UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería en Informática

TFG del Grado en Ingeniería Informática

Geoindoor
Documentación Técnica

Presentado por Juan Pedro Pascual Vitores
en Universidad de Burgos — 16 de diciembre de 2017
Tutor: Carlos López Nozal

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

La planificación es una parte muy importante del proyecto, ya que es la parte donde se estima el tiempo que se va consumir, el esfuerzo, y si ese proyecto va dar beneficios económicos.

He de decir que a la planificación de los tiempos y del trabajo se hace fundamental si se quiere ser productivo, existen factores no considerados como, que tipo de horas se ha invertido y con que lapsus de tiempo, que afectan a la productividad y a los tiempos. Existen muchos factores que pueden alterar estos parámetros, a continuación citaré algunos.

- Horas de desarrollo después de una jornada de trabajo.
- Horas de desarrollo dispersas, por ejemplo, se desarrolla durante una hora se para durante horas y se vuelve a retomar, lo que hace que la productividad disminuya .
- Curva de aprendizaje, en este proyecto el aprendizaje ha sido fundamental y en muchas ocasiones costoso, si se junta este factor con los 2 anteriores la curva de aprendizaje crece con facilidad.
- Horas de desarrollo de poca calidad, es importante tener un lugar donde trabajar donde no se pierda la concentración y el hilo de forma continua.

En este anexo se va analizar los factores por los cuales es conveniente o no realizar un proyecto.

- Planificación temporal

- Viabilidad del proyecto
- Viabilidad económica
- Viabilidad legal

A.2. Planificación temporal

La planificación del tiempo es fundamental, ya que es muchas veces las prisas no nos permiten razonar con serenidad, y tener mucho tiempo lleva a la procrastinación. Entonces lo ideal es llevar un trabajo constante y marcado, es decir crearse horarios de trabajo e intentar cumplirlos lo mejor posible.

Se ha utilizado una metodología ágil Scrum, basada en *Sprints*, y ZenHub, para el seguimiento de issues y tareas. La plataforma GitHub y ZenHub nos permiten administrar el trabajo y las tareas de forma más dinámica y sencilla. Se ha intentado marcar *Sprints* con una fecha final determinada, pero no ha sido posible debido a las dificultades a la hora de estimar el tiempo de trabajo y que no siempre existía la posibilidad de mantener un desarrollo en el tiempo, es decir, no siempre existían horas de calidad para aplicar al desarrollo. De todas formas si que se han determinado *Sprints* con un inicio y un final, los cuales detallaremos a continuación.

Estos son los pasos que se siguieron en el desarrollo:

- Aplicar una estrategia de desarrollo incremental a través de iteraciones y revisiones.
- La duración media de los *Sprints* debe ser de una semana.
- Al finalizar cada *Sprint* se entrega una parte del producto.
- Se realiza reuniones de revisión al finalizar cada *Sprint* y se vuelve a planificar *Sprint*.
- En la planificación del *Sprint* se genera una pila de tareas.
- Estas tareas se estiman y priorizan en un tablero.
- Para monitorizar el progreso del proyecto se utiliza gráficos

Se han seguido todos estos pasos, pero como ya se ha comentado anteriormente la finalización de los *Sprints* no estaba determinada así que su final se tomará cuando se acabaron todos los issues pertenecientes al mismo. Los gráficos en muchas ocasiones no son totalmente representativos del trabajo ni el tiempo utilizado ya que en muchas ocasiones el tiempo invertido no era de calidad, o no se podía invertir tiempo.

Sprint 0 (26/3/2017 - 3/4/2017)

El *Sprint* 0 fue el primer *Sprint* que se creo y sirvió de toma de contacto, se afianzaron las ideas y se especificaron los objetivos del proyecto. Se puede decir que ha servido para afianzar ideas sobre el proyecto.

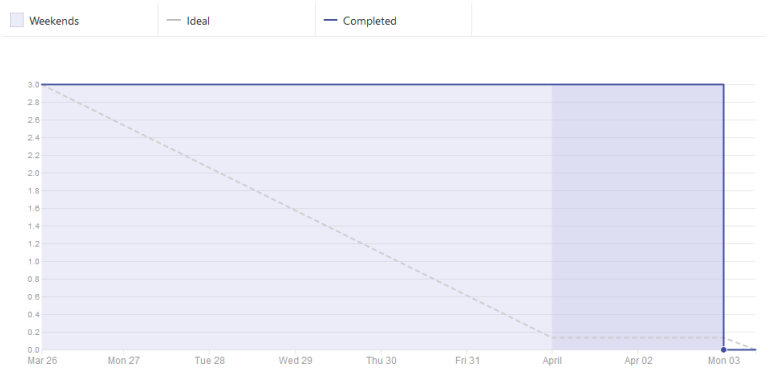

Figura A.1: Gráfica Sprint 0

Sprint 1 (10/4/2017 - 10/8/2017)

El *Sprint* 1 ha sido uno de los más importantes ya que es donde se ha centrado la mayoría del aprendizaje y una gran cantidad de desarrollo, este *Sprint* ha sido en el que se volcó por primera vez la herramienta Geoindoor en su conjunto, por lo que se crearon commits de gran tamaño.

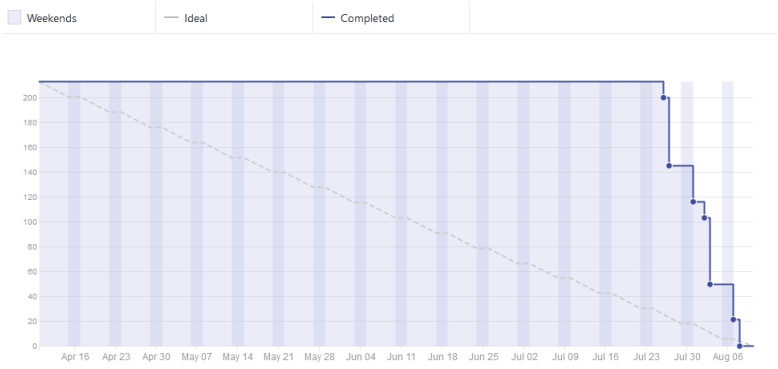

Figura A.2: Gráfica Sprint 1

Sprint 2 (10/8/2017 - 30/8/2017)

Si miramos el *Sprint* 2 podemos observar que la fase de aprendizaje está mas avanzada y que el desarrollo se hace menos costoso, pero también esta

cargado de menos tareas que el anterior. En este proyecto cabe destacar que el aprendizaje no acaba siempre se encuentra algo nuevo.

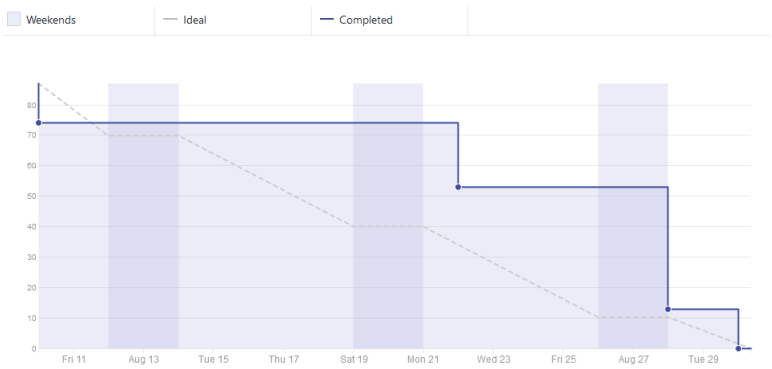

Figura A.3: Gráfica Sprint 2

Sprint 3 (25/9/2017 - 12/10/2017)

El *Sprint* 3 está más dedicado al desarrollo en la parte del viewer, con lo cual se tuvo que volver a investigar y a analizar esa parte de la herramienta para entender cual era la mejor manera para introducir modificaciones. Una ventaja es que entre Architect y Viewer hay un gran parecido, por lo tanto el esfuerzo no fue tanto como la primera vez. También decir que se tuvieron que hacer cambios para que las búsquedas de edificios fueran mas efectivas, lo que llevo un gran esfuerzo.

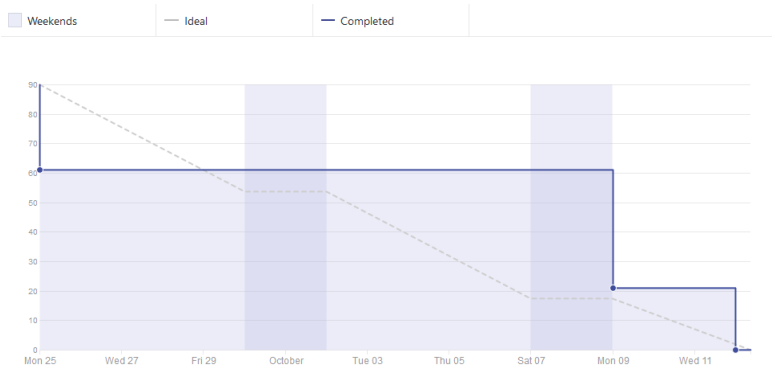

Figura A.4: Gráfica Sprint 3

Sprint 4 (17/10/2017 - *)

El *Sprint* 4 es la última iteración antes de la entrega en ella se encuentran los últimos retoques a la herramienta así como, las pruebas realizadas y

elementos dedicados a la documentación. No tiene fecha final ya que abarca, como hemos dicho antes, los últimas pinceladas a la herramienta y la documentación. No encontramos gráfico por no tener fecha final.

A.3. Estudio de viabilidad

En cuanto al estudio de la viabilidad, se ha investigado y no hay mucha competencia en cuanto a geolocalización indoor basada en GPS. La mayoría de servicios de geolocalización indoor son de pago, y utilizan bacons para la localización. Lo que hace a Geoindoor uno de los sistemas más atractivos.

Viabilidad económica

En cuanto a la viabilidad económica, el desarrollo de la aplicación ha sido totalmente gratuito, se ha aprovechado los servicios gratuitos de Heroku y Firebase que aunque tengan restricciones singuen dando rendimiento.

A continuación haremos un análisis de la viabilidad económica que puede tener Geoindoor.

Concepto	Coste
Servicio Heroku	0€
Servicio Firebase	0€
Horas de desarrollo	0€
Total:	0€

Tabla A.1: Gastos de desarrollo

Se cuenta que el desarrollo ha sido gratuito ya que se ha realizado sin ninguna remuneración económica, en cuanto a Firebase y Heroku, los servicios utilizados son gratuitos y no compensa cambiarlos a la tarifa de pago hasta que no haya un uso por parte de los usuarios, que sobrepase el rendimiento de ambos servicios.

Si nos imaginamos que los ingresos son 2000€y contamos al propietario de la herramienta como un autónomo de menos de 1 año, los gastos serían los siguientes.

En total el beneficio obtenido sería de unos 563,6€.

La herramienta no esta creada para sacarla un beneficio económico si no dar un servicio. De todas formas si se pretendiera obtener beneficios sería a partir de anuncios en la propia herramienta.

Concepto	Coste
Servicio Heroku	0€
Servicio Firebase	0€
Cuota de autónomo	264€
Ordenador	300€
Sistema Operativo(Ubuntu)	0€
Linea de teléfono	100€
Internet	100€
Marketing	0€
Seguridad Social	100€
IVA	420€
IRPF	152.4€
Total:	1436.4€

Tabla A.2: Gastos de comercialización

En cuanto a la distribución no se piensa distribuir a través de ninguna plataforma, se piensa dejar abierto a descarga desde el propio GitHub. En cuanto al marketing, se pretende promocionar la aplicación a través de las redes sociales y foros para minimizar así los gastos. Hemos considerado también que se deben incluir en los gastos de comercialización la linea telefónica, el ordenador, el SO e internet, ya que se hace necesario si se quiere dar un mínimo de soporte y mantenimiento.

Viabilidad legal

En cuanto a la viabilidad legal AnyPlace posee una licencia MIT de código abierto lo que nos permite modificar libremente la herramienta. El uso de los mapas de Google Maps y sus facilidades también se encuentra abierto al publico, así que en primera instancia, no nos encontraríamos ningún problema. Por otra parte hay un tema muy importante a tener en cuenta y es el derecho a la intimidad, y que puede afectar a la ley Orgánica 1/1982, de 5 de mayo, sobre protección civil del derecho al honor, a la intimidad personal y familiar y a la propia imagen.

A lo que nos referimos es que Geoindoor no se hace responsable de la colocación de los planos por la parte de los usuarios, pero se guarda el derecho a eliminar y borrar cualquier cuenta o plano si esta no respeta el derecho a la intimidad de la gente.

Otro tema a tener en cuenta es si se permitiría subir planos de infraestructuras críticas o edificios públicos que se pudieran utilizar con fines dañinos para la sociedad.

Teniendo todos estos factores en cuenta, se concluye con la certeza de que Geoindoor es legal pero, tiene cierto potencial que lo puede convertir en peligroso y se debe controlar.

Apéndice B

Especificación de Requisitos

B.1. Introducción

Este anexo recoge la especificación de requisitos, la cual define el comportamiento del sistema desarrollado, es utilizado tanto por el cliente y por el equipo para tener una concepto generalizado de lo que se quiere, para que ambas partes lleguen a una idea común. Para este fin muchas veces son utilizados diagramas los cuales aclaren el concepto y lo que se necesita.

Podemos distinguir entre requisitos funcionales y requisitos no funcionales.

- **Requisito funcional:** es aquel que especifica una función o un servicio que debe cumplir un sistema.
- **Requisito no funcional:** este especifica restricciones sobre el diseño y la implementción del sistema.

Las características de una buena especificación de requisitos son definidas por:

- **Completa:** deben estar todos los requerimientos y todas sus relaciones.
- **Consistente:** todos los requerimientos y otros documentos de especificación se debe relacionar de forma coherente.
- **Inequívoca:** se debe ser claro y darse a entender.
- **Correcta:** el sistema debe cumplir con todos los requisitos especificados.
- **Trazable:** los requerimientos deben de estar ordenados y organizados de una manera tal, que sea sencilla su identificación.

- **Priorizable:** los requisitos deben estar organizados de forma jerárquica, según su importancia.
- **Modificable:** los requerimientos deben ser fácilmente modificables.
- **Verificable:** se debe poder probar todo requerimiento.

B.2. Objetivos generales

- Desarrollar un sistema que permita la localización del individuo en interiores.
- Desarrollar un sistema que permita la localización de lugares en interiores.
- Desarrollar un sistema que permita el trazado de rutas en interiores.
- Desarrollar un sistema que permita mostrar lugares y rutas predefinidas en interiores.

B.3. Catalogo de requisitos

A partir de los objetivos anteriormente citados obtenemos los siguientes requisitos.

Requisitos funcionales

- **RF-1 Gestión de edificios:** la herramienta debe permitir la gestión de un edifico.
  - **RF-1.1 Emplazar edificio:** la herramienta debe permitir la colocación de un edifico sobre el plano de localización
  - **RF-1.2 Detallar información del edificio:** se debe permitir adjuntar información al edificio (nombre, descripción, etc...).
  - **RF-1.3 Emplazar plano:** la herramienta debe permitir la colocación del plano del edifico sobre el plano de localización.
  - **RF-1.4 Redimensionar y mover el plano:** se debe permitir el movimiento y redimensión del plano añadido.
  - **RF-1.5 Borrar edifico:** se debe permitir el borrado de un edificio.
- **RF-2 Gestión de pois:** la herramienta debe permitir la gestión de pois.

  - **RF-2.1 Emplazamiento de pois:** se debe permitir la colocación de puntos de interés sobre el plano.
  - **RF-2.2 Detallar información del poi:** se debe permitir adjuntar información al poi (nombre, descripción, tipo, etc . . .).
  - **RF-2.3 Ubicar poi:** la herramienta debe permitir la colocación del poi sobre una localización del plano de un edifico.
  - **RF-2.4 Enlazar pois:** se debe permitir crear un camino entre pois.

- **RF-3 Gestión de rutas:** se debe permitir la gestión de rutas predefinidas.
  - **RF-3.1 Crear ruta:** se debe permitir crear una ruta predefinida o predeterminada.
  - **RF-3.2 Detallar información de la ruta:** se debe permitir agregar información a la ruta, como el nombre o la planta (el número de planta se añade de forma automática, lo que nos permite hacer rutas multinivel).
  - **RF-3.3 Añadir pois a ruta:** la herramienta debe se capaz de añadir puntos de interés a la ruta.
  - **RF-3.4 Trazar ruta:** la herramienta tiene que ser capaz de mostrar el trazado de la ruta.
  - **RF-3.5 Seleccionar ruta:** la herramienta debe permitir elegir entre las diferentes rutas existentes.
  - **RF-3.6 Limpiar ruta:** la herramienta debe permitir dejar de mostrar el trazado de la ruta.
  - **RF-3.7 Borrar ruta:** la herramienta debe permitir el borrado de una ruta elegida.

- **RF-4 Visualización de edificios:** el edificio debe estar disponible para todos los usuarios si el usuario administrador o creador así lo decide.
  - **RF-4.1 Buscar edificio:** la herramienta debe permitir que el edificio se pueda encontrar y visualizar a partir de su identificador o nombre.
  - **RF-4.2 Visualizar información:** la herramienta debe permitir que la información del edificio sea visible a los usuarios visitantes, por lo tanto se debe visualizar también el plano.

- **RF-5 Visualización de pois:** los pois de un edificio deben ser visibles
  - **RF-5.1 Visualizar información:** la herramienta debe permitir que la información del poi sea visible a los usuarios visitantes.

  - **RF-5.2 Marcar un poi:** la herramienta debe permitir a los usuarios marcar o seleccionar un poi.
  - **RF-5.3 Proyectar camino de forma automática:** la herramienta debe permitir que al seleccionar un poi se pueda dibujar un camino automático desde la ubicación del usuario hasta el poi indicado.

- **RF-6 Visualización de rutas:** la herramienta debe permitir a los usuarios visualizar las rutas predefinidas del edificio seleccionado
  - **RF-6.1 Seleccionar ruta:** la herramienta debe permitir seleccionar rutas definidas del edificio.
  - **RF-6.2 Mostrar ruta:** la herramienta debe permitir mostrar el trazado de la ruta.
  - **RF-6.3 Limpiar ruta:** la herramienta debe permitir dejar de mostrar el trazado de la ruta.

Requisitos no funcionales

- **RF-1 Intuitiva:** la herramienta debe ser intuitiva y fácil de entender para que así el usuario se acostumbre fácilmente a su uso.
- **RF-2 Usable:** la herramienta debe ser fácil de usar, para atraer tanto a gente con voluntad de aprender como no.
- **RF-3 Portable:** la herramienta debe de necesitar de instalación alguna más allá de sus dependencias lógicas (Python y navegador).
- **RF-4 Rápida:** la herramienta debe proporcionar un servicio rápido, y con resultados tempranos para que el usuario este a gusto trabajando desde el primer instante.
- **RF-5 Para todo el mundo:** la herramienta debe tener como final, que un gran espectro de la población la use para ser así mas útil para la sociedad.

B.4. Especificación de requisitos

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de datos

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía