UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería en Informática

TFG del Grado en Ingeniería Informática

Geoindoor

Presentado por Juan Pedro Pascual Vitores
en Universidad de Burgos — 30 de noviembre
de 2017
Tutor: Carlos López Nozal

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería en Informática

D. Carlos López Nozal, profesor del departamento de ingeniería civil , área de lenguaje de sistemas.

Expone:

Que el alumno D. Juan Pedro Pascual Vitores, con DNI 13172380G, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Geoindoor de TFG.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 30 de noviembre de 2017

Vº. Bº. del Tutor:

D. Carlos López Nozal

Resumen

La geolocalización es un concepto que se lleva estudiando desde hace años, pero con el avance de la tecnología se ha conseguido acercar la geolocalización al ámbito cotidiano, consiguiendo que llegar a un un determinado lugar nos resulte más fácil.

Habitualmente se ha utilizado la geolocalización para orientarnos en extensiones relativamente amplias. Sin embargo el avance del hardware permite que la geolocalización sea más precisa, con lo que están apareciendo herramientas para que la geolocalización sea aplicada a extensiones de terreno más pequeñas, abriendo un mundo de posibilidades, y precisamente sobre esto trata el proyecto.

Geoindoor es una herramienta con múltiples aplicaciones, que añade servicios de geolocalización dentro edificios. Una aplicación Web permite etiquetar ubicaciones dentro del plano del edificio. Otra aplicación permite consultar las ubicaciones de un edificio , las posiciones en plano del edificio y navegar hasta ellas desde la ubicación actual.

Descriptores

Geolocalización indoor, sistema de localización, búsqueda de lugares, localización en interiores, rutas predeterminadas. ...

Abstract

Geolocation is a concept that has been studied since years, but the advancement of technology has been able to bring geolocation closer to us, because of this we arrive to the places faster.

Usually geolocation has been used to orient in large extensions. However, the hardware advancement allows the geolocation to be more accurate, because of this, there are more tools for the geolocation dedicated to smaller terrain extensions, opening a world of possibilities and the project is about it.

Geoindoor is a tool with multiple applications that add geolocation services within buildings. A web application allows you to tag locations within the building plan. Another application that allows you to consult the locations of a building, positions in the building plan and navigate to them from the actual location.

Keywords

Indoor geolocation, location system, location search, indoor location, predetermined routes. ...

Índice general

Índice de figuras

Índice de tablas

Introducción

"No se dónde está"

Es habitual escuchar esta frase, que multitud de veces irrumpe en nuestras vidas dificultando nuestras tareas, tareas tan simples como encontrar un baño, una oficina, o la cafetería. Estos son los típicos problemas que se encuentra uno cuando acude por primera vez a un lugar, pero es más grave para aquel que ofrece un servicio o unos productos en un determinado lugar y el usuario no lo localiza. Esto puede hacer que el flujo de personas que acudan al establecimiento sea mucho menor, lo que provocaría que el servicio estuviera poco solicitado o que los productos ofrecidos no fuesen vistos por tantas personas como se desearía. Es aún más grave si hablamos de un negocio el cual necesite que los usuarios localicen con rapidez donde se ofrecen los servicios o productos.

De ahí la necesidad de la implementación del proyecto, que propone dar servicios de geolocalización indoor. Esto resolvería diversos problemas en establecimientos como museos, exposiciones, centros comerciales, hospitales, o cualquier otro establecimiento de gran tamaño.

La geolocalización indoor además de servir para encontrar localizaciones determinadas, puede ser aplicada para programar rutas que se amolden a unos intereses, y así hacer de la experiencia de moverse por nuevos lugares algo más fructífero y gratificante.

1.1. Estructura de la memoria

La memoria tiene la siguiente estructura:

- **Introducción:** exposición del problema a resolver y como lo resuelve el proyecto. Estructura de la memoria y listado de materiales adjuntos.
- **Objetivos del proyecto:** objetivos del proyecto.

- **Conceptos teóricos:** explicación de los conceptos teóricos necesarios para comprender el proyecto.
- **Técnicas y herramientas:** técnicas y herramientas utilizadas para la realización de proyecto.
- **Aspectos relevantes del desarrollo:** explicación de dificultades y obstáculos encontrados en la realización del proyecto.
- **Trabajos relacionados:** trabajos relacionados con el proyecto.
- **Conclusiones y líneas de trabajo futuras:** conclusiones obtenidas tras realizar el proyecto y expectativas de futuro.

Junto a la memoria se proporcionan los siguientes anexos:

- **Plan del proyecto software:** sección centrada planificación temporal y viabilidad del proyecto.
- **Especificación de requisitos del software:** sección que se centra en la especificación de requisitos.
- **Especificación de diseño:** sección que se basa en describir la fase de diseño.
- **Manual del programador:** sección centrada en la explicación del código fuente (estructura, compilación, instalación, ejecución, pruebas,etc...).
- **Manual de usuario:** sección centrada en la explicación al usuario de como utilizar la herramienta.

1.2. Materiales adjuntos

Los materiales adjuntados con la memoria son:

- Architect, parte de la herramienta para crear rutas y edificios.
- Viewer, parte de la herramienta encargada de la visualización .
- Código fuente de la Rest API encargada de interactuar con la BD.
- Pruebas de integración.
- Pruebas de estrés y rendimiento.

Recursos en internet:

- Architect (Se lanza en el localhost).
- Viewer (Se lanza en el localhost).
- Rest API https://geoindoorapi.herokuapp.com/ .
- Base de datos de firebase https://geoindoordb.firebaseio.com/ .
- Repositorio del proyecto .

Objetivos del proyecto

A continuación, se especifican los objetivos que promueven la realización de este proyecto.

2.1. Objetivos generales

- Desarrollar una herramienta que permita añadir servicios de geolocalización en interiores.
- Facilitar la localización de lugares dentro de edificios.
- Creación de rutas predeterminadas que permitan al usuario tanto diseñar rutas para él mismo, como para todo el público.

2.2. Objetivos técnicos

- Uso del API proporcionado por la aplicación *Anyplace*
- Uso de Git como sistema de control de versiones junto con la plataforma GitHub.
- Uso de la metodología Scrum.
- Uso de ZenHub como herramienta de gestión de proyectos.
- Uso de JS en la parte del servidor (Node.js).
- Uso de bases de datos no sql.

2.3. Objetivos personales

- Entender como se realiza un proyecto desde su inicio hasta su final.
- Mejorar mis conocimientos y conceptos sobre el desarrollo.

Conceptos teóricos

A continuación se explicaran unos conceptos indispensables para el entendimiento del proyecto.

3.1. Geolocalización

La Geolocalización es le concepto mas importante en este proyecto, y primero hablaremos un poco del API utilizada y después profundizaremos más en el concepto.

En el proyecto, el mapa que ofrece Google Maps es fundamental, ya que nos permite dibujar sobre el plano y añadir imágenes sobre él.

La API utilizada para este proyecto se basa en la API Google Maps, que está escrita en JavaScript [6].

La clase principal es google.maps.Map, cuyo constructor nos permite crear un mapa dentro de un contenedor HTML. Después encontramos google.maps.LatLng, donde LatLng es un punto en coordenadas geográficas: latitud y longitud.

Para más información se puede visitar la documentación de Google Maps donde aparece información más detallada, ya que no profundizaremos en cuales, y como son las clases y funciones de esta API, si no como funciona Google Maps en si.

Google Maps funciona a través de conexión a internet y la función de GPS del dispositivo, las coordenadas de Google Maps están en el sistema WGS842 [1] y se mostrará la latitud y la longitud, positiva para Norte y Este, negativa para Sur y Oeste.

[1] WGS84 *es un sistema de coordenadas geográficas mundial que permite localizar cualquier punto de la Tierra (sin necesitar otro de referencia) por medio de tres unidades dadas.*

Cuando nos referimos a la función GPS, estamos hablando de la función de geolocalización del dispositivo, en la que profundizaremos con más detalle.

Definición: Geolocalización (Geo-localización) está formada por la palabra geo, que tiene origen griego en geos que significa tierra. Y la palabra localización, de las raíces latinas localis (relativo a lugar), que proviene de locus (lugar).

Por lo tanto, la geolocalización consiste en la ubicación de un "lugar" en la "tierra" [4].

Haciendo una definición más exacta, la geolocalización consiste en ubicar o determinar donde se encuentra un objeto o lugar en un determinado espacio.

Cuando nos referimos a objeto no tiene porqué ser inanimado, y el determinado espacio al que nos referimos es la tierra como tal, pero no siempre se la pone como plano de referencia.

Cuando hablamos sobre la geolocalización también debemos hablar de los sistemas de posicionamiento que son los encargados de analizar, manejar y procesar la información geográfica, que nos permite obtener la localización de un objeto u lugar.

El sistema de posicionamiento más conocido en el mundo es el GPS (Global Positioning System), que es un sistema que permite localizar objetos móviles, gracias a la recepción de señales emitidas por un conjunto de satélites. Para determinar las localizaciones este sistema se sirve de 24 satélites y trilateración. Y es de origen estadounidense, otros sistemas de posicionamiento conocidos son el GLONASS, de origen ruso y Galileo europeo [17].

La trilateración es un método matemático para determinar las posiciones relativas de los objetos usando la geometría de triángulos y la triangulación. La trilateración para ubicar el objeto usa las posiciones conocidas de dos o más puntos de referencia, y la distancia entre el objeto y cada punto de referencia [10].
- Explicación matemática de la trilateración:

- Se parte de 3 esferas.

$$r_1^2 = x^2 + y^2 + z^2,$$
$$r_2^2 = (x-d)^2 + y^2 + z^2,$$
$$r_3^2 = (x-i)^2 + (y-j)^2 + z^2,$$

Figura 3.1: Paso 1º trilateración

- Se halla la x restando la segunda a la primera.

$$x = \frac{r_1^2 - r_2^2 + d^2}{2d}.$$

Figura 3.2: Paso 2º trilateración

- Se sustituye la x en la primera esfera, con lo que obtenemos la fórmula de otro círculo.

$$y^2 + z^2 = r_1^2 - \frac{(r_1^2 - r_2^2 + d^2)^2}{4d^2}.$$

Figura 3.3: Paso 3º trilateración

- Se iguala la fórmula anterior a la de la tercera esfera, obteniendo.

$$y = \frac{r_1^2 - r_3^2 - x^2 + (x-i)^2 + j^2}{2j} = \frac{r_1^2 - r_3^2 + i^2 + j^2}{2j} - \frac{i}{j}x.$$

Figura 3.4: Paso 4º trilateración

- Teniendo las coordenadas del punto solución x e y, despejamos la z de la primera esfera .

$$z = \sqrt{r_1^2 - x^2 - y^2}$$

Figura 3.5: Paso 5º trilateración

De esta manera obtendríamos la solución para los tres puntos x, y y z. Al ser una raíz cuadrada la solución puede ser 0, uno o dos resultados.

- z = $\sqrt{x}$, $x \prec 0$: Una de las esferas no intersecciona con las demás (No hay solución real).
- z = 0 : Las esferas interseccionan justo en un punto.
- z = $\pm\sqrt{x}$, $x \succ 0$: Las esferas se interseccionan en 2 puntos.

Se consigue saber la distancia entre objetos y puntos de referencia gracias a que el dispositivo emite una señal y espera una respuesta de los puntos de referencia, el diferencial de tiempo entre el envío y la recepción se utiliza para determinar la distancia.

Como se muestra en las imágenes en la trilateracion en tres dimensiones, los puntos de referencia (satélites) crean una esfera virtual o imaginaria de radio igual a la distancia entre el objeto y el satélite, donde ellos son el epicentro. Se utilizan los puntos de intersección entre esferas para calcular la posición del objeto a localizar.

Los satélites tienen que saber la distancia entre ellos por lo tanto también mandan señales, además para hacer un cálculo correcto de distancias se necesitan tener relojes coordinados.

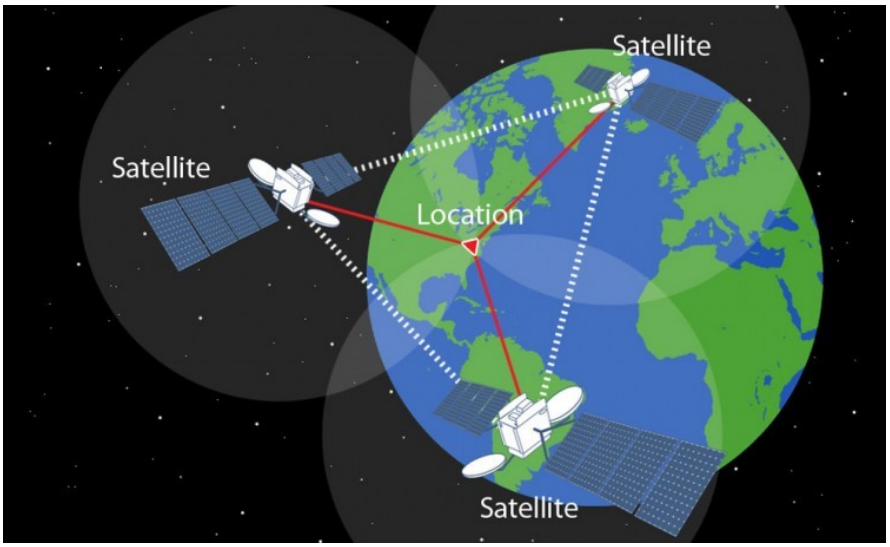

Figura 3.6: Explicacion de trilateración.

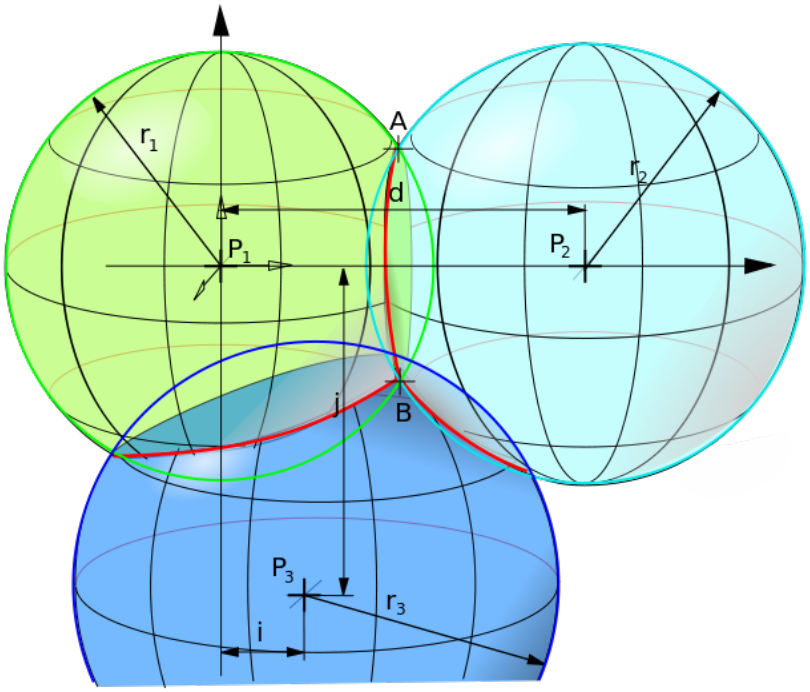

Figura 3.7: Explicacion de trilateración. [18]

La geolocalización por wifi funciona de forma parecida a la de los sistemas de posicionamiento con satélites. Esta vez los puntos de referencia en vez de ser satélites son puntos de acceso wi-fi [21], y hay varias técnicas y tecnologías para hallar la ubicación de los dispositivos, además de la explicada anteriormente. Una de estas técnicas es la de utilizar RSSI, que indica la intensidad de la señal, a mayor intensidad más cerca está del punto de acceso Wi-Fi que proporciona la señal. Pero normalmente para la ubicación en interiores se utilizan ondas de radio ya que se obtiene mayor precisión [7].

3.2. Formatos gráficos escalables

Los formatos gráficos escalables son fundamentales en el proyecto ya que son utilizados a la hora de hacer el dibujado de rutas así como la creación de iconos.

Cuando hablamos de formatos gráficos escalables, nos estamos refiriendo a aquellos formatos digitales basados en objetos geométricos, cuya labor es hacer que la imagen tenga formas más definidas. Los formatos más conocidos son SVG [2], WMF [12], ODG [3] y AI [5].

Debemos entender la importancia de los formatos gráficos escalables en la

mejora de imágenes para hacerlas más geométricas.

WMF

Es un formato de gráficos vectoriales creado por Microsoft, nació a principio de los 90 y ahora está en desuso. Un archivo WMF se utiliza para regenerar una imagen a través Windows GDI [2] . WMF guarda una serie de llamadas a funciones que son enviadas a Windows GDI para regenerar la imagen.

Enhanced Metafile (EMF) es una versión de 32 bits de WMF (16 bits), con comandos adicionales.

ODG

ODG formato de imagen vectorial enlazado con la versión 2 de OpenDocument de OpenOffice, es un estándar abierto para la creación de dibujos vectoriales, utiliza XML, fue creado por Sun Microsystems y OASIS.

AI

Formato de imagen vectorial de Adobe Illustrator, está desarrollado por Adobe Systems para la representación de gráficos vectoriales en formato EPS o PDF. El formato AI está compuesto por rutas conectadas mediante puntos, en lugar de datos de imagen. En principio el formato AI fue un formato nativo llamado PGF hasta que se compatibilizo con PDF mediante la copia completa de datos PGF en el archivo de formato PDF.

SVG

Este es el formato gráfico utilizado para el dibujado de rutas y caminos en el proyecto.

SVG (Scalable Vector Graphics) [20] es un formato de imagen vectorial recomendado y desarrollado por W3C (World Wide Web Consortium), es un estándar abierto, trabaja con gráficos vectoriales bidimensionales estáticos y animados, en formato XML [16]. SVG aparece de forma nativa en multitud de navegadores como Amaya, Mozilla Firefox, Google Chrome, Safari e Internet Explorer. Para navegadores que no tienen SVG de forma nativa existen plugins que permiten mostrar imágenes en formato SVG. SVG permite tres tipos de objetos graficos:

- Elementos geométricos vectoriales

[2] Graphics Device Interface *interfaz de programación de aplicaciones que se encarga del control gráfico de los dispositivos de salida*

  - Rectas
  - curvas

- Mapa de bits
- Texto

Los objetos gráficos con los que trabaja SVG pueden transformarse y agruparse para después ser compuestos en otros objetos renderizados con anterioridad. El dibujado con SVG es dinámico e interactivo, mediante el uso de DOM para SVG (Document Object Model), que mediante ECMAScript [3] o SMIL [4] permite animaciones de gráficos vectoriales sencillas y eficientes SVG tiene una gran cantidad de manejadores de eventos como "onMouseOverz "onClick", que pueden ser asignados a cualquier objeto SVG. La compatibilidad de SVG con estándares web permiten aplicar características como el scripting sobre elementos SVG o XML, de forma simultánea, en la misma página web desde distintos espacios de nombre XML. Las imágenes SVG pueden ser comprimidas con gzip (SVGZ)

[3]ECMAScript *es una especificación de lenguaje de programación publicada por ECMA International. Se empezó a desarrollar en 1996 y se basó en JavaScript, fue propuesto como estándar por Netscape Communications Corporation. Actualmente es el estándar ISO 16262.*

[4]SMIL (Synchronized Multimedia Integration Language)*es un estándar del World Wide Web Consortium (W3C) para presentaciones multimedia, permite integrar audio, video, imágenes, texto o cualquier otro contenido multimedia.*

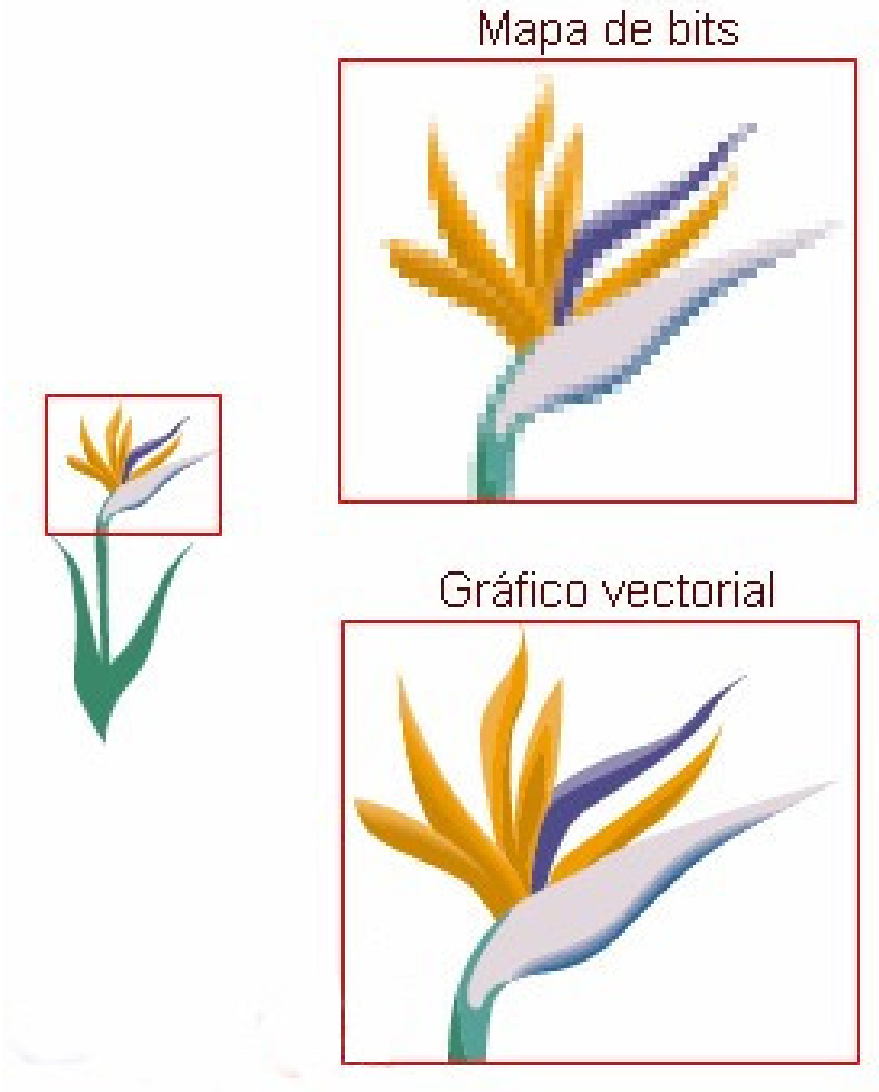

Figura 3.8: explicación diferencia mapa de bits y gráfico vectorial

3.3. Lenguajes de etiquetas

En cuanto al lenguaje de etiquetas o marcado al ser una herramienta para la web la interfaz se nutre de html con lo cual es una parte fundamental del proyecto.

Los lenguajes de etiquetas, o de marcado son utilizados para codificar un documento añadiendo etiquetas al texto, las cuales permiten otorgarle unas características o cualidades específicas. En este proyecto es mu importante ya que facilita tanto el procesado de la información, como a la hora de mostrar

la información [14].

No se debe confundir los lenguajes de etiquetas con los de programación, ya que el lenguaje de etiquetas no tiene funciones aritméticas ni variables.

Podemos diferenciar entre 4 tipos de lenguajes de marcado [1] o de etiquetas [8].

- Marcado de presentación
- Marcado de procedimientos
- Marcado descriptivo
- Lenguajes especializados

También existen lenguajes de etiquetado que son difíciles de enmarcar en uno de estos 4 tipos, ya que comparten características de varios de ellos, como podría ser el caso de HTML, que contiene etiquetas procedimentales, como la B de bold , con otras descriptivas como es BLOCKQUOTE y atributo HREF.

(Los ejemplos representan como se suele ver el lenguaje de marcado)

Marcado de presentación

Indica el formato del texto, y como su propio nombre indica es utilizado para cambiar la presentación de un texto o documento. En este tipo las etiquetas suelen estar ocultas al usuario.

Ejemplo: Microsoft Word

Cuantas veces habremos oído esa frase...
En muchas ocasiones nos encontramos en sitios nuevos

Marcado de procedimientos

El lenguaje de marcado de procedimientos también es utilizado para la presentación del texto, pero en este caso las etiquetas sí que son visibles para los usuarios que editan el texto, y se realiza un procesamiento dependiendo de la etiqueta asociada.

Ejemplo: LaTeX

```
\documentclass[10pt,a4paper]{article}
\usepackage[latin1]{inputenc}
\author{Juan Pedro Pascual Vitores}
\title{Introducción}
```

\begin{document}
\section{Introducción}\labelintroduccion
\texttt{“No se dónde está”}
Cuantas veces habremos oído esa frase...
En muchas ocasiones nos encontramos en sitios nuevos
\end{document}

Marcado descriptivo

Se utilizan etiquetas o marcas para describir el texto, sin especificar cómo deben ser presentados, ni en qué orden.

Este tipo de marcado tiene más usos que lo que aparenta a primera vista, pueden utilizarse para el almacenamiento de metadatos, comunicación de datos (cualquier aplicación pude escribir un documento de texto plano con datos en formato XML), migración de datos (si necesitamos migrar los datos de una base de datos a otra, si ambas trabajan en xml, el trabajo es más sencillo), almacenamiento de gráficos vectoriales (VML) [5] [13], fórmulas matemáticas con XML (MathML)[6], Estructuras moleculares e información cientifica y química con XML (CML) [7] [19]

Ejemplo: XML

<?xml version=“1.0.^encoding=“UTF-8¿>
<Documento>
    <Texto>
        <Parrafo>
            Cuantas veces habremos oído esa frase...
            En muchas ocasiones nos encontramos en sitios
        <\Parrafo>
    <\Texto>
<\Documento >

Lenguajes especializados

Los lenguajes especializados podríamos decir que derivan de los lenguajes de marcado descriptivo, ya que se diferencian con el tipo anterior, en que se

[5] VML *Vector Markup Language es un lenguaje XML destinado a la creación de los gráficos vectoriales en 2D o 3D*

[6] MathML *Mathematical Markup Language lenguaje de marcado basado en XML [9], cuyo objetivo es expresar notación matemática para que distintas máquinas puedan entenderla*

[7] Chemical Markup Lenguaje [11] *es un lenguaje de marcas basado en XML cuyo objeto es expresar información molecular*

especializan en una materia como podría ser la matemática (OpenMath o MathML) [15] [9] o los gráficos (SVG o VRML).

Ejemplo: OpenMath

<OMOBJ xmlns=“url”>
<OMA cdbase=“url”>
<OMS cd=“relation1”name=“eq/>
<OMV name=“x/>
<OMA\>
<\OMOBJ>
Ejemplo: SVG
<svg width=“100”height=”100”>
<circle cx=“50çy=”50r=”40”\>
<\svg4>

3.4. Algoritmo de Dijkstra

En Geoindoor es fundamental el algoritmo de Dijkstra ya que es utilizado para el trazado de rutas a la hora de elegir el camino. A continuación explicaremos en que consiste este algoritmo.

El algoritmo de Dijkstra o algoritmo de caminos mínimos es un algoritmo para encontrar el camino más corto en un grafo, desde un punto de partida u origen, hasta uno los demás puntos del grafo.

El grafo consta de vértices y aristas donde cada arista tiene un peso, mediante el cual se calcula cual es el camino mas corto de un nodo o vértice hasta otro nodo. El principal problema de este algoritmo reside en que es una especialización de la búsqueda de costo uniforme, y no funciona en grafos con aristas de coste o peso negativo, es decir, al elegir siempre el nodo con un peso menor, pueden quedar excluidos nodos que en próximas iteraciones bajarían el costo general del camino.

Algoritmo

- Teniendo un grafo dirigido ponderado de N nodos no aislados.
- Siendo x el nodo inicial.
- Teniendo un vector D de tamaño N en el cual se guardará las distancias desde x al resto de los nodos.

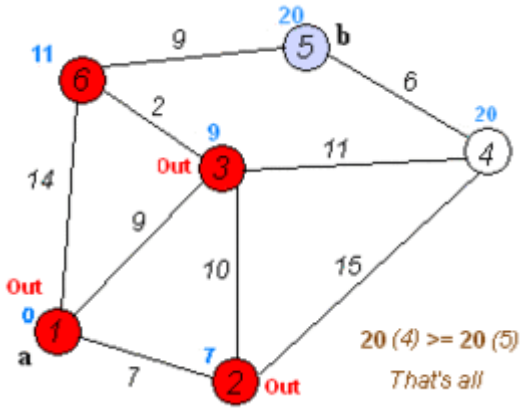

Figura 3.9: Ejemplo de grafo con aplicación del algoritmo Dijkstra

1. Inicializar la distancia de x en D a 0, debido a que la distancia de x a x es 0. Las demás distancias de D deben ser inicilizadas a un valor infinito, ya que al principio son desconocidas.
2. Siendo a = x tomamos a como nodo inicial.
3. Recorreremos todos los nodos adyacentes de a, menos los nodos que ya han sido marcados, llamaremos a los nodos no han sido marcados v_i.
4. Para el nodo actual, calculamos la distancia desde dicho nodo a los vecinos con la fórmula: dt(v_i) = Da + d(a,v_i), y Si dt(v_i) < Dv_i entonces Dv_i = dt(v_i)
5. Marcamos el noda a como completo.
6. Tomamos como próximo nodo actual el de menor valor en D, y volvemos al paso 3 mientras existan nodos no marcados. También se podría hacer almacenando los valores en una cola de prioridad.

Cuando termine el algoritmo, D contendrá las distancias desde x al resto de los nodos.
La complejidad de este algoritmo es O($|V|^2+|A|$) = O($|V|^2$) sin utilizar cola de prioridad, O(($|A|+|V|$) log $|V|$) = O($|A|$ log $|V|$) utilizando cola de prioridad.

- En cada iteración se añade un vértice al subconjunto, para operar y el algoritmo realiza en n-1 iteraciones como máximo.
- Se identifica el vértice con la menor etiqueta entre los que no están en el subconjunto, en cada iteración.

- Se realiza una operación de suma y una comparación para actualizar la etiqueta de cada uno de los vértices que no están en el subconjunto.

Por lo tanto en cada iteración se realiza como máximo 2(n-1) operaciones. Entonces el algoritmo de Dijkstra realiza O(n^2) operaciones para determinar la longitud del camino más corto entre dos vértices de un grafo. (El grafo debe ser ponderado simple, conexo y no dirigido con n vértices determinados)

Técnicas y herramientas

Manejadores de paquetes y Corredeores de tareas
Bower
Grunt

Tabla 4.1: Manejadores de paquetes y Corredeores de tareas utilizados en el proyecto

Formato de intercambio de datos
JSON

Tabla 4.2: Formato de intercambio de datos

Lenguajes y Frameworks
JavaScript
Python
AngularJS
Java (Selenium)
Node.js

Tabla 4.3: Lenguajes y frameworks utilizados en el proyecto

Herramientas para pruebas
Selenium
Webserver Stress Tool

Tabla 4.4: Herramientas utilizadas para realizar los test del proyecto

Control de versiones
Git
ZenHub

Tabla 4.5: Software de control de versiones

Documentación
TEX
Zotero

Tabla 4.6: Documentación

Servicios web
Heroku
Firebase

Tabla 4.7: Servicios web

Interfaz
HTML5
CSS3

Tabla 4.8: Herramientas utilizadas para la interfaz del proyecto

4.1. HTML5

HTML5 es un lenguaje de marcado que se ha utilizado para mostrar al usuario la información así como para aprovechar algunas de sus caractrísticas para el procesamiento de información.

Se ha utilizado porque es muy utilizado, y es soportado por todos los

dispositivos.

4.2. CSS3

CSS3 lenguaje de hojas de estilo en cascada utilizado para dar una impresión diferente al código HTML, hay que decir que permite crear estilos de forma rápida y sencilla, pero al aumentar el número de clases y el anidamiento es muy difícil de mantener.

4.3. JavaScript

JavaScript es un lenguaje de scripting, que en este proyecto es ampliamente utilizado en todos los ámbitos, ya sea para cambiar el flujo de salida (HTML,) como para procesamiento de información y llamadas a un sevidor. Ha sido elegido por su versatilidad y capacidad de adaptarse a cualquier entorno.

4.4. AngularJS

AngularJS (Google) es un framework JavaScript de desarrollo de aplicaciones web en el lado cliente, y utiliza el patrón MVC (Model-View-Controller).

AngularJS adquiere las buenas características que tiene JavaScript, pero su característica mas importante es que permite realizar un gran control sobre el flujo de salida.

4.5. Bower

Bower es un gestor de paquetes para la web, es indispensable, ya que permite manejar los paquetes del proyecto de una forma simple y rápida.

4.6. Grunt

Grunt es un corredor de tareas de javascript, que es utilizado para centralizar tareas y scripts, para tener que evitar procedimientos repetitivos. Si se une con Bower permite trabajar con una gran cantidad de paquetes y scripts de forma sencilla.

4.7. JSON

JSON (JavaScript Object Notation) es un formato de intercambio de datos. Es simple tanto a la hora de leerlo, como de escribirlo, es ideal para la transmisión de objetos por la red, y es esa la razón por la que es utilizado.

4.8. Node.js

Node es un intérprete Javascript en el lado del servidor. Permite construir aplicaciones muy escalables y maneja miles de conexiones simultáneas en una sólo una máquina física.

En el proyecto ha sido utilizado para crear la Rest API para manejar la base de datos, se ha elegido por su simpleza, aunque cabe destacar que debido a que te permite manejar muchas opciones, es fácil equivocarse y realizar acciones que no se tenían pensadas.

4.9. python

Python es un lenguaje de programación interpretado que busca ser un código legible. Es un lenguaje de programación multiparadigma, programación imperativa y programación funcional.

En este proyecto ha sido utilizado de manera escasa, pero ha sido muy útil para realizar pruebas individuales.

4.10. Git

Git es un software de control de versiones diseñado por Linus Torvalds, se ha elegido y usado porque es usado por el sevicio web heroku así como por GitHub.

4.11. Latex

TEX y Latex han sido utilizados para documentar ya que a pesar de que al principio se hace un poco difícil trabajar con ello, los resultados son excelentes.

4.12. Heroku

Heroku es una plataforma de servicio de computación en la Nube que soporta distintos lenguajes de programación. En el proyecto ha sido utilizado para sostener es servidor con Node.js con la Rest API. Se ha elegido Heroku por recomendación del tutor, asi como su facilidades para comenzar en ello, ya que ofrece servicios gratuitos y muy buenos tutoriales.

4.13. Firebase

Firebase es una plataforma de desarrollo de aplicaciones móviles y web, es un sevicio ofrecido por google y permite tanto el hosting, base de datos,

almacenamiento. Ha sido elegido en este proyecto por su base da datos (JSON) no SQL en tiempo real, que permite trabajar con la base de datos de forma muy rápida y sencilla. El utilizar una base de datos JSON al igual que AnyPlace (aplicación en la que se basa Geoindoor), si se quisiera aunar bases de datos, se conseguiría de una forma relativamente sencilla.

Aspectos relevantes del desarrollo del proyecto

En el desarrollo de este proyecto han surgido varios obstáculos que han sido resueltos de diversas maneras. A continuación los explicaremos.

5.1. Análisis e investigación de la aplicación

El principal problema que me encontré fue entender la aplicación sobre la que trabajar y como hacerlo, el principal problema que tuve es que no había una verdadera documentación donde se explicara como funciona al aplicación y cuales son sus funciones y métodos. Con lo cual tuve que investigar y analizar la aplicación a conciencia hasta entenderla.

Otro gran problema que surgió fue entender y crear el servidor que ofrecían, hasta que se decidió trabajar con la aplicación por separado, es decir Architect por un lado y Viewer por otro y después unificarlos para crear el servidor.

5.2. Viewer

El problema que surgió con ello, fue a la hora de crear la función que recoge la posición del usuario para compararla con los puntos determinados. El problema surgió porque en la función getPointLocationUser, watchPosition además de darte las coordenadas del usuario en le momento también te bloquea la la muestra de la posicion del usuario (el punto azul que indica donde esta el usuario), eso es debido a que watchPosition tiene una función callback que bloqueaba. Se consiguió resolver usando clearWatch, para evitar el callback.

5.3. Heroku

El principal problema que tuve con ello es que no lo conocía, por lo que tuve que adaptarme y aprender su funcionamiento, aunque hay que decir que los primeros pasos los facilitan gracias a sus tutoriales.

Una vez que se aprendió como funcionaba heroku se comenzo a realizar la Rest API cuyo principal problema fue entender como funcionaba Node.js y las llamadas POST Y GET en Node.js con express.

También es importante decir que con las llamadas al servidor surgieron problemas CORS (access-control-allow-origin), que se ha solucionado añadiendo las cabeceras de Access-Control-Allow-Origin.

5.4. Firebase

Lo que sucedió con Firebase esta relacionado con su base de datos JSON, cuya estructura de base de datos tuve que idear para que se pudiese conseguir una gran cantidad de información de calidad evitando cantidad de información de poco valor.

La idea es utilizar keys que iguales y añadir una palabra clave para utilizar esa clave como otra key. Es decir la key id2 es la key de un edificio y la key id2rutas tiene las rutas del edificio de key id2.

5.5. Architect

El principal problema en la parte de la aplicación de creación de caminos,edificios, y pois, es que las llamadas a al servidor de AnyPlace de determinas url daba problemas de CORS, lo que se soluciono obteniendo información a través de AngularJS

5.6. Otros problemas

Existieron otros muchos problemas pero que no son comentables ya que no son de tanto interés como estos.

Pero si hay que comentar alguno problema más que me he encontrado, ha sido el uso de jquery para las llamadas, sobre todo con el dataType, el hosting de firebase que no permite hacer llamadas fuera de su dominio y los dynos de heroku.

Trabajos relacionados

Los trabajos que estan relacionados con este son Google Maps, AnyPlace y goindoor.

6.1. Google Maps

Google Maps es la semilla del proyecto ya que AnyPlace se basa en ello. Google Maps ofrece un API en JavaScript que ha sido ampliamente utilizado en este proyecto, la principal ventaja que tiene utilizar el API de Google Maps es el soporte y la atención, además la amplia comunidad que trabaja con Google Maps hace que se resuelvan la dudas fácilmente.

La principal diferencia ente Google Maps y Geoindoor es que una Google Maps busca la geolocalización en extensiones amplias de terreno mientras que Geoindoor la busca en interirores, aunque sigue marcando rutas en largas distancias también.

6.2. AnyPlace

AnyPlace es la aplicación en la que se basa Geoindoor, con lo que las principales diferencias entre AnyPlace y Geoindoor son que Geoindoor además de cambios de estilo ofrece el agregado de rutas al edificio y su interactividad.

6.3. goindoor

goindoor es otra sistema utilizado para la geolocalización indoor y la principal diferencia entre Geoindoor y goindoor es que, goindoor utiliza bacons, o anclajes que son utilizados de sensores para obtener la geolocalización.

Conclusiones y Líneas de trabajo futuras

Para concluir puedo decir que el desarrollo ha sido difícil, sobre todo los primeros pasos que han consistido en el entendimiento del sistema y sus funciones.

A primera vista se podría pensar que la implementación de funcionalidades es sencilla pero no es tan sencillo una vez estas trabajando en ello. Sobre todo porque algunas llamadas de la API Rest de AnyPlace se encuentra capada.

En cuanto a los resultados, se ha conseguido integrar el agregado de rutas y su interacción.

En cuanto a las mejoras, se podría mejorar el viewer enlazando las rutas para su interacción. Ya se intentó pero debido a diversos problemas (CORS) no se ha conseguido para la entrega.

El problema del enlazado de las rutas con el Viewer se puede solucionar cogiendo la información a través de los controladores.

Este proyecto tiene un gran potencial y se podría utilizar en diversas situaciones, como la búsqueda de oficinas en infraestructuras publicas, rutas interactivas en centros comerciales o museos etc..

Bibliografía

[1] 1.1 Lenguajes de marcas: tipos y clasificación de los más relevantes.

[2] Curso gratis de Flash CS5. aulaClic. 9 - Básico: Gráficos vectoriales y mapas de bits.

[3] Definición de odg (formato de archivo, extensión).

[4] Diccionario Etimológico castellano en línea.

[5] Extensión de archivo AI.

[6] Google Maps JavaScript API V3 Reference | Google Maps JavaScript API.

[7] Las mejores aplicaciones de geolocalización.

[8] Lenguaje de marcado - EcuRed.

[9] Usos del eXtensible Markup Language (XML), December 2003.

[10] Trilateración, November 2015. Page Version ID: 87006146.

[11] Chemical Markup Language, December 2016. Page Version ID: 95720127.

[12] Metaarchivo de Windows, August 2016. Page Version ID: 93104087.

[13] Vector Markup Language, May 2016. Page Version ID: 91233187.

[14] Lenguaje de marcado, May 2017. Page Version ID: 99029240.

[15] OpenMath, April 2017. Page Version ID: 98718481.

[16] Scalable Vector Graphics, June 2017. Page Version ID: 100057888.

[17] Sistema de posicionamiento global, June 2017. Page Version ID: 100017790.

[18] Heriberto Arribas Abato. Español: Animación del proceso geométrico de Trilateración., April 2012.

[19] DesarrolloWeb.com. Objetivos y usos del XML.

[20] Iván Lasso. Formatos de imagen: imágenes vectoriales, February 2015.

[21] Escrito por Sergio De Luz. Wi-Fi Location: Qué es, cómo funciona y para qué sirve este estándar de geoposicionamiento en interiores con Wi-Fi, March 2017.