.

Los formatos de datos abiertos con los que se ha trabajado en este proyecto son:

- **CSV:** Archivo de agrupación de datos que facilita la representación de los mismos en forma de tabla. Las columnas se separan por comas y las filas por saltos de línea.
- **GeoJSON:** Formato de intercambio de datos geoespaciales basado en JSON (Morales, 2015).
- **SHP:** Formato de archivo informático propietario de datos espaciales.

Concretamente, el sistema ofrece la información cartográfica referente a los conjuntos de datos cartográficos de: todos los centros deportivos de la ciudad de Málaga, la definición de todos sus carriles bici, ubicación de los puntos de recogida de bicicletas correspondientes al proyecto “MálagaBici”, la definición de las áreas de Málaga en función de sus

niveles de ruido (decibelios) registrados el pasado año, ubicación de las zonas verdes registradas en la ciudad, definición de las playas aptas para el baño y localización de las pistas generadas para la práctica de vóley playa.

Una vez el sistema hace uso de todas estas fuentes de datos, se considera necesaria la utilización de una herramienta cartográfica que sea capaz de hacer un procesamiento de los mismos de una forma más concreta que las posibilidades que ofrece Google Maps al respecto.

Así se llega a la elección de CartoDB. Esta herramienta cartográfica está centrada en el procesamiento de grandes cantidades de datos en tiempo real para su representación cartográfica tanto en el perfil de usuario que lo requiera, como en cualquier aplicación externa a través de su API: “carto.js”.

Una vez elegida la herramienta, se crea un perfil de usuario de forma gratuita y se conectan las fuentes de datos abiertos. Ellas generan de forma directa un mapa con su representación sobre la ciudad de Málaga en el que cada fuente de datos se identifica con una capa del mapa. Esta generación otorga total libertad para la gestión del mapa en cualquier ámbito, siempre respetando los límites de los datos que se tienen.

En este caso, se han modificado los colores y formas de los datos representados para que se muestren de forma más significativa en la aplicación y se ha modificado el orden de las capas del mismo para que las capas que muestran marcadas áreas mayores del terreno no se superpongan sobre datos más concretos.

El número máximo de capas permitidas para un mapa es de ocho. Dado que el mapa generado aún puede admitir una más, se añade una fuente de datos generada de forma manual en formato “GeoJSON” con la muestra de un único punto. Este dato representa la ubicación actual del usuario sobre el mapa, como su última capa.

Carto presenta un cúmulo de facilidades y potencialidades que, según el estudio realizado en este proyecto, supera a Google Maps en la gestión

de datos abiertos. Sin embargo, presenta mayor complejidad de uso en tareas tan simples como añadir un marcador de forma dinámica sobre el mapa. Por lo tanto, para adaptar esta herramienta a la necesidad de ubicar al usuario en tiempo real sobre el conjunto de datos representados en el mapa, se almacena esta última fuente de datos cuyos valores geoespaciales se modifican a través de la base de datos del perfil del sistema en la herramienta.

A través de la API “carto.js” se crea el mapa generado en el perfil de usuario del sistema en la herramienta, a través de un enlace generado que corrobora las credenciales de autenticación el perfil de usuario en cuestión.

Una vez se obtiene esta referencia al mapa, una colaboración entre esta herramienta, el código abierto de OpenStreet Maps y la compañía Leaflet genera un conjunto de operaciones que permite mostrar en la aplicación el mapa del que se ha obtenido la referencia. Además, los cambios hechos sobre el mapa en su perfil de origen se muestran automáticamente en la aplicación y la visibilidad de las distintas capas pueden modificarse a través de las sencillas operaciones: “show()” y “hide()”.

Por último, a nivel cartográfico básico, cabe definir el tratamiento de la ubicación del usuario. Este es el momento en el que se hace uso de la API de geocodificación ofrecida por Google.

Dependiendo de si se realiza sobre la ubicación inicial del navegador del usuario o sobre una ubicación manual de este, se procede de un modo u otro:

- Ubicación automática: En este caso se invoca a la operación “getCurrentPosition()” que ofrece el apartado de geolocalización del navegador y directamente se obtienen las coordenadas que se esperaban o un mensaje de error si esta función no está disponible. Este mensaje de error avisa al usuario de que debe introducir su ubicación de forma manual para verse sobre el mapa.

- **Ubicación manual:** Aquí el usuario introduce una dirección postal en un campo de texto y esta se procesa internamente para obtener sus coordenadas. El uso de este proceso requiere la llamada a la API de Google con unas credenciales de API_KEY. Por lo tanto, no es seguro proceder con ella a través del cliente (a modo Javascript). Se estaría proponiendo la clave Google del sistema en todos los clientes. Por ello, se hace una llamada Ajax al servidor, este crea una interfaz REST a través de la capacidad “RestTemplate” que ofrece Spring Framework y hace la petición REST pertinente a la API de Google. Se procesa el JSON de respuesta y se comunican las coordenadas obtenidas nuevamente al cliente.

En ambos casos se comunica al controlador los nuevos valores de latitud y longitud del usuario a través de una conexión Ajax.

8.3 BASE DE DATOS

El sistema en desarrollo no presenta necesidad de creación y gestión de una base de datos que le de soporte. Esto es debido a que la información que se maneja se procesa en tiempo de ejecución a través de servicios REST y tecnologías externas sin necesidad de ser almacenadas. Desde la muestra del horario de amanecer y anochecer, hasta la muestra del mapa con los filtrados sobre su conjunto de datos.

Los datos almacenados por el sistema simplemente se corresponden con el código fuente del back-end y un conjunto pequeño de variables de acceso concurrente, el cual gestiona el servicio cloud elegido para su despliegue. Además de un conjunto de variables de operación definidas sobre el código javascript que lleva a cabo parte de la funcionalidad del sistema. Dichas variables no almacenan contenido sensible y datos relevantes a almacenar para la gestión futura del proyecto. Por pertenecer a la parte Javascript son ejecutadas en el lado del cliente.

Sin embargo, es evidente que se realiza un fuerte procesamiento de datos y análisis de flujo de información. Estas fuentes de datos son las almacenadas en la base de datos que caracteriza a los perfiles de usuario de la herramienta Carto.

Se trata de una base de datos PostgreSQL asociada a la cuenta personal que se ha hecho del sistema en dicha herramienta cartográfica. En principio, las fuentes de datos que se muestran en el mapa generado son invariables. Pero su almacenamiento en esta base de datos otorga la libertad de modificarlos a través de sentencias SQL restringidas a los límites de este tipo de base de datos relacional. Por tanto, una vez se tiene el mapa mostrado, este se modifica de forma dinámica a través de las peticiones de filtrado del usuario.

Estos filtrados son gestionados de formas diferentes según su complejidad y su tipo ofrecido:

- **Filtrado por polideportivo:** Es el activado por defecto e indica a la capa de datos de centros deportivos que muestre todas sus ubicaciones y hace visible la capa de zonas verdes.
- **Filtrado por ciclismo:** Hace visible las capas de carriles bici y puntos de recogida de bicicletas “MálagaBici”. No aplica SQL, ya que la información mostrada por dichas capas siempre se mantiene.
- **Filtrado por zonas de musculación:** Se modifica la capacidad de mostrado de datos de los centros deportivos a aquellas que correspondan solo a zonas de musculación al aire libre o gimnasios.
- **Filtrado por natación:** Hace visible las capas de playas y puntos de vóley playa, y además modifica la muestra de centros deportivos, restringiéndola a aquellos que sean piscinas.
- **Filtrado por ruido:** Hace visible la capa de niveles de ruido de la ciudad de Málaga. No necesita aplicar SQL.

- **Filtrado por distancia:** Este es uno de los puntos más conflictivos del filtrado, ya que se requiere un conocimiento específico de las operaciones ofrecidas por el tipo de base de datos “PostgreSQL” y resta portabilidad al código.

Para poder realizarlo ha sido necesario operar sobre la base de datos de dos formas diferentes:

- **Petición de distancia:** Se hace una petición Ajax a la base de datos con la sentencia incrustada como valor GET, en la que se devuelvan los identificadores únicos de todos los centros deportivos y la distancia de estos a la ubicación que se le pasa a modo de coordenadas: (latitud, longitud). Esta distancia se calcula con la operación “ST_Distance”.

Esta operación “ST_Distance” calcula el valor de distancia en metros entre el atributo “the_geom” de cada centro deportivo y otro atributo the_geom que se genera de forma dinámica en la petición a través de la operación “ST_SetSRID” con las coordenadas pasadas. Dichas coordenadas son las que tiene el usuario como ubicación actual a tener en cuenta.

- **Generación de listado de centros deportivos a mostrar:** Una vez se obtienen los valores de identificación y distancia de todos los centros deportivos, estos atraviesan un procesamiento que genera un lista con todos aquellos identificadores cuya distancia sea menor (en kilómetros) al valor de distancia introducido por el usuario.

Por último, se modifica a través de una sentencia SQL como las anteriores los centros deportivos a mostrar, respetando el filtrado presenta y cuyos identificadores estén en la lista especificada.

- **Ubicación de usuario:** Con las coordenadas que se tienen del usuario, se modifica el valor “the_geom” asociado a la capa dedicada a la muestra de esta localización y se actualiza su vista.

8.4 ANÁLISIS DE TENDENCIA

Este apartado compone el segundo objetivo más ambicioso del proyecto. Esta parte de la implementación se corresponde con el requisito funcional: RF-006.

En ella se ofrece al usuario la posibilidad de interactuar con un listado deportivo conformado por: “Correr”, “Ciclismo”, “Natación” y “Otros”. La selección de uno o más deportes por parte del usuario desencadena el proceso de análisis de tendencia a través de Twitter.

En primer lugar, se establece una comunicación Ajax con el controlador en la que se envían los deportes seleccionados por el usuario. El controlador los recibe y modifica este atributo en el POJO Java que se tiene para contener las variables de interacción concurrente que se necesitan. A continuación, modifica una variable que da acceso al procesamiento de tuits y crea una hebra en segundo plano que se identifica con el análisis en Twitter.

Esta hebra es la base del uso de la API de Twitter. Para implementarla se ha hecho uso de la librería “twitter4j”. Esta librería está especializada en la interacción de código Java con las operaciones de Twitter. Por ejemplo, una de las facilidades que decantan su uso es el acceso al flujo de tuits actuales a través de una simple operación “getTweets()” sobre la que se ha podido establecer un objeto Query que filtre dichos tuits a recoger. Además, ofrece dichos tuits como un listado, habiéndolos ya tratado desde el formato JSON de origen que provee la API de Twitter.

Por tanto, la hebra de procesamiento comienza creando una instancia de comunicación entre el sistema y Twitter API. Esto se hace a través de unas credenciales obtenidas con anterioridad, dado que se debe registrar el sistema como aplicación en el sistema de Twitter para obtenerlas.

Una vez autenticado de forma automática, se consulta la variable de acceso que le indica si debe detener su procesamiento y esperar de forma no activa (await()), o puede continuar. En caso de estar analizando, la hebra consulta la ubicación actual del cliente y genera un objeto Query

filtrado con un componente de localización, generado con los valores de latitud y longitud del usuario. Dicho filtrado se hace sobre un radio preestablecido de cien kilómetros. A continuación, se recogen los cien tuits más actuales en dicho radio de acción y según, los deportes seleccionados, se comparan individualmente con un conjunto de palabras clave preestablecidos para cada selección. Dichos conjuntos de palabras claves se implementan a modo de enumerado independiente para no sobrecargar el contenido de la hebra. Para cada coincidencia de palabra entre el conjunto de palabras claves y un tuit se aumenta un valor inicial de tendencia en una unidad. Si un tuit ya tiene una coincidencia no se sigue analizando.

El valor final de tendencia que se comunica al controlador es el sumatorio de todas las coincidencias dadas, dividido por una marca temporal que se reinicia cada vez que se actualiza la selección deportiva y que va aumentando en cada interacción (de modo que una selección con mucho tiempo de análisis es menos significativa que la obtención de un gran número de coincidencias al inicio de ello), y se multiplica por un factor de incremento que es relativo al número máximo de tuits al que puede acceder el sistema. Este proceso se ejecuta cada veinte segundos.

El valor de tendencia a mostrar al usuario se obtiene debido a una tarea periódica que se inicia en el lado del cliente, en javascript, cuando da comienzo el análisis de tendencia. Esta tarea periódica realiza una petición web del valor de tendencia a través de Ajax al controlador. Este le devuelve el valor de tendencia y el número de tuits analizados por minuto. Lo establecido es que se obtenga un valor de trescientos tuits analizados por minuto, pero como dado que el flujo de Twitter puede variar según el momento del día, cabe la posibilidad de que no pueda ofrecer tal cantidad de tuits no repetidos. Este valor ayuda a ofrecer un mensaje de ayuda al usuario en el que si se identifica un análisis sobre una cantidad de tuits del setenta por ciento menor que lo esperado (menos de doscientos diez tuits), se propone la detención del análisis por ser poco significativo.

Por último, el valor de tendencia recibido se expone a través de una barra de progreso fácil de visualizar por el usuario.

Este proceso de análisis de tendencia puede ser detenido al pulsar el botón de “Stop” que se muestra. Este detiene la tarea periódica generada e informa al controlador mediante Ajax del suceso. El controlador modifica la variable de consulta de continuación a la que accede la hebra, de forma que la próxima vez que intente proseguir se verá obligada a realizar una espera no activa. De este modo no consume recursos pero no tiene que eliminarse. Esta detención también se da al deseleccionar todos los deportes que estuviesen seleccionados.

Si el usuario vuelve a pulsar alguna selección deportiva, el sistema vuelve a crear la hebra periódica y una comunicación con el controlador hacer que este modifique el valor de la variable de consulta de la hebra de análisis y la incita a despertarse (notify()).

Para facilitar la identificación de lo significativo de cada tipo de deporte seleccionado y dado que se pueden seleccionar varios simultáneamente, cada vez que se deselecciona un deporte, esto se actualiza a través del controlador en la variable de consulta de la hebra. Esta decremента el valor de tendencia mostrado con respecto a las coincidencias que había aportado el deporte eliminado hasta el momento.

8.5 PROGRAMACIÓN CONCURRENTE

La exposición del punto anterior, referente al análisis de tendencia, hace evidente la necesidad de introducir en el sistema un conjunto de variables que puedan ser accedidas y modificadas tanto por la hebra principal (por el controlador), tanto por una hebra secundaria, la hebra de análisis en este caso.

El hecho de que una variable pueda ser modificada por una hebra y accedida por otros procesos hace vigente un problema de exclusión mutua que debe ser solucionado. Puede darse el caso en el que una hebra consulte una variable que está siendo modificada por otra hebra y, finalmente, la consulta devuelva el valor menos actual y, por tanto, equivocado de la variable. Se debe asegurar que todo acceso a una

variable de este tipo se hace de forma no simultánea entre hebras (exclusión mutua).

En este sistema se ha generado una clase Java contenedora de todas las variables que pueden ser accedidas y modificadas por las distintas hebras. Estas variables se definen de forma estática, de modo que puedan ser accedidas desde cualquier punto del sistema, sin necesidad de generar un objeto concreto de ello. Sin embargo, esto no basta para solucionar el problema.

Para todas estas variables se han definido sus métodos de extracción y modificación (getters y setters) aplicando el concepto Java: “synchronized”.

En principio, se desarrollan “mutex” que restringen el acceso de forma literal a la variable. De este modo, una hebra que quiere leer el valor de una variable, siempre obtendrá el valor adecuado de esta. No puede acceder a dicha variable si está siendo tratada por otra hebra. Esta solución es efectiva pero no explota toda la facilidad que ofrece Java para solventar estos problemas.

Aplicando la etiqueta “synchronized” a los métodos que afectan a la variable que se quiere asegurar como concurrente, se consigue el mismo efecto que con los “mutex” pero sin necesidad de definir nuevas variables especializadas y realizar llamadas de sincronización entre ellas (lock(), unlock()).

La etiqueta “synchronized” aplica instantáneamente el concepto de acceso en exclusión mutua al método que la contiene. Por tanto, esta solución asegura que la comunicación de variables entre el controlador y la hebra de análisis siempre se realice de forma correcta y respetando la concurrencia que se genera.

En este sistema, las variables accedidas bajo condiciones de exclusión mutua son:

- Latitud/Longitud de usuario: Se modifica en la hebra principal y se accede a ella en el análisis para establecer el radio de filtrado de tuits.
- Valor de tendencia: Se genera en la hebra de análisis y se muestra en la hebra principal.
- Número de tuits: Se genera en la hebra de análisis y se muestra en la hebra principal.
- Listado de deportes seleccionados: Se modifica en la hebra principal y se consulta en la hebra de análisis.
- Condición de continuación: La modifica la hebra principal para establecer el carácter de actividad o no de la hebra de análisis.

8.6 FRAMEWORKS

Aunque ya se ha hablado de ellos de forma breve, en este apartado se definen los frameworks aplicados a las tecnologías de base utilizadas en este proyecto.

8.6.1 FRONT-END

A nivel de front-end el lenguaje de base utilizado es Javascript. Dado el desconocimiento inicial que se tenía de esta tecnología, se ha aprendido a programar en este idioma durante el proyecto y gran parte de la funcionalidad definida a nivel de cliente se hace sobre este lenguaje base.

Sin embargo, alguna de las capacidades del lenguaje son claramente mejoradas en uno de los frameworks más usados actualmente sobre esta tecnología: JQuery.

Este framework ayuda al programador a identificar nuevas formas de acceso y tratamiento de las variables tratadas a nivel HTML. Reduce en gran medida el número de caracteres y sentencias necesarias para ejecutar distintas funcionalidades, aplicando una nueva notación basada en el símbolo “\$”. Además, dado su grado de uso y

la facilidad que ofrecen otros frameworks similares como Angular, es muy sencillo encontrar documentación funcional de actividades complejas.

Por este último aspecto, se ha usado el framework JQuery para el acceso a la comunicación Ajax en este sistema.

Uno de los requisitos primordiales de uso del proyecto es mantener una facilidad y velocidad de uso atractivas. Por ello, el uso de una comunicación asíncrona que no haga necesaria la recarga de la página en ninguna ocasión, por muchas que sean las interacciones del usuario, es una decisión importante a tener en cuenta.

Ajax es la herramienta que permite dicha comunicación asíncrona en todas las ocasiones en que se debe ejecutar. La fácil estructura de uso que JQuery proporciona para su implementación ha sido esencial para su elección.

Con un formato en el que se nombra la tecnología (Ajax), se especifica el tipo de petición y la url a la que se hace dicha petición, se consigue el funcionamiento esperado. Además, se recoge de forma asíncrona el resultado deseado a través de una función “success” en caso de que la petición tenga éxito, o “error” si la petición no obtiene un resultado válido. Esto hace que el control de errores sea muy cómodo y el concepto de prueba unitaria respecto a los resultados obtenidos quede en manos de los servicios proveedores.

8.6.2 BACK-END

A nivel de back-end, ya se ha especificado que el lenguaje base elegido para su implementación es Java. Se tienen en cuenta otras elecciones, tales como: PHP o Node.js, siendo estas últimas una de las opciones más relevantes a nivel de servidor en los últimos tiempos de la informática y una tecnología en crecimiento exponencial en este campo, respectivamente.

Sin embargo, en la comparativa de elección de tecnología a utilizar con respecto a las características a nivel de servidor de este sistema se opta por la opción de Spring Framework.

Las tres opciones se presentan como desconocidos al comienzo del proyecto. PHP proporciona una baja curva de aprendizaje baja y tiene una comunidad grande para su documentación. Sin embargo, uno de sus puntos más fuertes es la fácil integración de los servidores con distintos sistemas de bases de datos. El uso de Node.js resulta muy interesante dado que se desarrolla por Google con el objetivo de ejecutar el código javascript del navegador de forma extremadamente rápida, abriendo la posibilidad de ejecutar su motor en el nodo servidor (Vergara, 2016). Su nivel de adaptación al sistema en desarrollo es superior al que presenta PHP, por la carencia de base de datos propia. No obstante, existe una desventaja fundamental frente a la consideración de Spring Framework. El uso procesos concurrentes.

La necesidad de creación de nuevas hebras, la sincronización entre ellas y el establecimiento de acceso en exclusión mutua a ciertas variables hacen que finalmente se defina Spring Framework como la tecnología a utilizar en este proyecto.

El lenguaje Java sigue siendo muy usado en la informática actual pero de forma plana presenta un enorme desuso frente a cualquier otro framework. Sin embargo, Spring es su abstracción más usada a nivel servidor para aplicaciones web. Este alto grado de uso, unido a las características del lenguaje Java y al conocimiento que ya se tenía sobre él, lo hacen la elección idónea para su implementación frente al resto de tecnologías analizadas.

Spring no necesita la incorporación de clases propias del framework a la nomenclatura o definición del lenguaje. El único requisito para proceder a su implementación es la incorporación de su sistema de etiquetado al proyecto en desarrollo.

Dada la simplicidad que se otorga al sistema, solo se tiene un controlador. Este será el controlador principal que filtra el acceso inicial al sistema a través de una petición GET. Este controlador se define como una clase Java común (POJO) con la etiqueta “@Controller” en su cabecera y “RequestMapping” que indica la cadena de acceso requerida en la URL.

Como se centra en el desarrollo de aplicaciones empresariales (JavaEE) y, en este caso Java Web, cada operación definida está precedida de la petición que la caracteriza. En vez de tener un método “GET” y otro “POST” con distintas opciones de uso según el requisito que haga de él el nivel cliente, se han definido tres peticiones GET y dos peticiones POST:

- Init (GET): Inicializa la conexión de Twitter y la clase de variables estáticas al inicio de la aplicación.
- Actualiza tendencia (GET): Método al que accede la petición que hace la tarea periódica del cliente que recoge la tendencia analizada.
- Mostrar mapa (GET): Método que recibe la dirección pasada por el usuario, haya las coordenadas a través de Google Maps API y las devuelve al cliente.
- Mi tendencia (POST): Método que gestiona la selección y deselección de deportes en el análisis de tendencia. Modifica las variables compartidas según los criterios antes analizados.
- Flujo Twitter (POST): Detiene el procesamiento de la hebra secundaria de análisis y prepara las variables compartidas para un posible nuevo inicio.

El uso de este framework facilita también la modificación de objetos y valores a través de formularios. Sin embargo, estos obligan a recargar la página con cada interacción (submit) del usuario. El uso

del framework a través de peticiones asíncronas se adapta mejor al sistema deseado.

Por último, aunque sus ventajas no necesitan ser aplicadas en este caso, Spring Framework presenta los elementos web como “.jsp”, que permiten la introducción de código Java explícito entre etiquetas “%” el cual es ejecutado conforme se carga la página.

8.7 DISEÑO WEB

El diseño web de la aplicación se ha previsto desde el comienzo de su implementación. Dada la facilidad de uso de la tecnología Bootstrap, sus ventajas respecto al diseño multiplataforma de las aplicaciones web y la enorme comunidad de usuarios que se centra en su desarrollo de código abierto, ha sido la herramienta de diseño web ideal para el planteamiento de la interfaz.

Salvo el mapa, del que ya se ha especificado su origen, todo elemento que se añade a la interfaz del sistema se identifica con un componente bootstrap prediseñado. Estos componentes presentan una congregación preestablecida entre diseño CSS e interacción Javascript que los hacen muy fáciles de usar y, sobre todo, adaptables a cualquier tipo de resolución de pantalla.

Esta última característica es la primordial de Bootstrap. La capacidad de diseño “responsive” que presenta cuadra completamente con el objetivo multiplataforma que presenta este proyecto.

Se van a enumerar los elementos elegidos para formar la interfaz del sistema:

- Bootstrap-carousel: Elemento de paso de imágenes automático de la cabecera del sistema.
- Bootstrap-slider: Componente de selección de distancia de filtrado.

- Progress-bar: Barra de progreso que muestra la tendencia analizada.
- Panel-default: Paneles que modularizan de forma dinámica y adaptable su contenido. En este sistema dividen el contenido de la página por bloques temáticos.
- Bootstrap-toggle: Botones de diseño bootstrap que tienen implícita la funcionalidad de un checkbox pero se muestran como botones de dos estados.

Por último, aunque los componentes Bootstrap ya lo incorporan, se completa el diseño de la interfaz con CSS.

Dada la tecnología utilizada, simplemente es necesaria su utilización para dimensionar los componentes y ubicarlos coherentemente sobre la interfaz. Además, la aplicación de la extensión ofrecida por CSS “Media queries”, hace posible la definición de distintas distribuciones y dimensionamiento de pantalla en función de la resolución del dispositivo que ejecuta la aplicación. En este sistema se mantiene el diseño para “media queries” características de ordenadores, tablets y smartphones. Esto acaba completando su carácter multiplataforma.

8.8 CLOUD COMPUTING. DESPLIEGUE

Una vez se tiene el sistema completado, se requiere un canal de difusión web operativo que soporte el acceso al sistema de forma simultánea por diversos usuarios y a través de distintos dispositivos.

El carácter no comercial, por el momento, del sistema “2Train – Análisis deportivo de la ciudad de Málaga” hace prácticamente inapreciables sus requisitos de escalabilidad y aceptación de flujo de tráfico. Sin embargo, necesita un servidor web donde alojarse y que asegure su correcta ejecución.

La opción más favorable para obtener este servicio se presenta a través de “Platform as a Service” (PaaS). Como ya se definió, es el segundo nivel de la tecnología Cloud Computing y permite que el sistema no se tenga que encargar de adquirir, configurar o gestionar un servidor web. Este simplemente es ejecutado en uno de los servidores de aplicaciones web de la compañía elegida.

En este caso, se ha elegido la compañía “Heroku”. El sistema requiere poca capacidad de flujo de tráfico simultáneo y debe ser sostenido sobre la ejecución de la tecnología Spring.

Dados sus requisitos no comerciales, los servidores PaaS ofrecen una cuenta gratuita con grandes restricciones de operación que es válida para el despliegue de este proyecto.

Se han valorado los servidores “Google App Engine” y “Heroku”. El primero de ellos requiere un mayor grado de restricciones respecto al código aceptado y librerías aportadas. Además, supone una sobrecarga mayor de documentación interna al proyecto. Necesita la incorporación de librerías propias del servidor y un documento “appengine-web.xml”.

Este proyecto ha sido subido a la plataforma GitHub para poder acceder a los privilegios de “RawGit” (CDN) en cuanto a la distribución de sus recursos de forma eficiente y ejecutable. Estos recursos se componen de las imágenes dispuestas a nivel local y el conjunto de ficheros auxiliares “.js” y “.css” implementados para la descomposición funcional del código. Este último criterio acentúa una buena práctica de programación que agrupa los centenares de líneas de código en módulos de temática común y menor dimensión. La existencia de dichos recursos ha sido esencial para la elección de Heroku como servidor de despliegue.

Heroku ofrece una opción de despliegue en la que se puede conectar un proyecto Maven propio de GitHub con sus servidores. El sistema en cuestión cumple estas dos características de forma completa. De este modo, a través de comandos “git” y con la simple instalación de un

“plugin” a él, denominado “heroku”, se puede desplegar y actualizar el sistema sobre su servidor sin necesidad de cambios adicionales.

A continuación, en la Figura 16, se expone una imagen con las limitaciones de despliegue que presenta el sistema dado su carácter gratuito:

Dyno Type	Sleeps	Professional Features	Memory (RAM)	CPU Share	Dedicated	Compute
free	yes	no	512MB	1x	no	1x-4x

Figura 16: *Detalles de contratación gratuita de “Heroku”*

Siendo Dyno la denominación del servidor, tras treinta minutos de inactividad este se va a dormir para no consumir recursos que puedan acabar con el carácter gratuito del servidor. No es escalable, presenta 512MB de memoria, ofrece una única CPU no exclusiva para este sistema y su capacidad de cómputo va desde 1,2 hasta 4,8 GHz.

Finalmente, se puede acceder al proyecto a través del siguiente enlace:

<https://protected-reef-73935.herokuapp.com/>

9

Mantenimiento y pruebas

Este apartado se corresponde con la última fase que compone a la metodología de iteración utilizada.

En ella, se definen las pruebas realizadas sobre el proyecto para comprobar que el sistema realiza las funcionalidades esperadas. Dicha evaluación se ha ejecutado de forma incremental sobre los requisitos desarrollados en cada iteración del software. Como resultado se obtienen posibles fallos de desarrollo, características que sin producir un error explícito no provocan el resultado que se esperaba o evaluaciones positivas sobre conceptos que aportan más valor del esperado sobre el software desarrollado.

Una vez especificado este concepto, se pasa a una definición tentativa y recomendable del mantenimiento considerado para el soporte de la aplicación. Esta especificación representa un plan de acción que, de llevarse a cabo, es capaz de alargar la vida del producto, resolver posibles problemas obtenidos en tiempo de ejecución y proponer mejoras a definir.

9.1 CONSIDERACIONES PREVIAS

Atendiendo a las pruebas del software, a continuación se especifican los criterios tenidos en cuenta para ejecutarlas.

El sistema “2Train – Análisis deportivo de la ciudad de Málaga” se corresponde con un software de congregación de distintas funcionalidades y flujos de información para proporcionar un servicio coherente y concreto. Por esta misma definición, se evidencia que los recursos que se encarga de reunir y tratar ya han sido implementados por desarrolladores o compañías externas.

Según las funcionalidades generales descritas para este sistema, los servicios que componen su funcionamiento son:

- Generación y alteración de un mapa gestionado a través de la plataforma Carto.
- Obtención del horario de amanecer y anochecer del día presenta a través del servicio REST de AEMET.
- Tratamiento de una hebra de análisis de Twitter, que opera a través de la librería “twitter4j” y esta de Twitter API.

El análisis de funcionamiento a nivel más atómico de detalle posible sería la realización de pruebas unitarias. Debido al nivel de abstracción al que se trabaja, el contenido sensible de la información a tratar es el proveído por los servicios mencionados, los cuales ya han debido ser probados a ese nivel de detalle en su implementación. Además, dada la comunicación Ajax que se implementa para tratar dichos servicios, ya se tienen en cuenta las posibilidades de retorno de información de forma correcta (función success) o con errores (función error). No se considera oportuna la realización de dichas pruebas.

Por otro lado, se debe tener en cuenta una especificación de pruebas de integración de componentes que aseguren el buen funcionamiento de los servicios tratados en conjunto. Además, el aspecto más importante del

que asegurar su exactitud es la navegación del sistema. Esto se consigue a través del análisis de pruebas de navegación, teniendo en cuenta la simplicidad que se le otorga al usuario en la que no existe más de una interfaz.

Ampliando el rango de pruebas que completan el sistema, es evidente la importancia que se le da a lo largo del desarrollo a la consecución de una interfaz de usuario simple, rápida e intuitiva con la que captar la atención de los usuarios. Por lo que, además de evaluar el funcionamiento a nivel funcional y web, también se pretende hacer uso de alguna técnica de evaluación de interfaz de usuario que proporcione posibles mejoras a tener en cuenta.

Dado el contexto académico del proyecto no se cuenta con el tiempo necesario para realizar por separado la implementación de paquetes de prueba de integración, de navegación y evaluación de interfaces de usuario. Además, elaborar un conjunto de pruebas a nivel de implementación no es viable sobre este sistema. Esto se debe a que la información obtenida por él no es contrastable a nivel de datos, sino que gran parte de ella (mapas, filtrados, etc.) solo producen cambios visuales y mensajes de alerta que se perciben a nivel humano. Resultados poco significativos si no se evalúan a nivel de usuario.

Una vez establecidos estos criterios y limitaciones de realización de pruebas sobre el sistema, se ha decidido realizarlas a través de tests experimentales sobre un conjunto de usuarios. Ello hace que se pueda reunir en un mismo concepto de prueba el correcto funcionamiento de la integración de componentes, la correcta navegación del sistema y el uso de una técnica concreta de evaluación de interfaz de usuario.

9.2 TESTS EXPERIMENTALES

Esta técnica de prueba experimental se ha derivado de las técnicas de evaluación heurística conocidas de interfaces de usuario y aplicadas a nivel de usabilidad de la interfaz, pero centrando su uso en el análisis funcional esperado del sistema.

De forma incremental junto con el desarrollo del sistema se ha desarrollado un test indicativo de los objetivos que se pretenden analizar. Este test se ha completado hasta obtener una versión final que provee la evaluación heurística esperada.

Su contenido viene expuesto desde la Figura 17 hasta la Figura 31:

En el despliegue inicial de la aplicación... *

- Tengo claro lo que se me está mostrando
- No sé qué información muestra el mapa
- No se ha cargado correctamente

Figura 17: Test experimental – Pregunta 1

La opción de filtrado por "Ciclismo"...

- Deja claro la muestra de carriles bici y puntos de recogida de bicis
- La información que muestra no queda clara
- No recarga bien en alguna ocasión

Figura 18: Test experimental – Pregunta 2

...

La opción de filtrado por "Ruido" opera correctamente *

- Sí
- No
- No siempre

Figura 19: *Test experimental – Pregunta 3*

...

Las opción de filtrado por "Centros deportivos y Zonas verdes"...

- Ubica las áreas en el sentido que esperaba
- Muestra la información pero no le encuentro utilidad
- No recarga bien en alguna ocasión

Figura 20: *Test experimental – Pregunta 4*

El filtrado por "Musculación" opera correctamente *

- Sí
- No
- No siempre

Figura 21: *Test experimental – Pregunta 5*

Las opciones de filtrado por "Natación"... *

- Me deja distinguir claramente las piscinas, playas públicas y puntos de voley-playa que esperaba
- No funciona siempre
- No tiene utilidad

Figura 22: *Test experimental – Pregunta 6*

Mi ubicación en el mapa aparece correctamente *

- Sí
- No
- No aparece

Figura 23: *Test experimental – Pregunta 7*

Al añadir mi ubicación manualmente... *

En caso de haber filtrado por distancia a mi posición, si cambio mi ubicación *
manualmente el filtrado se actualiza con ella

- Si
- No
- Deja de funcionar

Figura 24: *Test experimental – Pregunta 8*

En caso de introducir una dirección errónea... *

- La aplicación deja de funcionar
- Salta un mensaje de error y la aplicación deja de funcionar
- Salta una guía de error y puedo continuar con la ejecución

Figura 25: *Test experimental – Pregunta 9*

En caso de haber filtrado por distancia a mi posición, si cambio mi ubicación manualmente el filtrado se actualiza con ella *

- Si
- No
- Deja de funcionar

Figura 26: *Test experimental – Pregunta 10*

El rango de distancias de filtrado respecto a mi posición me parece... *

- Insuficiente
- Suficiente
- Sin utilidad real

Figura 27: *Test experimental – Pregunta 11*

Entiendo lo que el apartado de "Tendencia Deportiva" pretende *

- Sí
- No
- A medias

Figura 28: *Test experimental – Pregunta 12*

La selección deportiva en cuanto a tendencia me parece suficiente *

- Sí
- No
- No tiene utilidad

Figura 29: *Test experimental – Pregunta 13*

El análisis de tendencia opera correctamente (actualiza la tendencia, la selección y se detiene cuando debe) *

- Si
- No
- A veces

Figura 30: *Test experimental – Pregunta 14*

Los enunciados de apartados y botones de ayuda me parecen suficientes para entender el sistema *

- Si
- No
- Si pero difícil de usar

Figura 31: Test experimental – Pregunta 15

Tomando los criterios citados anteriormente de integración, navegación y usabilidad, se han asignado valores de 2, 4 y 6 respectivamente a las preguntas según el criterio de interfaz o funcionalidad que evalúan.

A continuación se presenta la Tabla 15, que asigna a cada pregunta del test el valor heurístico preestablecido para ella y el criterio de prueba seguido:

Tabla 15: Ponderación heurística de las preguntas

PREGUNTA	VALOR	CRITERIO
1	2	Interfaz
2	6	Funcionalidad. Filtrados del mapa es una característica esencial.
3	6	Funcionalidad. Filtrados del mapa es una característica esencial.
4	6	Funcionalidad. Filtrados del mapa es una característica esencial.
5	6	Funcionalidad. Filtrados del mapa es una característica esencial.

6	6	Funcionalidad. Filtrados del mapa es una característica esencial.
7	4	Funcionalidad. La ubicación de usuario es importante pero no central.
8	4	Funcionalidad. La ubicación manual de usuario es importante pero no central.
9	4	Funcionalidad. Control de error en el único dato introducido por el usuario. A nivel unitario lo analiza Google API.
10	4	Funcionalidad avanzada.
11	2	Interfaz. Usabilidad de filtrado por distancia.
12	2	Interfaz.
13	4	Evaluación funcional subjetiva.
14	6	Funcionalidad central. Tendencia.
15	2	Interfaz.

9.2.1 PARTICIPANTES SELECCIONADOS

La selección de participantes para realizar los tests experimentales de un modo significativo se ha hecho obteniendo individuos que practican deportes habitualmente, en diferentes rangos de edad y habitantes de la ciudad de Málaga. Además, puesto que el uso de tecnologías y aplicaciones web presenta una importancia estadística mayor entre el sector “joven” de la población (se estima que un joven

español puede estar conectado a su Smartphone unas cinco horas y media al día, (Summers, 2017)).

Con este perfil, se ha decidido realizar los tests evaluando la edad de los usuarios en los siguientes rangos:

- Joven: 10-25 años.
- Adulto: 25-45 años.
- Sénior: 45 en adelante.

Para decidir qué cantidad de usuarios es la adecuada para realizar este test, se ha tenido en cuenta la propuesta de Nielsen en el ámbito del análisis de usabilidad en la que se establece que cinco es el número adecuado de ellos. Es un criterio ampliamente seguido por la industria, bajo el principal argumento de que esa cantidad de usuarios permite revelar alrededor del ochenta y cinco por ciento de los problemas de usabilidad del sitio o software estudiado. Aunque este test se centra en funcionalidad y aspectos más amplios que la usabilidad, se adapta dicho criterio. (Exss, 2017)

Los usuarios elegidos se muestran en la Tabla 16 expuesta a continuación:

Tabla 16: *Usuarios elegidos para pasar el test experimental*

IDENTIFICADOR	NOMBRE	EDAD	CARACTERÍSTICAS
1	Antonio Vallecillo Moreno	Sénior	Tutor de proyecto y practicante de atletismo.
2	Dolores Burgueño Caballero	Adulta	Cotutora de proyecto y usuaria de gimnasio.

3	Javier Sánchez Gil	Joven	Estudiante de Málaga y usuario de gimnasio.
4	Sara Rojo Guerrero	Joven	Estudiante de Málaga y practicante de ciclismo.
5	Jesús Velasco Siles	Joven	Entrenador personal en Málaga.

9.2.2 RESULTADOS OBTENIDOS

Aplicando la ponderación asignada para cada pregunta como valor heurístico, se otorga una puntuación de entre cero y dos a cada una de las tres opciones de cada pregunta, según justifique el aspecto más negativo o más positivo respectivamente (Pizarro, 2017).

Para mantener el anonimato de las respuestas, se nombran a los usuarios de la forma: "U.ID".

A continuación se muestra la Tabla 17 con los resultados individuales y totales de puntuación obtenidos en la evaluación del test:

Tabla 17: Puntuación de test

PREGUNTA	U.1	U.2	U.3	U.4	U.5	TOTAL
1	2	2	2	2	2	10*2
2	2	2	2	2	2	10*6
3	2	2	2	1	2	9*6
4	2	2	2	2	2	10*6
5	2	2	0	2	2	8*6
6	2	2	2	2	2	10*6
7	2	2	2	2	2	10*4
8	2	2	1	2	2	9*4
9	2	2	2	1	2	9*4
10	2	2	2	2	2	10*4
11	2	2	2	2	2	10*2
12	2	2	1	1	1	7*2
13	2	2	2	2	2	10*4
14	2	2	2	2	2	10*6
15	2	1	2	2	1	8*2
TOTAL	30	29	26	27	28	564

9.2.3 CONCLUSIONES

Una vez obtenidos los resultados de la evaluación, se establecen los criterios de puntuación que provocan una acción inmediata en la resolución del problema propuesto por la pregunta evaluada y se aplican los cambios que se consideran oportunos.

La evaluación de una pregunta provoca un cambio en esa especificación del sistema siempre que se obtenga una puntuación inferior al ochenta por ciento de su valoración heurística máxima. Si el sistema en general obtiene un resultado total heurístico inferior al ochenta por ciento de su máximo, será remodelado respecto a las cuestiones más conflictivas, y evaluado nuevamente.

Bajo los resultados obtenidos, se muestra que el sistema obtiene una puntuación de quinientos sesenta y cuatro puntos heurísticos, mientras que su máximo alcanzable es de seiscientos cuarenta puntos. La razón de estos valores otorga al sistema una puntuación superior al ochenta y ocho por ciento. Generalmente, el sistema puede mantener su estado de interfaz y funcionalidad.

A nivel atómico de preguntas, toda operación tiene un valor heurístico obtenido igual o superior al ochenta por ciento. La única cuestión que no supera esta restricción es la número doce: “Entiendo lo que el apartado de tendencia deportiva pretende”.

Para solventarlo, se añade al sistema un botón de ayuda que al pulsarse muestra un mensaje al usuario que indica: “Calculo del porcentaje de tuits que hablan sobre el tema de los deportes seleccionados”. Además, se aporta a la interfaz un panel de texto que muestra el número de tuits analizados por minutos, el cual aparece y desaparece de forma automática cuando se realiza el análisis de tendencia.

9.3 MANTENIMIENTO DE LA APLICACIÓN

El software que se trata se define como multiplataforma, de servicio rápido e intuitivo. Para mantener estos criterios una vez se ejecuta su fase de producción, es necesario mantener unas pautas predefinidas de acción que aseguren su mantenimiento funcional.

El mantenimiento de la aplicación se ha considerado relevante desde el inicio del proyecto. La documentación exhaustiva dedicada a los requisitos funcionales y no funcionales y de documentación, y al modelado y diseño web de la aplicación son de vital importancia para que desarrolladores ajenos al autor puedan entender el sistema. Esto se acompaña de una implementación de código modular en la que los componentes a nivel de servidor y cliente se agrupan en operaciones de nombre significativo, con operaciones auxiliares que reducen la cantidad de código por operación. Además, dicha modularización se ejecuta a nivel funcional. Es decir, los distintos componentes de implementación solo tratan la funcionalidad descrita en su título.

Con estos valores de desarrollo establecidos, el proceso de mantenimiento se enfoca a ser llevado a cabo por cualquier desarrollador de software con acceso a la documentación oportuna.

Como plan de acción tentativo, se proponen los criterios a seguir para los tres aspectos que pueden darse en el proceso de mantenimiento:

- **Arreglo de problemas existentes:** Si se encuentran errores de diseño o implementación en fase de producción, el encargado de mantenimiento debe remitirse a la documentación referente al apartado en cuestión y solventar el problema de manera que se mantengan intactas las características no funcionales del sistema.
- **Implementación de mejoras:** Es viable la implementación de mejoras funcionales, de rendimiento y/o velocidad de ejecución, siempre que los objetivos funcionales del sistema no se reduzcan. Estos pueden ampliarse si se observa un nicho de funcionalidad

requerida por más de un veinte por ciento de los usuarios de la aplicación.

- **Mantenimiento base:** Cada cuarenta y ocho horas de servicio del sistema se deben revisar los niveles máximos de flujo de tráfico y realizar un test de rendimiento y de estrés a un nivel de un diez por ciento superior al flujo de tráfico máximo. Además, si este nivel de tráfico está de un diez a un uno por ciento por debajo del límite permitido por el servidor, se deben renegociar las condiciones de uso.

10

Conclusiones

Se cierra la exposición del trabajo realizado sobre este proyecto con la definición de los factores de éxito alcanzados, los inconvenientes encontrados durante su desarrollo y las decisiones tomadas para resolverlos, y con una breve especificación de las posibles opciones de futuro que se le atribuyen al sistema.

10.1 OBJETIVOS CUMPLIDOS

El sistema “2Train – Análisis deportivo de la ciudad de Málaga” nacía con el objetivo de cualquier aplicación web real: facilitar la toma de decisiones de sus usuarios. En este caso, centrado en el deporte de la ciudad de Málaga.

Se han definido todos y cada uno de los métodos aplicados para alcanzar un sistema funcional que se establezca en los límites establecidos para él. Si se ha llegado al presente apartado es porque se considera que se han alcanzado los factores de éxito necesarios para lograr su conclusión.

Como ya se han definido los criterios de éxito necesarios, se argumenta su consecución:

- Se finaliza el proyecto en su conjunto en torno a seis días antes del límite de entrega establecido por la Escuela Técnica Superior de Ingeniería Informática.
- Se concluye el proyecto tras una duración de un siete por ciento inferior a la estimada en su planificación temporal.
- Se ofrece información de más de tres tipos de áreas deportivas. Entre ellos: centros deportivos generales, puntos de recogida y carriles bici o áreas establecidas para la práctica de vóley playa.
- Se ofrecen más de tres capacidades de filtrado sobre la información cartográfica: Centros deportivos dedicados a la musculación, centros deportivos con piscinas municipales, establecimiento de las áreas de Málaga con sus niveles de ruido o filtrado por distancia a la ubicación del usuario.
- Se muestra la ubicación del usuario tanto de forma automática, como de forma manual.
- Se analiza la tendencia deportiva en Twitter de los deportes seleccionados por el usuario, en un radio significativo para la ciudad de Málaga.
- Se da la capacidad de detener el proceso para evitar un excesivo consumo de recursos.
- Se usa comunicación asíncrona en las peticiones pertinentes para asegurar una respuesta menor a cinco segundos, bajo ciertos criterios mínimos de conexión a internet.
- Se establece una única interfaz de fácil interacción en la que ningún objetivo supera los tres clicks.

Con el cumplimiento de dichos requisitos se cumplen todos los objetivos marcados en el proyecto, habiendo adoptado un diseño propio, con las tomas de decisiones que han sido pertinentes y aplicando las técnicas

formativas aprendidas a lo largo del grado en cuanto a la gestión y ejecución de proyectos software.

10.2 DIFICULTADES ENCONTRADAS

Alcanzar la aceptación del proyecto no ha sido un camino fácil. En su primera definición, el sistema esperado debía acometer la función de cartografiar de forma funcional las posibilidades deportivas de la ciudad de Málaga, a la vez que aplicaba ciertos criterios de procesamiento de eventos complejos sobre los flujos de información utilizados.

En una colaboración con el Ayuntamiento de Málaga, se alcanzaron los flujos de información necesarios para aplicar el concepto cartográfico del sistema. Sin embargo, el procesamiento de eventos complejos requiere una conexión a flujos de datos activos. Es decir, estos deben estar introduciendo información en tiempo real en intervalos cortos de tiempo. Los datos a los que se han podido acceder otorgan la funcionalidad cartográfica esperada pero no presentan la modificación continua que debe tener su uso en el procesamiento de eventos complejos.

El conocimiento de este factor hizo que al comienzo del proyecto, tras la consecución de una primera iteración tentativa, se debiese redefinir un factor muy importante del sistema. En la evaluación de la situación, no se consideró viable cambiar el concepto central del análisis deportivo de la ciudad de Málaga, dadas las curiosidades de uso de nuevas herramientas y viabilidad como producto final que presentaba. Por lo tanto, se debía sacrificar el uso de procesamiento de eventos complejos. La primera opción tomada fue hacer un procesamiento manual de los datos que se usaban para demarcar diferentes aspectos sobre el mapa.

No obstante, la formación previa a la ejecución del proyecto en aspectos de procesamiento de datos se veía desaprovechada en esta modificación. Así que, finalmente, se decidió hacer uso de un procesamiento de datos interactivo en tiempo real que no hacía modificar la información cartográfica considerada. El análisis de tendencia de Twitter provee la

complejidad de procesamiento de datos interactivos a la que se quería llegar desde un principio, manteniendo los objetivos primarios que dieron origen al proyecto. Además, este cambio ha provocado que el concepto de procesamiento de eventos complejos se elimine del título del proyecto y se adquiriera uno más comercial: “2Train – Análisis deportivo de la ciudad de Málaga”, en el que “2Train” intenta realizar un juego de palabras en inglés con el pronombre español “Tú” derivados del número dos en inglés “two” y la palabra “train”, de traducción: entrenar.

Otro problema que se quiere mencionar es la consideración del cambio de proyecto de implementación en tiempo de ejecución. Aplicando, los conocimientos adquiridos a lo largo del grado, se comenzó la implementación desarrollando un proyecto Spring – Java Web. Sin embargo, el uso de los diversos servicios y tecnologías comenzaron a sobrecargar demasiado el contenido final del proyecto. Para un concepto de sistema funcional, rápido y de fácil acceso no se pretendía aplicarle una carga desmedida en el uso de librerías y factores externos. Por ello, se tomó la decisión de convertir el proyecto Java Web en Maven. Esta tecnología deshacía la sobrecarga de librerías que se quería evitar, aportando su potencialidad de establecer conexiones a los recursos necesarios a través simples enlaces a su repositorio y con la resolución de dependencias de forma automática. Además, ello ha facilitado mucho el despliegue del proyecto en GitHub y en su servidor de ejecución (Heroku) a través de su potencial en línea de comandos (mvn).

Por último, el problema tal vez de menor envergadura pero que más difícil ha sido de resolver tiene que ver con el uso de la API de Carto “carto.js”. Como se ha especificado, esta tecnología cartográfica crece exponencialmente pero aún no tiene unos niveles de usabilidad tan amplios como por ejemplo, Google Maps. Ha sido necesaria la captación de información de diversas fuentes online para llegar a ejecutar las funcionalidades que se quería y, además, para lograr identificar la correlación entre la API y el manejo de su base de datos (PostgreSQL). A pesar de ello, la comunidad que rodea a la herramienta ha facilitado el éxito de esta labor.

Sin embargo, ha sido el carácter económico el que ha supuesto una dificultad mayor en su uso. Carto es una herramienta pensada para el uso de empresas. A nivel estudiantil, las posibilidades presentadas se alcanzan a través de su perfil gratuito, el cual es muy limitado y sin establecimiento de clave de usuario (API_KEY) con la que operar sobre modificaciones en base de datos. Con solo permiso para consultar información, no se podía realizar el tratamiento de la ubicación de usuario que pretendía la aplicación. Sin una cuenta profesional con API_KEY, hacer operaciones como editar (update) o insertar (insert) no estaba permitido.

En este punto se planteó el cambio de tecnología de Carto a Google Maps. Sin embargo, las múltiples ventajas que aporta en el procesamiento de datos abiertos, la originalidad de su uso y el grado de finalización que ya tenía el proyecto no lo hacía viable. La única opción que cabía era contactar con el servicio técnico de “Carto”, explicar la situación en la que se encontraba el perfil de usuario de este sistema y el nivel formativo, y no lucrativo, del proyecto en cuestión.

Mientras se esperaba respuesta de la compañía, se aplazó la ejecución de esta funcionalidad pero, finalmente, tuvieron un comportamiento ejemplar al responder. Al sistema, por su carácter de Trabajo de Fin de Grado, se le amplió un perfil profesional de forma gratuita con la API_KEY necesaria para cumplimentar todos los objetivos previstos.

El proyecto ha sido la cumplimentación de un sistema que abarque el uso de un número significativo de tecnologías desconocidas para el autor. Pero, sobre todo, se ha tratado de una congregación de todos los conocimientos adquiridos a lo largo de los últimos cuatro años en todas las áreas software tratadas.

Además, ha sido el conocimiento que se tiene acerca de todos estos conceptos, el que ha permitido actuar de la forma más profesional posible sobre un proyecto complejo real. Como se expone en este apartado, la ejecución de un sistema partiendo desde cero no siempre se lleva a cabo de la forma teórica esperada. La toma de decisiones necesaria para

alcanzar los niveles de éxito y aceptación requeridos tiene sus pilares en los conocimientos estudiados, pero esta se ejecuta de forma práctica. Es este proyecto el que consigue establecer un ámbito real a la realización de un sistema con el que concluir finalmente la formación esperada.

10.3 LÍNEAS FUTURAS

Por último, tras el análisis realizado del camino llevado a cabo para gestionar y ejecutar el proyecto, se establecen las líneas futuras de aplicación que se ven potencialmente factibles para él.

- Definición de la misma funcionalidad para más de una ciudad sobre el mismo sistema.
- Uso de mayor cantidad de información acerca de cada sitio ubicado sobre el mapa, si se llega a un acuerdo en el que los flujos de datos se actualicen de forma más frecuente.
- Ampliación de los conceptos deportivos de los que analizar la tendencia en Twitter.

En definitiva, se proponen tres aspectos que pueden hacer que la aplicación expuesta mantenga un crecimiento progresivo, ya sin las restricciones temporales y de ejecución individual que conlleva en su principio.

10.4 VÍDEO-PITCH

El proyecto se genera con el objetivo académico del Trabajo de Fin de Grado que constituye, pero también con la esperanza de que pueda generar un producto de uso real.

El número de horas dedicadas a su generación y las múltiples decisiones tomadas para mejorar su funcionalidad y concepto general hacen que “2Train – Análisis deportivo de Málaga” se vea como una aplicación web

real que puede generar interés tanto a al Ayuntamiento de Málaga, como a la Universidad de Málaga.

Por ello, en una colaboración con esta última, se ha realizado una vídeo-presentación del sistema generado, en la que se expone de forma breve todo su potencial, las ventajas y capacidades que aporta a sus usuarios, de forma concisa y enfocada a la captación de atención de posibles interesados.

Se dispone acceso a dicha presentación en el siguiente enlace:

<http://atenea.lcc.uma.es/Descargas/TFGs/2Train.mp4>





Referencias

- Alegsa, L. (5 de Diciembre de 2010). *Definición de manual de usuario*. Recuperado el 2018, de http://www.alegsa.com.ar/Dic/manual_de_usuario.php.
- Bueno Vallejo, D. (2015). *CEMI*. Recuperado el 2018, de <http://cemi.malaga.eu/es/el-cemi/presentacion/#.Wmd3LK7iZdg>
- Doom, J. (26 de Agosto de 2016). *¿Cuánto espera un usuario a que se cargue una página web?* Recuperado el 2018, de <http://digitaltransformation.ituser.es/noticias/2016/08/cuanto-tiempo-espera-un-usuario-a-que-se-cargue-una-web>
- Durán Toro, A. (2005). *Documentación de requisitos mediante casos de uso*. Recuperado el 2018, de <http://www.1si.us.es/docencia/get.php?id=2008>
- Estado, J. d. (2000). BOE.
- Exss, C. (2017). *¿Cuántos usuarios necesito para un test de usabilidad?* Recuperado el 2018, de <https://www.ida.cl/blog/experiencia-de-usuario/usuarios-necesarios-test-usabilidad/>
- García, H. (7 de Diciembre de 2014). *Identificación de los stakeholders*. Recuperado el 2018, de <https://www.rekursosenprojectmanagement.com/identificacion-de-los-stakeholders/>
- Garzas, J. (2014). *Simple y rápido. Entiende qué es Maven en menos de diez minutos*. Recuperado el 2018, de <http://www.javiergarzas.com/2014/06/maven-en-10-min.html>
- Halad, G., Ridao, M., & Kaplan, G. (2004). *Facilitando la asignación de prioridades a los requisitos*. Recuperado el 2018, de http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER09/ridao.pdf.
- Iglesias Fraga, A. (2016). *¿Qué es el desarrollo ágil y cómo está transformando la industria del software?* Recuperado el 2018, de <http://www.ticbeat.com/tecnologias/que-es-el-desarrollo-agil-y-como-esta-transformando-la-industria-del-software/>

- Luján Mora, S. (2014). *Aprende los fundamentos de la Ingeniería Web*. Recuperado el 2018, de <http://gpsli.dlsi.ua.es/asignaturas/aplicaciones-web/>
- Luque Baena, R. (Noviembre de 2017). *Arquitecturas REST*. Recuperado el 2018, de https://informatica.cv.uma.es/pluginfile.php/274314/mod_resource/content/0/4.%20Arquitecturas%20REST.pdf
- Luque Baena, R. (2009). *Javascript*. Recuperado el 2018, de https://informatica.cv.uma.es/pluginfile.php/261749/mod_resource/content/0/Javascript2.pptx.pdf
- Managers, H. P. (10 de Julio de 2014). *¿Qué es plan de gestión de alcance?* Recuperado el 2018, de <http://www.uv-mdap.com/blog/plan-de-gestion-del-alcance/>
- Martín Alonso, J., Suárez, G., Pérez, I., Pérez, V., Galván, P., & Meléndrez, I. (2015). *Métodologías ágiles de desarrollo software*. Recuperado el 2018, de https://es.wikiversity.org/wiki/Metolog%C3%ADas_%C3%A1giles_de_desarrollo_software
- Maximilian University Munich, L. (19 de Enero de 2010). *UWE - UML-based web engineering*. Recuperado el 2018, de <http://uwe.pst.ifi.lmu.de/toolMagicUWEReferenceV1.3.html>
- Morales, A. (2015). *GeoCSV, GeoPackage y GeoJSON*. Recuperado el 2018, de <https://mappinggis.com/2015/09/geocsv-geopackage-y-geojson-las-nuevas-alternativas-al-shapefile/>
- Oliveira, W. (24 de Julio de 2017). *¿Qué es la gestión de riesgos? Propósitos y concepto*. Recuperado el 2018, de <https://www.heflo.com/es/blog/gestion-de-riesgos/que-es-gestion-de-riesgos/>
- OVH. (2015). *CDN OVH: El complemento indispensable para sus sitios web*. Recuperado el 2018, de <https://www.ovh.es/cdn/ventajas.xml>
- Pérez, M. (2015). *Participación ciudadana en ciencia, tecnología e innovación*. Recuperado el 2018, de <https://prezi.com/v0dzuih1b2ns/participacion-ciudadana-en-ciencia-tecnologia-e-innovacion/>
- Pizarro, N. (2017). *¿En qué consiste una evaluación heurística?* Recuperado el 2018, de <https://www.ida.cl/blog/experiencia-de-usuario/que-es-evaluacion-heuristica/>
- Rodríguez Domínguez, P. (2018). *La importancia de la claridad y sencillez de una interfaz de usuario*. Recuperado el 2018, de <https://sg.com.mx/revista/31/la-importancia-la-claridad-y-sencillez-una-interfaz-usuario>

- Summers, F. (7 de Septiembre de 2017). *Crece la adicción al móvil entre los jóvenes*. Recuperado el 2018, de <http://www.rastreator.com/sala-de-prensa/notas-de-prensa/2017-08-crece-adiccion-movil-entre-jovenes.aspx>
- Vergara, A. (2 de Febrero de 2016). *5 ventajas de usar Node.js*. Recuperado el 2018, de <https://www.facilcloud.com/noticias/5-ventajas-de-usar-node-js/>



Apéndice A

Planificación temporal

La planificación temporal llevada a cabo para la realización de este proyecto tiene como base dos conceptos técnicos fundamentales:

El establecimiento de una Estructura de Desglose de Trabajo (EDT) en la que se especifica el conjunto de actividades atómicas que constituyen el proyecto y el establecimiento de un calendario temporal con las estimaciones necesarias para planificar su ejecución.

La EDT realizada para el proyecto se hace de forma jerárquica, de forma que el nivel superior constituye las funcionalidades entregables en cada una de las seis iteraciones planificadas y la memoria de proyecto. Estos entregables se desglosan en paquetes de trabajo más concretos hasta alcanzar las actividades atómicas, las cuales son las que reciben los identificadores que se usan para su posterior planificación. Todas ellas se corresponden con un seguimiento coherente de la metodología iterativa incremental que se usa.

Se muestra la EDT seccionada en cada uno de los siete entregables definidos en las Figuras 32-38:

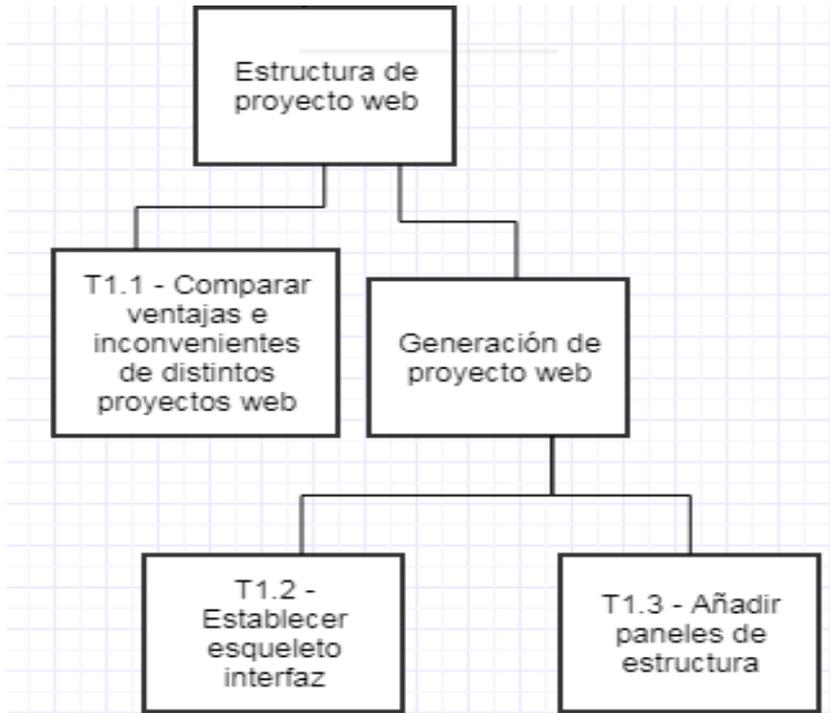


Figura 32: Descomposición de entregable "Estructura de proyecto web"



Figura 33: Descomposición de entregable "Sistema con mapa mostrado"

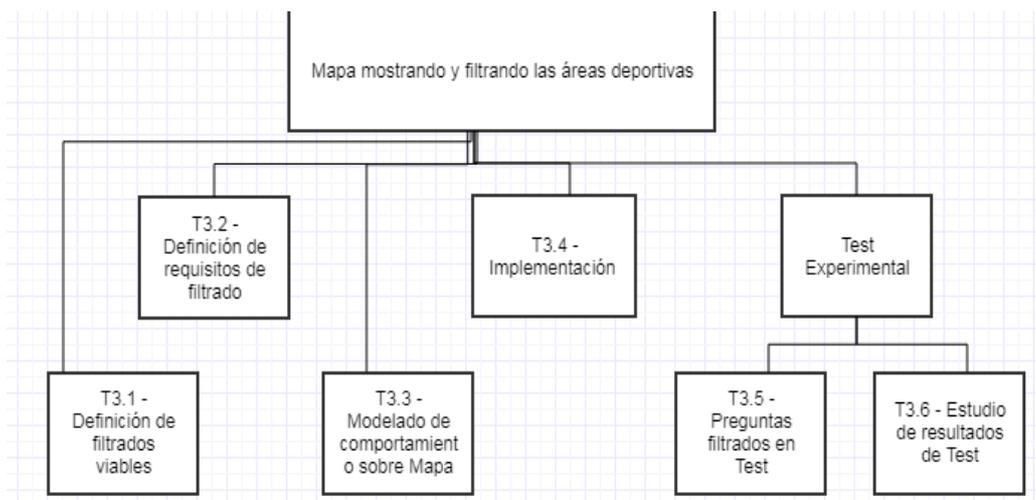


Figura 34: Descomposición de entregable “Mapa mostrando y filtrando las áreas deportivas”

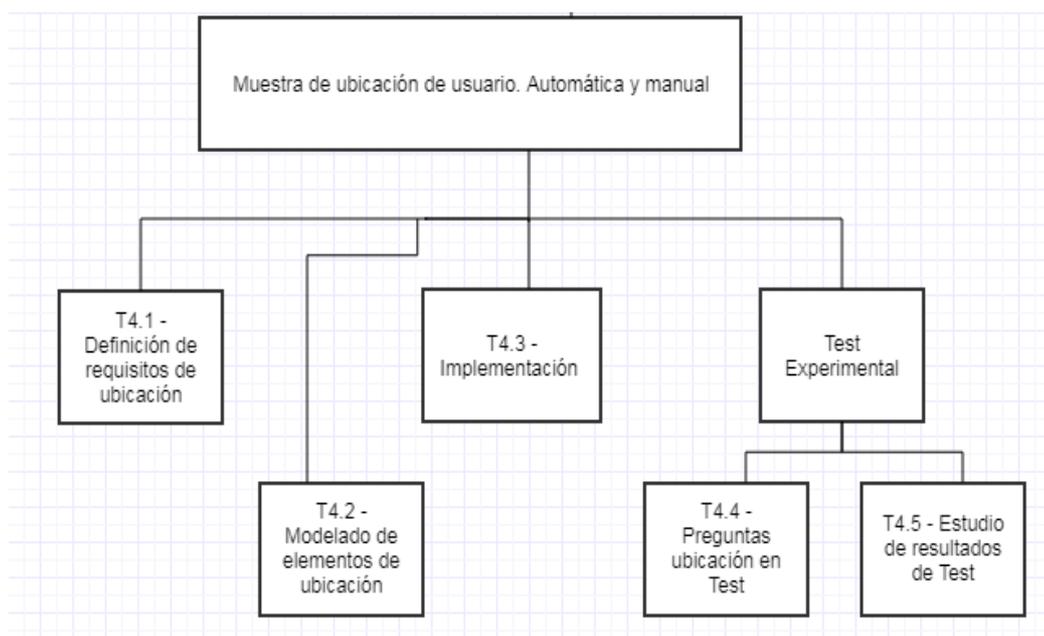


Figura 35: Descomposición de entregable “Muestra de ubicación de usuario. Automática y manual”

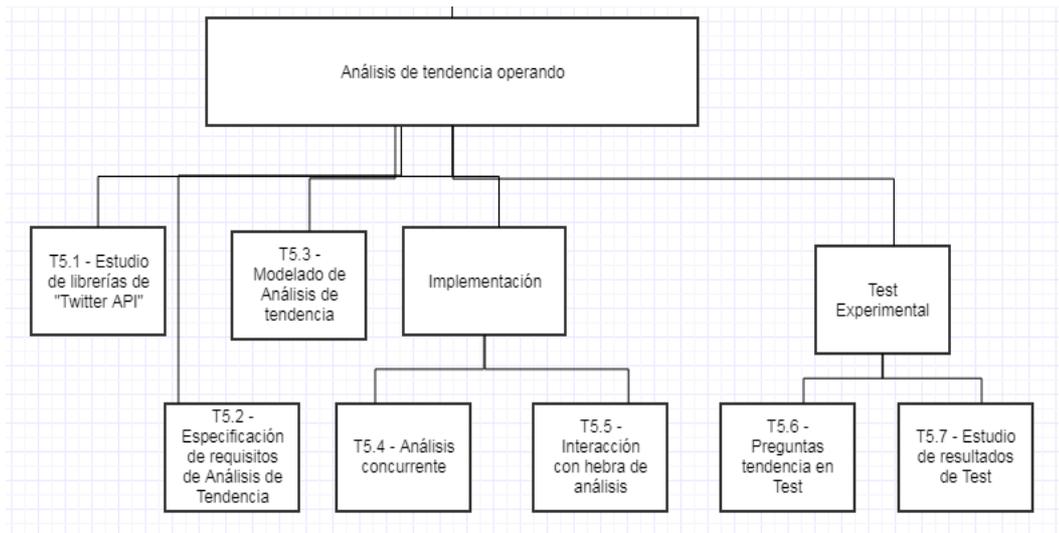


Figura 36: *Descomposición de entregable "Análisis de tendencia operando"*

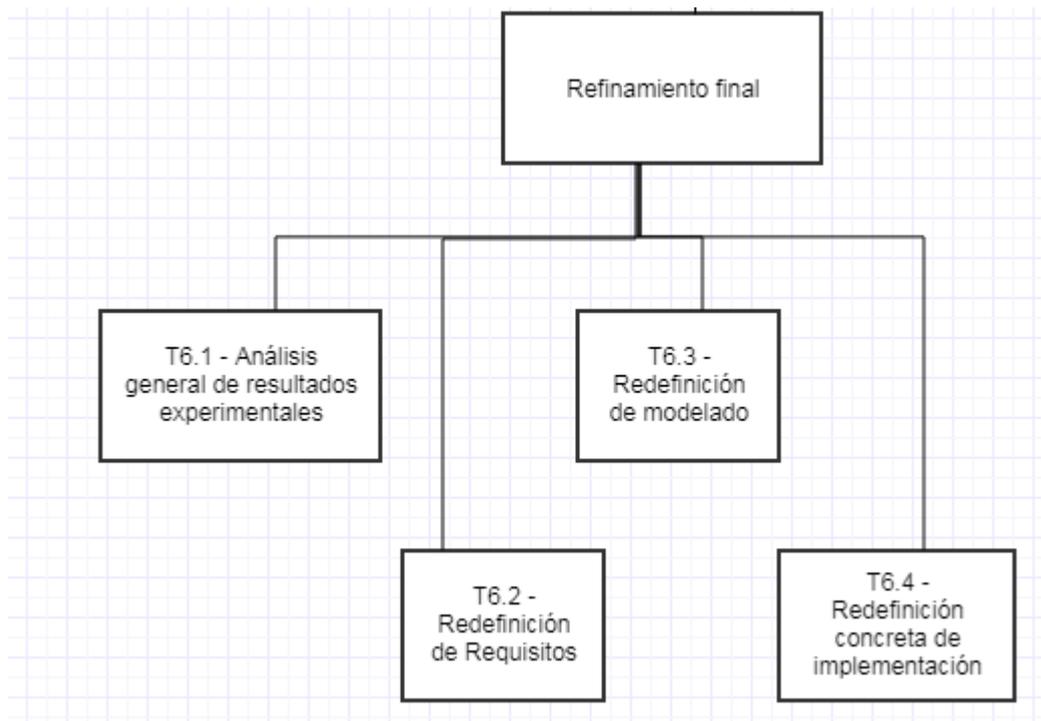


Figura 37: *Descomposición de entregable "Refinamiento final"*

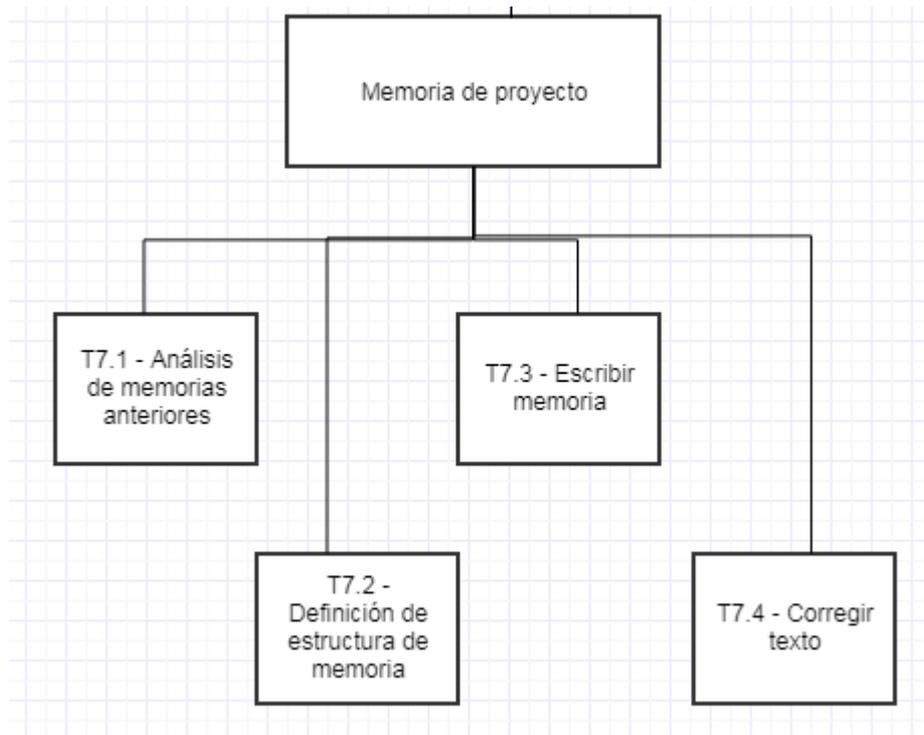


Figura 38: Descomposición de entregable “Memoria de proyecto”

Atendiendo a las actividades atómicas a ejecutar en cada interacción, se expone la planificación temporal explícita con ayuda de la herramienta “MS Project”. Para los aspectos generales de la estimación temporal se considera necesario establecer cuatro días para tareas de especificación de requisitos y modelado de la funcionalidad, seis días para la fase de implementación y dos días para redefinir el test experimental y analizar sus resultados. Esta definición temporal tiene como base la experiencia personal tras la realización de tareas similares en asignaturas anteriores.

La estimación temporal respecto a la consecución de la memoria de proyecto se basa en el objetivo marcado de obtener un documento de entre diez y doce capítulos, otorgando un día para su redacción y un período posterior para su corrección.

Finalmente, el calendario temporal obtenido se muestra en la Tabla 18 mostrada a continuación:

Tabla 18: *Calendario temporal*

Identificador	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	TFG - "2Train - Análisis deportivo de la ciudad de Málaga"	90 días	lun 09/10/17	vie 09/02/18	
2	Estructura de proyecto web	6 días	mar 10/10/17	mar 17/10/17	
3	T1.1 - Comparar ventajas e inconvenientes de distintos proyectos web	4 días	mar 10/10/17	vie 13/10/17	
4	Generación de proyecto web	3 días	vie 13/10/17	mar 17/10/17	
5	T1.2 - Establecer esqueleto interfaz	1 día	lun 16/10/17	lun 16/10/17	3
6	T1.3 - Añadir paneles de estructura	2 días	lun 16/10/17	mar 17/10/17	3
7	Sistema con mapa mostrado	11 días	jue 19/10/17	jue 02/11/17	2
8	T2.1 - Definición de requisitos iniciales	2 días	vie 20/10/17	lun 23/10/17	
9	T2.2 - Modelado del concepto de "Mapa"	2 días	mar 24/10/17	mié 25/10/17	8
10	T2.3 - Implementación de mapa	6 días	jue 26/10/17	jue 02/11/17	9
11	T2.4 - Creación de test experimental	1 día	vie 03/11/17	vie 03/11/17	10
12	Mapa mostrando y filtrando las áreas deportivas	14 días	lun 06/11/17	jue 23/11/17	7

13	T3.1 - Definición de filtrados viables	2 días	lun 06/11/17	mar 07/11/17	
14	T3.2 - Definición de requisitos de filtrado	2 días	mié 08/11/17	jue 09/11/17	13
15	T3.3 - Modelado de comportamiento sobre Mapa	2 días	vie 10/11/17	lun 13/11/17	14
16	T3.4 - Implementación	6 días	mar 14/11/17	mar 21/11/17	15
17	Test Experimental	2 días	mié 22/11/17	jue 23/11/17	16
18	T3.5 - Preguntas filtrados en Test	1 día	mié 22/11/17	mié 22/11/17	
19	T3.6 - Estudio de resultados de Test	1 día	jue 23/11/17	jue 23/11/17	18
20	Muestra de ubicación de usuario. Automática y manual	12 días	vie 24/11/17	lun 11/12/17	7
21	T4.1 - Definición de requisitos de ubicación	2 días	vie 24/11/17	lun 27/11/17	
22	T4.2 - Modelado de elementos de ubicación	2 días	mar 28/11/17	mié 29/11/17	21
23	T4.3 - Implementación	6 días	jue 30/11/17	jue 07/12/17	22
24	Test Experimental	2 días	vie 08/12/17	lun 11/12/17	23
25	T4.4 - Preguntas ubicación en Test	1 día	sáb 09/12/17	sáb 09/12/17	
26	T4.5 - Estudio de resultados de Test	1 día	lun 11/12/17	lun 11/12/17	25

27	Análisis de tendencia operando	14 días	mar 12/12/17	vie 29/12/17	2
28	T5.1 - Estudio de librerías de "Twitter API"	2 días	mar 12/12/17	mié 13/12/17	
29	T5.2 - Especificación de requisitos de Análisis de Tendencia	2 días	jue 14/12/17	vie 15/12/17	28
30	T5.3 - Modelado de Análisis de tendencia	2 días	lun 18/12/17	mar 19/12/17	29
31	Implementación	8 días	lun 18/12/17	mié 27/12/17	30
32	T5.4 - Análisis concurrente	4 días	lun 18/12/17	jue 21/12/17	
33	T5.5 - Interacción con hebra de análisis	4 días	vie 22/12/17	mié 27/12/17	
34	Test Experimental	2 días	jue 28/12/17	vie 29/12/17	31
35	T5.6 - Preguntas tendencia en Test	1 día	jue 28/12/17	jue 28/12/17	
36	T5.7 - Estudio de resultados de Test	1 día	vie 29/12/17	vie 29/12/17	35
37	Refinamiento final	10 días	lun 08/01/18	vie 19/01/18	2;7;12;20;27
38	T6.1 - Análisis general de resultados experimentales	2 días	lun 08/01/18	mar 09/01/18	
39	T6.2 - Redefinición de Requisitos	2 días	mié 10/01/18	jue 11/01/18	38
40	T6.3 - Redefinición de modelado	2 días	vie 12/01/18	lun 15/01/18	39
41	T6.4 - Redefinición de implementación	3 días	mar 16/01/18	jue 18/01/18	40

42	Memoria de proyecto	16 días	sáb 20/01/18	vie 09/02/18	
43	T7.1 - Análisis de memorias anteriores	2 días	sáb 20/01/18	lun 22/01/18	
44	T7.2 - Definición de estructura de memoria	1 día	mar 23/01/18	mar 23/01/18	43
45	T7.3 - Escribir memoria	11 días	mié 24/01/18	mié 07/02/18	44
46	T7.4 - Corregir texto	4 días	mié 07/02/18	lun 12/02/18	45

Las dependencias definidas entre actividades siguen un patrón común para todo entregable. Siguiendo la metodología de trabajo aplicada, la fase de requisitos siempre precede a la de modelado, la de modelado a la de implementación y esta última a la de pruebas (test experimental). En caso de ser necesario un estudio previo a la funcionalidad aplicada, este precede a la fase de requisitos.

Como consideraciones finales de la planificación temporal establecida, se expone que a nivel de entregables las dependencias no están predefinidas y que el nivel de jerarquía atómica o de paquete de trabajo lo define la letra en negrita. Se establece que la escritura de la memoria no depende de entregables anteriores y que todos los que tienen relación con la cartografía dependen del entregable de identificador número siete. Por último, la consecución del entregable de "Refinamiento final" depende de todos los anteriores. Esta iteración se establece como colchón de contingencia en la parte final de desarrollo del sistema aplicando la ley de Parkinson. En ella se especifica que el trabajo se expande hasta rellenar el tiempo disponible para su ejecución.



Apéndice B

Manual de usuario



“2Train – Análisis deportivo de la ciudad de Málaga” es una aplicación web que te ayuda a conocer todos los centros y áreas deportivas de Málaga, dándote la oportunidad de filtrarlos según más te convenga. Además, te permite averiguar la tendencia actual de tu entorno a realizar cualquiera de los deportes que selecciones.

¿Cómo puedes alcanzar estos objetivos?

A través de una única interfaz que se caracteriza por su simpleza y rapidez de uso, a la que puedes acceder desde cualquier dispositivo con acceso a internet. No se requiere navegación compleja para conseguir lo que esperas.

- **Inicio por defecto:**

Sin tener que operar de ninguna forma, la aplicación te muestra desde el inicio un mapa aéreo de la ciudad de Málaga con las ubicaciones de todos sus centros deportivos. Además, también puedes identificar tu propia ubicación a través del punto de color morado que se distingue en mayor tamaño sobre el resto. La etiqueta “YO” puede darte alguna pista.

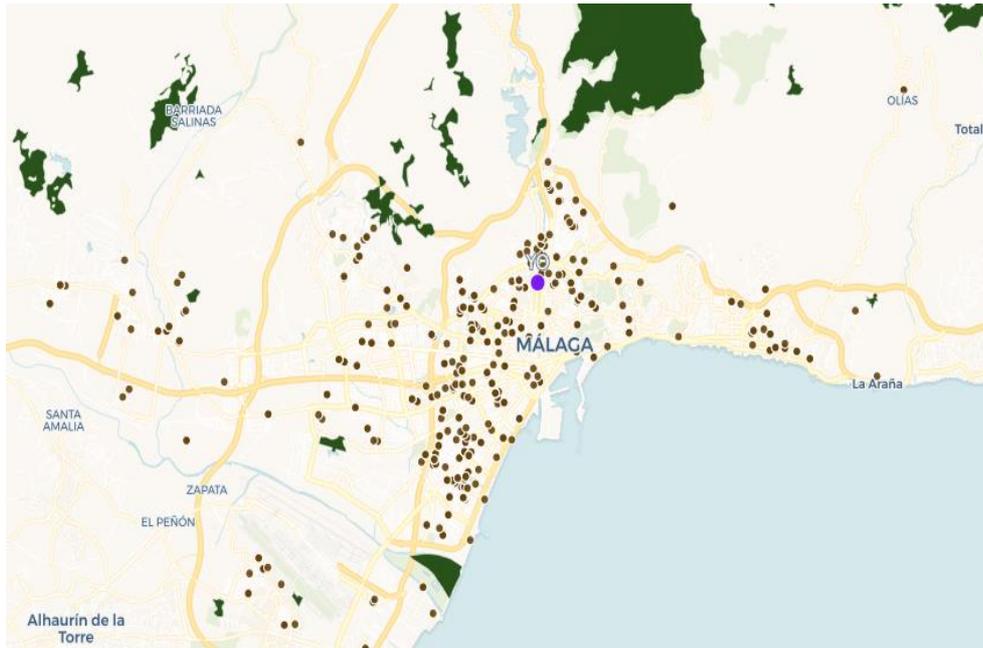


Figura 39: Mapa por defecto "2Train"

También puedes consultar el horario de amanecer y anoecer de la ciudad sin necesidad de buscar ningún campo complejo. Esto puede afectar a tu decisión deportiva. El panel "Horario" te muestra esta información debidamente etiquetada.

Horario	
Amanecer	Atardecer
7:15:34 AM	7:15:43 PM

Figura 40: Panel horario "2Train"

- **¿Qué puedo hacer sobre el sistema?**

Además de mostrar el estado por defecto, la interfaz está esperando tu interacción para ofrecerte distintos niveles de información en dos campos a tratar:

- **Filtrados personalizados sobre el mapa:**

La parte principal inferior del mapa te muestra una botonera con cinco opciones de filtrado y una barra deslizante que aporta algo más de funcionalidad.

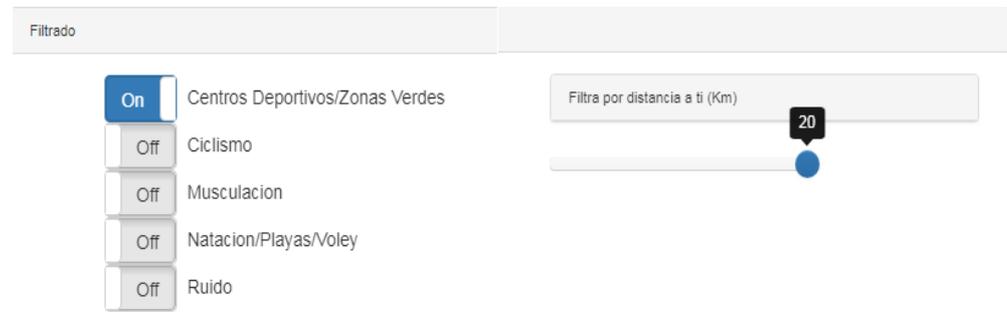


Figura 41: *Filtrados “2Train”*

Cada uno de estos botones muestra su estado de activo o no activo. Puedes cambiar este estado con su pulsación. De estar activo, la visión de su información asociada se superpone en el mapa sin destruir la que podías visualizar. Cada uno de ellos muestra respectivamente:

- Centros deportivos de carácter general y zonas verdes de la ciudad. Mostrado por defecto.
- Carril bici y puntos de recogida de bicicletas (MálagaBici).
- Zonas de musculación. Tanto gimnasios como áreas al aire libre.
- Piscinas municipales, playas aptas para el baño y áreas preparadas para la práctica del vóley playa.
- Niveles de ruido medio mostrados en las distintas zonas de Málaga.

La opción de “Filtra por distancia a ti (Km)”, te permite seleccionar el rango desde cero a los kilómetros (con un máximo de veinte) que quieras en el que quieres que se muestren los centros deportivos de la ciudad.

- **Análisis de tendencia deportiva:**

Para conocer la tendencia deportiva a tu alrededor en ese momento puedes seleccionar uno o varios de los deportes que se muestran en esta área y te interesen:

Correr
Ciclismo
Nadar
Otros

Figura 42: Selección deportiva de análisis de tendencia “2Train”

Ello consulta un flujo de tuits en un radio de cien kilómetros a tu alrededor y muestra el porcentaje de tuits analizados que hablan sobre temas relacionados a los deportes que te interesan.

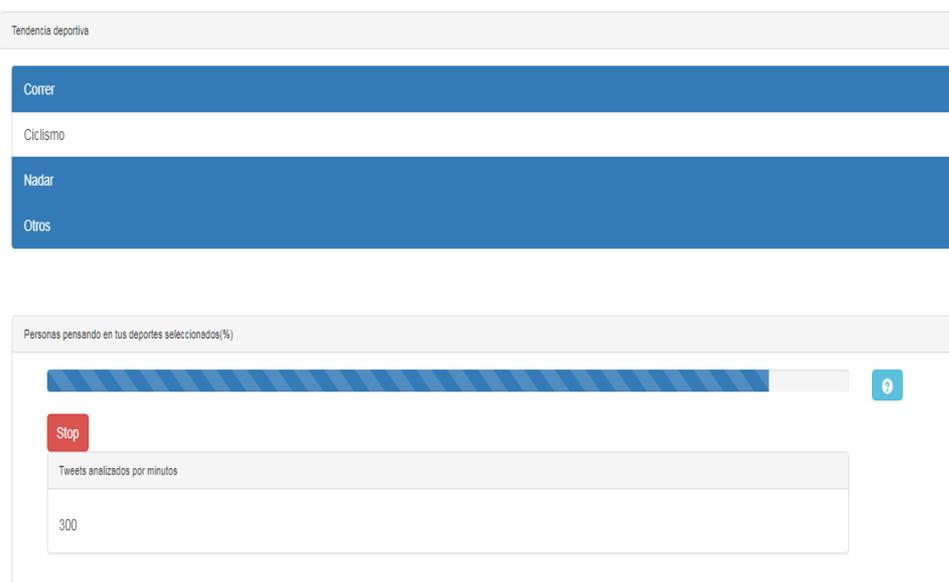


Figura 43: Análisis de tendencia operativo “2Train”

Puedes modificar la selección deportiva que has iniciado cuantas veces quieras, con la selección o deselección de unos u otros deportes. Además, puedes detener el análisis de tendencia de forma automática con la pulsación del botón rojo “Stop” o la deselección de todos los deportes activos.

Cada vez que deseleccionas un deporte, la cantidad de tendencia asociada a él se decrementa en la barra de progreso. Además, si no tienes claro cuánta información se está analizando, justo debajo del botón de “Stop” se te muestra el número de tuits que tu aplicación está analizando por minuto. Si este valor es un setenta por ciento inferior al esperado (300), se te recomendará detener el proceso e intentarlo más tarde.

- **¿Y si mi ubicación no aparece o quiero modificarla?**

Esta opción está considerada de la forma más intuitiva posible. Si ves el campo “Modifica tu ubicación manualmente”, puedes activar su botón “Visualizar” y se mostrará un pequeño formulario.

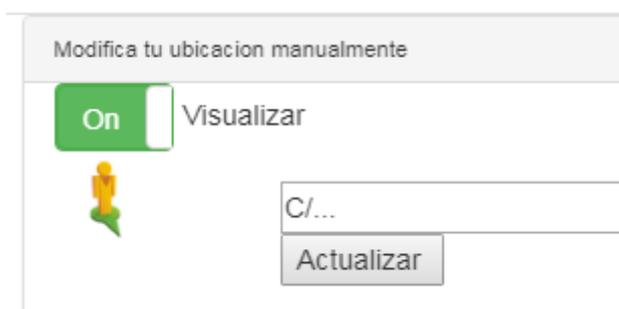


Figura 44: *Panel de modificación de ubicación “2Train”*

En el campo de texto que aparece puedes introducir la dirección en la que quieres ubicarte sobre el mapa. Esta será la nueva localización sobre la que el sistema filtrará los centros deportivos y analizará la distancia.

Si no tienes claro la ubicación o esta puede ser errónea, no te preocupes. El sistema te avisará si existe algún problema con el reconocimiento de la dirección introducida.

Si tu navegador permite la geolocalización, puedes recuperar tu ubicación inicial desactivando el botón verde (Visualizar) que activaste para ver el formulario.

Una aplicación que te informa del carácter deportivo que más te interesa en la ciudad de Málaga, de una forma tan sencilla como los pasos necesarios para alcanzarlo.



Apéndice C

Normas aplicadas

Para realizar este proyecto software se han seguido los criterios propuestos por dos normas estandarizadas en dos diferentes aspectos del desarrollo.

La primera y más importante mención es la norma: “Norma UNE - 157801 -Proyectos Sistemas de Información”. Esta norma, desarrollada por la Asociación de normalización española AENOR, establece formalmente los requisitos de entrega indispensables para todo proyecto relacionado con los sistemas de la información.

Aunque la ejecución de este proyecto académico y su tiempo asociado no han permitido ejecutar todos los documentos que se relacionan en dicha norma, se ha seguido el patrón de entrega que ésta establece, reuniendo así todos los apartados coherentes con este proyecto en la presente memoria.

Por último, en el ámbito de las referencias del documento, se han seguido las normas establecidas por la APA (American Psychological Association).