UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Widget Android para la consulta de los horarios de llegada de los autobuses municipales (EMT) de la ciudad de Valencia

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Sergio Cócera Soriano

Tutores: Juan Luis Posadas-Yagüe
Jose Luis Poza Luján

2014/2015

Agradecimientos

Gracias a mis tutores,
el *Doctor Juan Luis Posadas Yagüe* y el *Doctor Jose Luis Poza Luján*,
por la ayuda prestada para la realización de este proyecto,
pero sobre todo por el apoyo recibido en todo momento para seguir adelante.

Resumen

Este proyecto se centra en el desarrollo de un widget para dispositivos móviles basados en el sistema operativo Android, para la obtención de las próximas llegadas de los autobuses de la red metropolitana de Valencia en la parada en la que el usuario se encuentre y de manera totalmente automática sin la intervención de éste.

La obtención de la información de llegadas se lleva a cabo mediante la utilización de una serie de servicios web a través de la API que el Ayuntamiento de Valencia proporciona a todos los desarrolladores que deseen explotarla. Dichos servicios web permiten la comunicación entre el widget o aplicación y los servidores remotos de EMT Valencia.

En este documento se analiza el entorno y el estado de las aplicaciones actuales existentes, se especifican los requisitos funcionales y no funcionales, se establece la arquitectura del sistema y se propone una interfaz de usuario.

Respecto a la implementación, se describen las herramientas y el entorno de desarrollo utilizado, las API de las que el sistema hace uso y la comunicación entre los distintos módulos del sistema que hacen posible el funcionamiento y la obtención y presentación de datos final.

Palabras clave: Android, widget, transporte, autobuses, EMT, Valencia, servicios web, API.

Tabla de contenidos

Índice de tablas

Índice de ilustraciones

1. Introducción

1.1 Presentación

El Ayuntamiento de Valencia, a través de Internet, pone a disposición de todos los ciudadanos una serie de datos pertenecientes al sector público y que están accesibles con el objetivo de que permitan llevar a cabo desarrollos, utilidades y nuevos servicios por parte de empresas, instituciones y particulares.

En el tiempo en que la API Ayuntamiento de Valencia viene siendo accesible, han surgido una serie de aplicaciones y utilidades que permiten el acceso a una serie de datos de utilidad para el ciudadano: la localización y otras características de los monumentos falleros, el número de bicicletas disponibles en el servicio Valenbisi, la localización y plazas libres en los aparcamientos públicos de la ciudad…

Entre estas utilidades, en su mayoría aplicaciones para dispositivos móviles, algunas de las más solicitadas son aquellas que ofrecen información referente al transporte público. Esta demanda, junto a la facilidad de utilización de los datos públicos que el Ayuntamiento entrega en una serie de servicios web con su correspondiente documentación, consigue que más desarrolladores se interesen por este tipo de aplicaciones, intentando aportar nuevas funcionalidades no existentes en anteriores aplicaciones o permitiendo y distribuyendo las consultas a servicios de una forma más intuitiva.

1.2 Motivación

Mediante estas utilidades y aplicaciones, los usuarios de dispositivos móviles (teléfonos, tablets…) pueden consultar desde cualquier lugar en que se encuentren los horarios de llegada de su transporte, pudiendo así tener una mejor planificación del recorrido desde su casa o lugar de trabajo hasta las paradas, o bien calcular el tiempo de espera si ya se encuentran en éstas.

Haciendo un análisis de la oferta existente actualmente en el sector de las aplicaciones para dispositivos móviles con sistema operativo Android, nos damos cuenta que se han publicado algunas aplicaciones móviles que presentan las llegadas de los autobuses de la EMT, pero a diferencia de otras ciudades, no se ha incluido la posibilidad de utilizar un widget Android para el acceso en un vistazo en la pantalla de inicio a la información de llegadas.

1.3 Objetivos

El objetivo del proyecto *Mi Bus Valencia* es crear una aplicación Android basada principalmente en un widget para dispositivos móviles con sistema operativo Android. Este widget es una pequeña aplicación que no se ejecuta a pantalla completa, sino que ofrece una pequeña ventana que suele ser colocada en el escritorio o pantalla de inicio de los dispositivos (generalmente junto a otros widgets en una configuración flexible para el usuario, el cual selecciona qué widgets son de su interés) y que permite consultar las llegadas de las líneas pertenecientes a cualquier parada de autobús perteneciente a la red de autobuses municipales de la empresa EMT de la ciudad de Valencia.

Este widget, al poder ser colocado en la pantalla de inicio de cualquier dispositivo Android adaptándose a las características de la pantalla, posibilita la consulta de las llegadas en un vistazo y sin la necesidad de abrir ninguna aplicación, actualizándose cada minuto para recibir los cambios en los tiempos de llegada.

1.4 Descripción del documento

El presente documento está organizado por apartados:

- **Apartado 1:** Es el apartado actual y contiene una breve introducción al proyecto.
- **Apartado 2:** En él se expone el entorno del producto con un estudio de otras aplicaciones similares en el mercado.
- **Apartado 3:** En este apartado se detalla la especificación de requisitos mediante el estándar IEEE830.
- **Apartado 4:** Se introducen los aspectos del diseño de la aplicación, arquitectura del sistema y diseño de interfaces.
- **Apartado 5:** En este apartado se trata información relativa a la implementación del producto.
- **Apartado 6:** En el último apartado se exponen las conclusiones finales, problemas encontrados y soluciones aplicadas, las mejoras aportadas por este producto frente a otros productos de similares características ya existentes en el mercado y las posibles ampliaciones futuras.

2. Entorno

2.1 Introducción

Previo al desarrollo de una aplicación Android, es conveniente realizar un pequeño estudio de las aplicaciones similares que ya existen en el mercado para este sistema operativo y/o de las funcionalidades que éstas todavía no han implementado.

Además, un aspecto muy importante que hay que tener en cuenta antes de llevar a cabo el desarrollo de una idea para dispositivos móviles con sistema operativo Android y debido a las distintas versiones existentes de dicho sistema, es en qué rango de versiones Android funcionará la aplicación diseñada.

Con la información anteriormente comentada, se lleva a cabo un análisis para extraer el máximo de conclusiones útiles con la finalidad de colocar la aplicación en el mercado de forma que esté dirigida al sector más adecuado y tenga una máxima visibilidad.

2.2 Aplicaciones similares

A la hora de diseñar el proyecto y dado que ya existen aplicaciones con las que se pueden consultar los horarios de llegada de los autobuses EMT de Valencia, se ha llevado un estudio previo de las aplicaciones existentes actualmente en Google Play.

El objetivo de este estudio es detectar las deficiencias o aspectos a mejorar en estas aplicaciones y concluir si se podía mejorar la funcionalidad, aportar alguna función nueva o si por el contrario, no tenía interés desarrollar algo que terminase siendo más de lo mismo ya existente.

Las aplicaciones existentes para Android en Google Play que ofrecen servicio de previsión de llegadas para la red de autobuses EMT de la ciudad de Valencia son las siguientes:

EMT Valencia

Es la aplicación oficial desarrollada por la EMT de Valencia, además de la más completa. Ofrece prácticamente todo lo que ofrece la página web oficial de la EMT de Valencia, en versión reducida para dispositivos móviles. No obstante, es algo lenta a la hora de obtener la lista de paradas y como todas, necesita que el usuario la abra, acceda a la opción del menú y teclee el código numérico de la parada en la que se encuentra para recibir entonces la lista de llegadas. Aún y siendo tan completa, se echa de menos un widget.

Ilustración 1. Captura de pantalla EMT Valencia

eBus

Fue una de las primeras aplicaciones que apareció en Google Play para consulta de llegadas de autobús, es sencilla y ofrece las funciones básicas (consulta de llegadas introduciendo el código de parada, saldo de viajes en el bonobús y paradas cercanas en el mapa) ocupando tan sólo 161Kb.

Ilustración 2. Captura de pantalla eBus

ValenBus

También de las primeras aplicaciones en Google Play para consulta de llegadas, no ofrece ninguna novedad significativa respecto a otras como *eBus*, salvo algunos juegos para pasar el tiempo mientras se espera la llegada del autobús.

Ilustración 3. Captura de pantalla ValenBus

Urban Step

Es prácticamente un clon de *eBus*, incluso en el diseño por pestañas (consulta de llegadas, saldo de viajes y paradas cercanas en el mapa). Como novedad, incorpora la opción de recibir la información en idioma inglés.

Ilustración 4. Captura de pantalla Urban Step

Buses Valencia

Es probablemente la aplicación más sencilla de todas las existentes para la consulta de las llegadas y cuenta tan sólo con una pantalla desde donde se realiza la consulta. Permite consultar también el saldo de viajes del bonobús.

Ilustración 5. Captura de pantalla Buses Valencia

Valencia Bus

Muy similar a *eBus* y *Urban Step* incluso en el diseño. Al igual que las aplicaciones mencionadas, permite la consulta de las llegadas por código de parada y del saldo de viajes.

Ilustración 6. Captura de pantalla Valencia Bus

Transport Stops

Es una aplicación diseñada fuera de España y no dedicada a Valencia. Ofrece información de llegadas para un gran número de ciudades, entre las que se incluye Valencia. La consulta se realiza a través de un mapa de paradas y no introduciendo un código de parada como las aplicaciones anteriormente comentadas. Uno de los mayores problemas de las aplicaciones que ofrecen soporte para distintas ciudades del mundo, es la dificultad de mantener la información relativa a la red de paradas de cada ciudad y de estar al tanto de los posibles cambios en las peticiones a los servicios web que cada proveedor pueda modificar.

Ilustración 7. Captura de pantalla Transport Stops

Smart Transit

De igual manera que *Transport Stops*, es una aplicación no dedicada a la ciudad de Valencia y cuenta con información sobre transporte público de diversas ciudades. Esta aplicación en concreto ofrece no sólo información acerca de los autobuses, sino también de metro y tranvía.

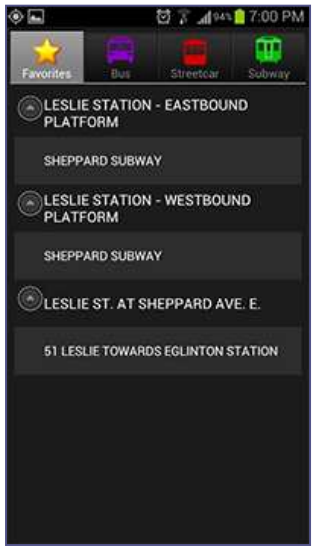

Ilustración 8. Captura de pantalla Smart Transit

2.3 Estudio de Mercado: Análisis cuantitativo

Continuando con el estudio previo de las aplicaciones existentes actualmente en Google Play, la siguiente tabla muestra un resumen de los aspectos más relevantes en cuanto a frecuencia de actualización y lanzamiento de nuevas versiones por parte del desarrollador o desarrolladores, versiones distintas de dispositivos Android que admiten la instalación o valoración por parte de los usuarios de las aplicaciones anteriormente comentadas.

Aplicación	Actualización	Versiones soportadas	Instalaciones	Valoración media
EMT Valencia	Media (Nov. 2014)	2.1+	De 100.000 a 500.000	4,2 / 5
eBus	Baja (Mar. 2012)	1.5+	De 10.000 a 50.000	4,1 / 5
ValenBus	Alta (Ene. 2015)	2.3+	De 10.000 a 50.000	4,4 / 5
Urban Step	Muy baja (Mar. 2011)	1.5+	De 10.000 a 50.000	4,0 / 5
Buses Valencia	Baja (Abr. 2012)	2.1+	De 500 a 1.000	3,2 / 5
Valencia Bus	Baja (Mar. 2013)	1.6+	De 5000 a 10.000	3,9 / 5
Transport Stops	Media (Nov. 2014)	2.3.3+	De 1.000 a 5.000	3,8 / 5
Smart Transit	Baja (Ago. 2013)	1.5+	De 100.000 a 500.000	2,9 / 5

Tabla 1. Análisis cuantitativo de aplicaciones similares (Abril 2015)

2.4 Estudio de Mercado: Análisis cualitativo

De igual manera, se han analizado las funcionalidades y facilidades de estas aplicaciones tratando de suplir en este nuevo desarrollo todas esas carencias que en algunas ocasiones suelen ser importantes, como la posibilidad de añadir un widget que ofrece información de un vistazo en el escritorio del dispositivo y sin necesidad de tener que abrir ninguna aplicación o la detección de ubicación del usuario sin necesidad de su intervención.

La siguiente tabla muestra una relación de características y funcionalidades de aplicaciones similares para la red EMT de Valencia ya existentes en Google Play.

Aplicación	Widget	Sin intervención	Funcionalidad	Avisos / incidencias	Idiomas	Dedicada
EMT Valencia	No	No	9	No	Sí	Sí
eBus	No	No	4	No	No	Sí
ValenBus	No	No	6	No	No	Sí
Buses Valencia	No	No	1	No	No	Sí
Urban Step	No	No	4	No	Sí	Sí
Valencia Bus	No	No	1	No	No	Sí
Transport Stops	No	No	3	No	No	No
Smart Transit	No	No	3	No	No	No

Tabla 2. Análisis cualitativo de aplicaciones similares (Abril 2015)

2.5 Fragmentación del Mercado

El sistema operativo Android para dispositivos móviles cuenta con un número considerablemente alto de versiones distintas. Desde que se lanzara con la versión 1.0 en el año 2007, cada versión de Android ha ido incorporando nuevas funcionalidades, modificando la forma en que se tratan algunas o directamente eliminando otras. Esto que a primera vista tan sólo parece una ventaja al crear con cada versión nuevas posibilidades de desarrollo y de servicios desde el punto de vista del usuario, desde el punto de vista del programador plantea un problema a la hora de tener en cuenta qué funcionalidades de las que está dotando a su aplicación no serán accesibles en terminales antiguos.

Versión	Nombre	API	Distribución
2.2	Froyo	8	0.4%
2.3.3 a 2.3.7	Gingerbread	10	6.4%
4.0.3 a 4.0.4	Ice Cream Sandwich	15	5.7%
4.1.x	Jelly Bean	16	16.5%
4.2.x		17	18.6%
4.3		18	5.6%
4.4	KitKat	19	41.4%
5.0	Lollipop	21	5.0%
5.1		22	0.4%

Tabla 3. Fragmentación Android (Abril 2015)

De aquí la importancia de realizar un estudio previo teniendo clara la idea de qué es lo que se desea que haga la aplicación y teniendo en cuenta cuáles serán los dispositivos móviles con sistema operativo Android para los cuales será posible dicha implementación. Cuanto más amplio sea este rango, más usuarios tendrán la posibilidad de instalar y utilizar la aplicación; pero como no todas las versiones Android cuentan con las mismas funcionalidades, incluir a versiones antiguas significará eliminar algunas funcionalidades o realizarlas de una manera más simple o menos potente en aquellos casos en los que al menos esto sea posible. Será por tanto necesario también, evaluar si se le otorga más prioridad al hecho de poder alcanzar un número mayor de potenciales usuarios rebajando a cambio las posibilidades de la aplicación.

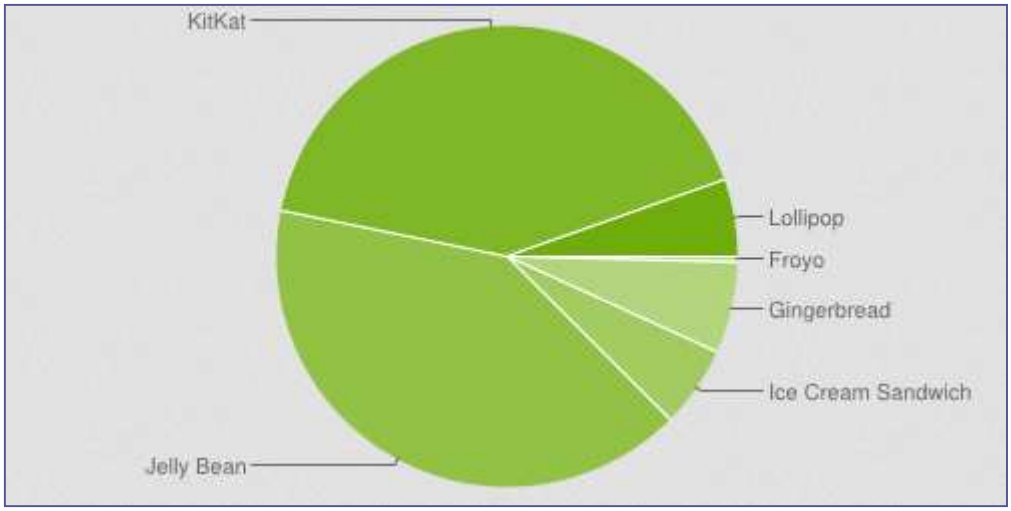

Ilustración 9. Fragmentación Android (Abril 2015)

Para el presente proyecto se decidió dar cobertura a todos los dispositivos listados por Google, esto es, soportar todas las versiones de las que actualmente Google tiene notificación que están siendo utilizadas por los usuarios, ya que las principales características de la aplicación (ubicación geolocalizada, peticiones a servicios-web…) están disponibles desde los inicios de la plataforma.

No obstante, algunas características como la forma en que se presentan los menús o las llamadas a los procedimientos que determinan si el dispositivo se encuentra bloqueado o en uso, es necesario realizarlas de maneras alternativas, introduciendo excepciones en la programación, basadas en la versión de dispositivo que el usuario estuviese utilizando en ese momento.

2.6 Conclusiones

Teniendo en cuenta las carencias o funciones nunca implementadas en aplicaciones similares a partir de los datos anteriormente analizados, se decidió basar el proyecto en el

desarrollo de un widget, algo que inexplicablemente no existía todavía en aplicaciones de consulta de llegadas de autobuses EMT para la ciudad de Valencia.

Esta aplicación basada en widget, debe permitir de forma rápida y de un vistazo, la consulta de las próximas llegadas de autobuses en el dispositivo móvil del usuario de manera similar a como el usuario lo haría; simplemente mirando en los paneles luminosos situados en las marquesinas de la red. Las posibilidades gráficas de Android permiten un diseño visualmente agradable y similar a dichos paneles reales, con una tipografía de matriz de puntos y colores idénticos al panel electrónico.

Un widget Android permite, además, ser arrastrado y colocado en cualquier parte del escritorio del dispositivo móvil Android, sin necesidad de tenerlo como una aplicación que el usuario debe abrir (aunque el widget debe permitir pulsar sobre éste y abrir una pequeña aplicación a pantalla completa con información ampliada sobre las llegadas). Es posible, además, tener en ejecución simultánea varias instancias del widget, para colocar si el usuario lo desea, en distintas páginas de su dispositivo.

Otro aspecto a mejorar después del estudio de las diversas aplicaciones ya existentes, es el de la inmediatez que se desea en la consulta de las llegadas. Este aspecto, mejorado significativamente por la naturaleza del diseño principalmente como widget y no como aplicación de pantalla completa, se verá mejorado por la no necesidad de intervención del usuario a la hora de comunicar el código o nombre de parada del cual desea obtener las llegadas: el widget detectará automáticamente cualquier cambio en la ubicación del usuario actualizando de inmediato la parada en la que el usuario se encuentra actualmente y calculando en consecuencia las nuevas llegadas.

El usuario estará, por tanto, despreocupado de tener que abrir ninguna aplicación ni tener que introducir ningún dato ni seleccionar ninguna opción. Con el simple gesto de desbloquear su terminal (encender la pantalla) y mirar en su escritorio, obtendrá los tiempos de próximas llegadas de la parada en la que se encuentre además de cualquier incidencia que se produzca en cualquiera de las líneas vinculadas a su parada actual (mediante un rótulo deslizante similar al de los paneles electrónicos reales).

Otra mejora que desea llevarse a cabo, después de analizar otras aplicaciones similares, parte del hecho de que la mayoría de aplicaciones que utilizan la ubicación geolocalizada para determinar la posición actual del usuario (ya sean aplicaciones de transporte o no), suelen necesitar de la activación del receptor GPS cuando se desea acotar esta ubicación a una precisión máxima (de otro modo y dependiendo de nuestra ubicación, tendríamos un error de precisión de entre 100 y 500 metros) con el consiguiente aumento en la espera por parte del usuario (pues se suele necesitar la recepción de la señal de varios satélites) y en el consumo de batería. Se desea mejorar este aspecto de tal manera que, tan sólo con la utilización de los servicios de ubicación básicos (Wi-Fi y antenas de telefonía móvil), sin necesidad de la activación de GPS y con el diseño de un algoritmo eficiente y optimizado al máximo, sea viable la ubicación del usuario con un error tan mínimo que haga posible localizarlo en la parada en la que se encuentra con total exactitud.

No obstante, y pese a la característica principal de widget de este desarrollo, se desea implementar unas funciones mínimas que a modo de pequeña aplicación, permitan al usuario

obtener una información ampliada de las llegadas, así como cambiar el modo de consulta y permitirle una consulta manual (por código, nombre de parada o de calle) además de la gestión de favoritos para las paradas que más frecuentemente utilice.

Por otra parte y dado que un widget Android tiene una ejecución permanente (y no sólo cuando se abre o cierra como una aplicación), se desea minimizar el gasto de recursos y batería, haciéndolo no operativo en aquellos momentos en que el terminal del usuario esté bloqueado (pantalla apagada) y que además éste también tenga la posibilidad de apagar el widget manualmente (aún con la pantalla encendida y con el dispositivo móvil en uso) en aquellos momentos en que sepa que no va a necesitar utilizarlo.

Se desea además, como se comentó anteriormente, que el widget/aplicación sea instalable en la mayoría de dispositivos posible, dando compatibilidad por tanto a la lista completa de versiones Android actualmente utilizadas (Abril 2015) además de al mayor número de resoluciones, densidades y tamaños de pantalla existentes (desde los terminales con pantalla pequeña prácticamente en desuso, hasta los tablets o los teléfonos con mayores resoluciones).

Por último, se considera interesante que sea traducida no sólo al idioma Valenciano (tratándose de una aplicación diseñada para un uso en la ciudad de Valencia), sino a otros idiomas como son el inglés, siendo accesible así también por turistas extranjeros.

3. Especificación de requisitos

3.1 Introducción

Este apartado describe la especificación de requisitos software (ERS) del widget a desarrollar. Se sigue la estructura definida según el estándar IEEE 830-1998.

3.1.1 Propósito

El propósito del presente apartado es el de definir los requisitos funcionales y no funcionales del widget, estando dirigido al equipo de desarrollo y que debe servir como base para su implementación. El documento de especificación de requisitos sirve también como un documento de acuerdo con el cliente, de manera que siempre quede bien definido qué es lo que se desea que haga la aplicación o widget.

3.1.2 Ámbito del sistema

Con el desarrollo del presente proyecto, se desea que el usuario disponga de un widget, algo no existente hasta ahora para Android, con el que pueda consultar la llegada de los autobuses municipales de la ciudad de Valencia sin necesidad de abrir ninguna aplicación y con el simple gesto de mirar la pantalla (previo desbloqueo en caso necesario).

El usuario tampoco necesitará introducir ningún dato relativo a su localización o parada en la que se encuentra, dejando en manos del widget detectar los posibles cambios de posición y ubicación del usuario en una parada de la red de autobuses para consultar y mostrar a continuación las llegadas de las líneas pertenecientes a esta, todo de manera automática.

Adicionalmente y pulsando sobre el panel de llegadas (widget), se abrirá una pequeña aplicación con información ampliada acerca de las llegadas y algunas opciones adicionales como la búsqueda manual o la gestión de favoritos.

3.1.3 Acrónimos

Nombre	Descripción
ERS	Especificación de Requisitos Software
RFxx	Requisito Funcional nº xx
RNFxx	Requisito No-Funcional nº xx

Tabla 4. Acrónimos

3.1.4 Definiciones

Nombre	Descripción
Widget	Porción de aplicación cuya interfaz gráfica es visible desde el escritorio del terminal sin necesidad de abrirla (está siempre en ejecución). Puede ser arrastrada y colocada en cualquier lugar del escritorio junto a otros widgets en una distribución y configuración flexible para el usuario.
Parada / paradas	Cada uno de los postes o marquesinas de la red de autobuses destinados a acoger a los pasajeros que están esperando el autobús. La mayoría de las marquesinas disponen de un panel electrónico informativo con los horarios de llegada de los autobuses, aunque no todas. Los postes, en general, no disponen de pantalla electrónica, salvo algunos pocos generalmente en el centro de la ciudad de Valencia. No siempre los paneles electrónicos se encuentran operativos.
Llegada	Para este proyecto, se considera una llegada cuando el autobús se detiene en una de las paradas de la red para recoger y/o dejar pasajeros, únicos lugares donde tiene autorizado realizar estas acciones. Las llegadas tienen asociado un tiempo de llegada estimado, que es el que muestran los paneles electrónicos en las paradas, siempre que estos se encuentren presentes y en servicio, así como en el widget/aplicación.
API	Contiene una serie de rutinas, funciones y servicios que son ofrecidas por un sistema operativo, en forma de librerías o en este caso por una entidad externa, para facilitar la construcción de otras aplicaciones.

Servicios web	Provee de un interfaz web que permite la comunicación e intercambio de información entre máquinas dentro de una red. Se puede decir que es un tipo de API que cuya lista de funciones o servicios pueden ser consultadas desde un software externo (programas, aplicaciones remotas…). Aunque la información que esta entidad externa desee consultar se encuentre en bases de datos, servidores distribuidos, con formas de acceso complejas… es transparente para ella, ya que es a la API (a través del servicio web) a la que solicita los servicios y ésta la que se encarga de obtener la información y entregarla en un formato sencillo y concreto.
Servidores EMT	Red de ordenadores que obtienen, almacenan y difunden una serie de datos de su flota de autobuses (posición, conductor, averias…). Algunos de esos datos, como la posición del autobús, son utilizados y procesados para determinan los tiempos de llegada en cada parada y puestos a disposición pública a través de servicios web.
Servicios de ubicación de Google	Servicio ofrecido por Google con el que es posible conocer con precisión la localización del usuario (país, localidad, calle…) conociendo sus coordenadas geográficas (latitud, longitud). Estas coordenadas son enviadas automáticamente por el terminal a través de Internet y la ubicación es recibida de vuelta. La información de las coordenadas geográficas no sólo pueden ser obtenidas del GPS del terminal; Google dispone de un sistema de geolocalización propio basado en el cálculo de posición por triangulación, utilizando la localización de las antenas de telefonía y localización de los router que se encuentran alrededor del usuario (mediante las direcciones MAC que son difundidas por éstos).
XML	Es un lenguaje que organiza la información por medio de etiquetas, facilitando así el acceso o búsqueda de un valor en concreto.
JSON	Puede ser utilizado como una alternativa a XML, aunque es más utilizado para transmitir información del tipo atributo-valor
Densidad de pantalla	La densidad de pantalla o densidad de píxeles, es el resultado de calcular la cantidad de píxeles alojados por unidad de medida (centímetros, pulgadas) en una pantalla, generalmente de dispositivos móviles y normalmente es medida en píxeles por pulgada (ppp o dpi en inglés). Aumentar la densidad de píxeles por pulgada permite que, al aumentar el tamaño de pantalla de un dispositivo móvil, no perdamos definición al crecer también el tamaño de píxel (y por tanto hacerlo “más visible”)
MAC	Identificador de 48 bits que corresponde de manera única a un dispositivo de red (tarjetas Wi-Fi, routers…)
Watchdog	En español “perro guardián”, es un mecanismo que generalmente mediante un contador, impide que un sistema se bloquee o termine encerrado en un bucle sin control. Mientras el sistema se encuentre trabajando normalmente, se controla que dicho contador nunca alcance su límite. Sólo en el caso de que el sistema haya perdido el control, el contador llegará a su límite y se llevará a cabo una desactivación controlada de todos sus componentes o del sistema completo.

Servicio Android	Un servicio es un componente que es arrancado y trabaja en un segundo plano sin intervención del usuario (ni le provee a éste de ninguna interfaz ni modo de interactuar). Los servicios son procesos invisibles que están diseñados para llevar a cabo operaciones a largo plazo (por ejemplo para reproducir música o recuperar datos de servidores remotos), todo esto ejecutándose de fondo, mientras el usuario puede seguir utilizando su dispositivo. Las aplicaciones con las que el usuario sí que puede estar interactuando en ese momento, pueden solicitar datos e intercambiar información con estos servicios que están corriendo todo el rato en segundo plano.

Tabla 5. Definiciones

3.1.5 Referencias

IEEE 830-1998: Recommended Practice for Software Requirements Specifications.

3.1.6 Visión general de la Especificación de Requisitos

La Especificación de Requisitos consta de tres secciones:

- La primera sección contiene la introducción al apartado de especificación de requisitos (sección actual).
- La segunda sección describe todos los factores que afectan al producto y a sus requisitos. Aquí no se describen los requisitos, sino su contexto. Proporciona una visión general del sistema.
- La tercera y última sección describe los requisitos a un nivel de detalle mayor, suficiente para que los desarrolladores puedan llevar a cabo un sistema que cumpla lo deseado. Además, es de utilidad para el equipo de pruebas, pues les permite realizar las pruebas necesarias que confirmen el buen funcionamiento y que determinen si el sistema satisface los requisitos.

3.2 Descripción general

En este apartado se lleva a cabo una descripción de alto nivel del sistema y se analizan todos aquellos factores que puedan afectar al producto y a los requisitos.

3.2.1 Perspectiva del producto

Mi Bus Valencia es un producto independiente, si bien necesita comunicarse con los servicios web ofrecidos por el Ayuntamiento de Valencia a fin de obtener la información de llegadas que la empresa municipal de transportes EMT facilita a los servidores del ayuntamiento.

El widget utiliza distintos servicios remotos para obtener la información y ubicación de paradas, llegadas de los próximos autobuses de una parada en concreto o la lista completa de paradas de la red de autobuses. Mediante una serie de peticiones en función de la ubicación o solicitud del usuario, obtiene los datos necesarios de estos servidores remotos para presentarlos en pantalla.

Ilustración 10. Ilustración básica de comunicación widget-servidores

3.2.2 Funciones del producto

El producto debe proporcionar a los usuarios que lo utilicen, una información precisa acerca de las llegadas de la parada en la que se encuentra, o la parada que el usuario haya solicitado en caso de que éste desee una búsqueda manual o por favoritos.

El usuario tiene dos vistas para las consultas: la del widget propiamente dicho, que ofrece las llegadas en el mismo formato y manera en que lo hacen los paneles electrónicos reales de la EMT o pulsando sobre el widget, a través de una pequeña aplicación que ofrece mayor información de llegadas en forma de listado.

Además, el usuario tiene la posibilidad de realizar una búsqueda de una parada en concreto o configurar las que desee como favoritas para ser utilizadas más tarde. En cualquier momento puede volver a la búsqueda automática por ubicación y cambiar la vista entre widget/aplicación.

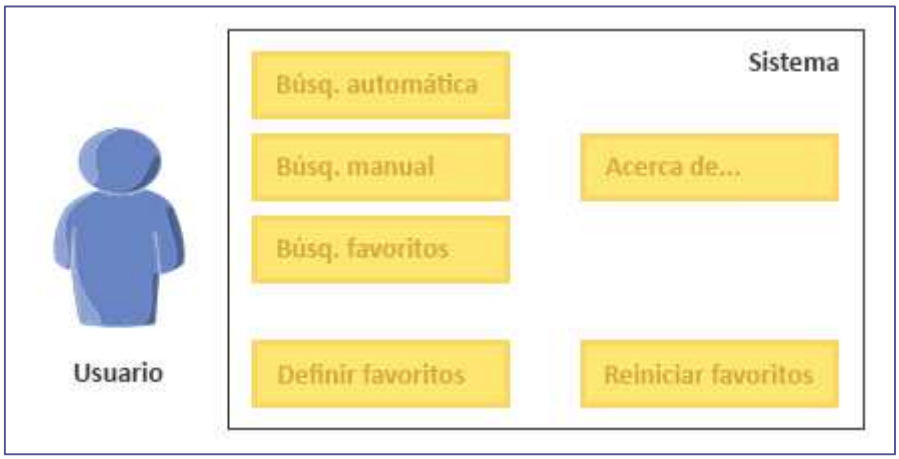

Ilustración 11. Diagrama de casos de uso

3.2.3 Características de los usuarios

Cualquier usuario debe ser capaz de utilizar el widget o aplicación, no habiendo un perfil concreto de usuarios que utilicen el transporte público, ni basado en la edad ni de cualquier otro tipo. Es por ello que deber ser fácil de usar, con unos menús y opciones intuitivos y fáciles de encontrar y entender, con una vista y gráficos agradables y debe proveer a los usuarios de una información concisa, rápida y precisa.

3.2.4 Restricciones

- Requiere ser instalado en un dispositivo Android con una versión igual o superior a la 2.2 (Android "Froyo", dispositivos del año 2010).

- Requiere que el dispositivo donde se ejecute, cuente con conexión a Internet.

- Aunque la conexión a Internet sea a través del proveedor de telefonía del usuario (conexión de datos de la operadora móvil), se requiere que la conectividad Wi-Fi esté siempre activada para el uso de los servicios de ubicación de Google.

- Requiere que los servicios de ubicación en general, estén activados en el dispositivo móvil.

3.2.5 Suposiciones y dependencias

- Se asume el correcto funcionamiento del sistema de localización e información de la red de autobuses que la empresa EMT tiene implementado. Se cuenta asimismo, con que la información transferida a los servidores del Ayuntamiento de Valencia esté disponible y accesible a través de los servicios web.

- Se espera que cualquier cambio en la definición de los servicios, llamadas o parámetros necesarios en la API del Ayuntamiento de Valencia, sea notificado a fin de que los desarrolladores del producto puedan modificarlo y adaptarlo a cualquier nuevo requerimiento de la API.

3.3 Requisitos específicos

En esta sección se detallan los requisitos funcionales y no funcionales con un nivel de detalle lo suficientemente específico para que un equipo de desarrollo pueda llevar a cabo todas las necesidades solicitadas por el cliente.

3.3.1 Requisitos de interfaz externo

Interfaz de usuario

Una aplicación o un widget para la consulta de las llegadas de los autobuses, debe estar destinado a cualquier público, por lo que el manejo y el uso de los menús dentro del programa, tiene que ser sencillo e intuitivo.

El widget debe ser sencillo de leer y atractivo visualmente, se desea que éste imite el aspecto de un panel electrónico de llegadas real. De los diversos modelos de panel electrónico con que cuenta la empresa EMT, se diseñará para el widget uno de tres líneas, el cual vaya mostrando las llegadas de la parada actual de forma rotativa (manteniendo en pantalla las tres primeras llegadas y cambiando cada 5 segundos para mostrar las tres siguientes).

Al pulsar sobre el widget se debe abrir una pequeña aplicación, la cual contará con una barra superior donde se muestre la parada y el modo de búsqueda actual, una barra inferior que permita añadir a Favoritos la parada actual y un listado que ocupe la parte central de la pantalla, con la información de todas las llegadas de la parada actual (y no sólo las que caben en el panel de tres líneas cuando la vista es la del widget).

La vista de aplicación también contará con un menú a través del cual sea posible: cambiar a modo de búsqueda manual (buscando la parada de la cual se desea obtener las llegadas, ya sea por código de parada, nombre de parada o nombre de la calle donde se encuentra ésta), cambiar al modo de búsqueda automático por ubicación (es el modo por defecto), seleccionar como parada actual cualquiera de las paradas existentes en Favoritos, borrar todos los favoritos y consultar el "Acerca de" de la aplicación.

Interfaz hardware

Si bien la operación con el widget o aplicación se lleva a cabo a través de la pantalla táctil del dispositivo, algunos dispositivos Android que incluyan teclado físico podrán utilizar este para la introducción de cualquier texto (por ejemplo a la hora de hacer una búsqueda manual de la parada de autobús).

Interfaz software

Los dispositivos Android que no poseen teclado físico, desplegarán en pantalla un teclado virtual (en aquellos casos en que se requiera) con el que es posible interactuar con la aplicación de igual manera que si fuese un teclado físico.

Interfaz de comunicaciones

La comunicación se llevará a cabo mediante peticiones HTTP a los servicios web que el Ayuntamiento de Valencia dispone para la consulta de llegadas de la red de transportes.

Se requiere de distintos servicios y parámetros asociados para obtener: la lista de paradas que coinciden con un criterio de búsqueda concreto (ya sea por código de parada, nombre de parada o nombre de la calle donde está ubicada la parada), la lista de paradas que se encuentran en un cuadrante geográfico determinado (latitud/longitud), los datos de una parada en concreto (su código, nombre, ubicación…), las llegadas de una parada en concreto.

La respuesta recibida por parte del widget o aplicación tras una petición determinada, puede ser en forma de documento XML o documento JSON, dependiendo del servicio solicitado.

3.3.2 Requisitos funcionales

Identificador	RF01
Nombre	Búsqueda automática
Descripción	El usuario podrá recibir las llegadas de la parada en la que se encuentre sin necesidad de introducirla (ubicación geolocalizada)
Entradas	Ninguna.
Salidas	Lista de llegadas de la parada actual del usuario.

Identificador	RF02
Nombre	Búsqueda manual
Descripción	El usuario deberá introducir un código, nombre de parada o nombre de calle en la que se encuentra, para poder recibir la lista de llegadas de esa parada en concreto.
Entradas	Código de parada, nombre de parada o nombre de la calle en la que se encuentra la parada.
Salidas	Lista de llegadas de la parada seleccionada.

Identificador	RF03
Nombre	Búsqueda por Favoritos
Descripción	El usuario podrá recibir las llegadas de alguna de las paradas que tenga guardadas en su lista de favoritos.
Entradas	Selección del Favorito (el cual tiene asociado un código de parada)
Salidas	Lista de llegadas de la parada seleccionada desde la lista de favoritos.

Identificador	RF04
Nombre	Búsqueda automática en Favoritos
Descripción	El usuario podrá recibir las llegadas de la parada más cercana (ubicación geolocalizada) de entre las existentes en su lista de favoritos.
Entradas	Selección del Favorito (el cual tiene asociado un código de parada)
Salidas	Lista de llegadas de la parada de favoritos más cercana.

Identificador	RF05
Nombre	Añadir a Favoritos
Descripción	El usuario podrá añadir la parada que actualmente se esté consultado, a su lista de favoritos.
Entradas	Ninguna
Salidas	Mensaje informando que la parada ha sido añadida a favoritos.

Identificador	RF06
Nombre	Reiniciar favoritos
Descripción	El usuario podrá reiniciar (borrar) la lista de favoritos.
Entradas	Ninguna
Salidas	Mensaje informando que la lista de favoritos ha sido reiniciada.

Identificador	RF07
Nombre	Mostrar información de la aplicación (about)
Descripción	El usuario podrá consultar información de la aplicación: equipo de desarrollo, versión actual, fecha de compilación. Podrá acceder a la ficha de la aplicación en Google Play para votar, comentar o recibir soporte.
Entradas	Ninguna
Salidas	Ventana “Acerca de” de la aplicación

3.3.3 Requisitos no funcionales

Identificador	RNF01
Tipo	Robustez y fiabilidad
Nombre	Tratamiento de excepciones
Descripción	Se deben tener en cuenta todas las posibles excepciones.

3.4 Conclusiones

En este capítulo se han definido unas directrices dirigidas al equipo encargado del desarrollo del producto. Se han establecido requisitos específicos, características y funciones del producto, cualquier tipo de suposición o restricción que pudiera existir así como la relación de los posibles usuarios y la forma que estos deben interactuar con la aplicación.

A continuación, se dará una definición más concreta en cuanto al diseño del sistema, entrando en la parte de su arquitectura, su comunicación con el exterior, el comportamiento de sus funcionalidades y aportando una serie de ideas en cuanto al aspecto visual de la interfaz para el usuario.

4. Diseño del sistema

4.1 Introducción

Este apartado contiene una descripción del diseño del producto. Se trata el diseño conceptual (arquitectura del sistema) y el diseño formal, especificando el comportamiento deseado, los diagramas de flujo, además de los prototipos de las pantallas del widget/aplicación.

4.2 Diseño conceptual del sistema

4.2.1 Arquitectura

El widget/aplicación están concebidos con una arquitectura cliente-servidor. El Ayuntamiento de Valencia es quien aloja las bases de datos y los servicios web. La empresa EMT de autobuses municipales provee a estas bases de datos de la información relativa a la posición GPS de cada uno de los autobuses de la flota. El widget/aplicación, a través de las funciones ofrecidas en la API del Ayuntamiento, realiza las llamadas necesarias a los servicios web para obtener la información deseada y presentarla en pantalla.

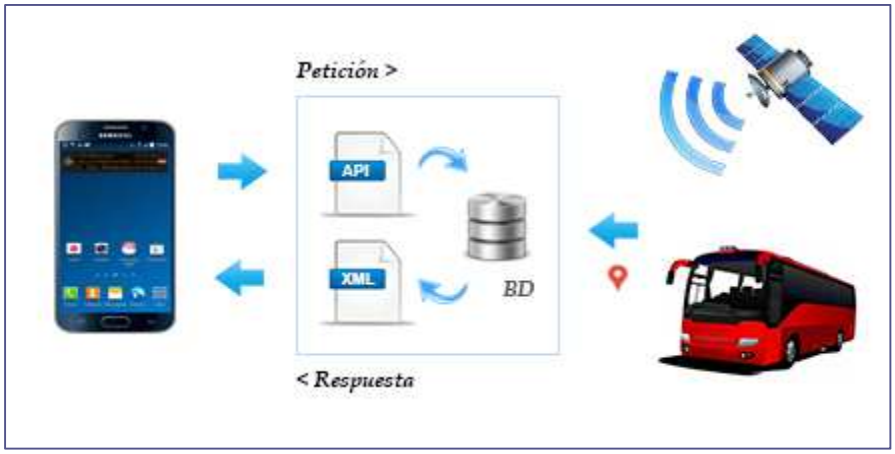

Ilustración 12. Arquitectura del Sistema

4.3 Diseño formal del sistema

4.3.1 Diseño de interfaces

Vista Widget

Es la vista por defecto. Una vez instalada la aplicación, el usuario podrá acceder a su cajón de aplicaciones y widgets instalados en el sistema y seleccionar el widget de *Mi Bus Valencia*. Una vez el widget sea colocado en cualquier lugar del escritorio de su dispositivo, éste deberá mostrar los horarios de llegada de la parada que se encuentre más cercana al usuario. No será necesario que el usuario abra ninguna aplicación para obtener de un vistazo los horarios de llegada, ya que el widget aparecerá con tan sólo desbloquear el dispositivo (encender la pantalla) en su escritorio junto a otros elementos e iconos y siempre actualizado.

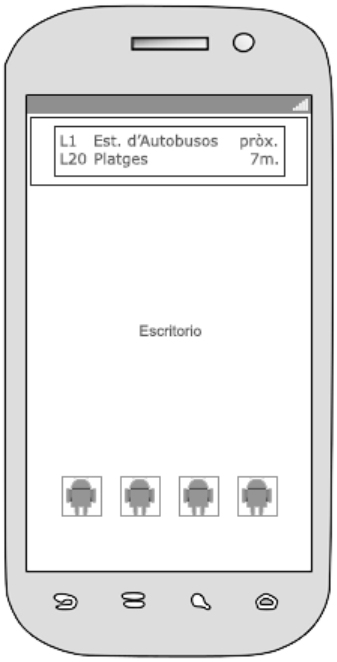

Ilustración 13. Widget (escritorio del terminal)

Vista Aplicación

Aunque el usuario no tiene la necesidad de hacer uso de ninguna aplicación, tendrá la posibilidad de obtener más información tocando con el dedo una sola vez sobre el widget, acción que abrirá una aplicación a pantalla completa. Esta aplicación contendrá mayor información sobre los horarios de paso de los próximos autobuses para todas las líneas que pasan por la parada seleccionada. Ha de tenerse en cuenta que el widget simula un panel electrónico de llegadas, por lo que sólo dispondrá de unas pocas líneas para mostrar los datos, lo cual se considera suficiente para mostrar la información que el usuario desea (la llegada del próximo autobús). Esta información visible en el widget debe ir rotando o pasando de página para mostrar las siguientes llegadas. Sin embargo con la vista de aplicación, el usuario será capaz de ver todas las llegadas de todas las líneas de autobús asociadas a la parada.

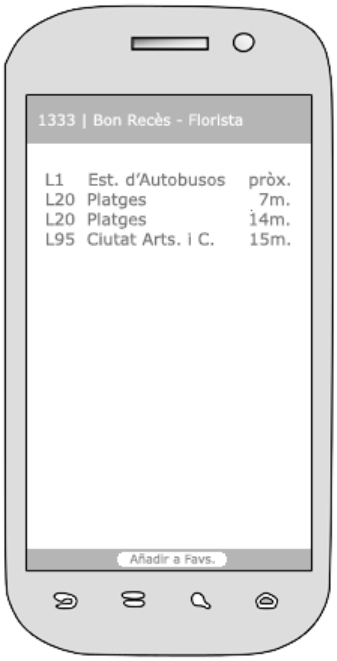

Ilustración 14. Aplicación asociada al widget

Menú de la aplicación

Además de mostrar mayor información sobre los horarios de llegada de las líneas asociadas a la parada actual del usuario, desde la vista de aplicación será posible definir paradas favoritas para acceder de un modo más rápido a ellas y cambiar el modo de búsqueda entre automático (auto-detección de la parada actual del usuario) y manual (selección manual de una parada de la red de autobuses). Para ello se hará uso de un menú desplegable, el cual permitirá seleccionar el modo de búsqueda, además de cualquiera de los favoritos que el usuario haya definido.

Ilustración 15. Menú de la aplicación

Búsqueda manual (Paso 1)

La pantalla de búsqueda manual, debe permitir al usuario seleccionar cualquier parada de la red de autobuses para mostrar las llegadas de una parada que no sea la más cercana al usuario. De esta manera puede obtener información de una parada que va a utilizar en breve (pero que no es su parada más cercana), con el objetivo, por ejemplo, de planificar su salida de casa o lugar de trabajo. Para seleccionar una parada de manera manual, el usuario deberá tan sólo introducir el código numérico de parada, el nombre de la parada o el nombre de la calle en la que se encuentra (no será necesario introducir la información completa siendo tan sólo necesarias unas cuantas letras o parte del nombre).

Ilustración 16. Pantalla de búsqueda manual (Paso 1)

Búsqueda manual (Paso 2)

Una vez el usuario haya introducido información en la caja de texto para llevar a cabo una búsqueda manual de la parada, una nueva pantalla deberá mostrarle todas aquellas coincidencias, para que el usuario elija a través de una lista la parada que desea establecer como parada actual en el sistema y comenzar a recibir información sobre las próximas llegadas de todos los autobuses asociados a dicha parada.

Ilustración 17. Pantalla de búsqueda manual (Paso 2)

4.3.2 Diagramas de flujo

Recuperar información de llegadas

La consulta para la recuperación de llegadas se lleva a cabo utilizando el código de parada. El código de parada forma parte de las variables del núcleo y debe estar siempre actualizado.

Cuando el usuario tiene activado el modo automático, el sistema es el que se ocupa de obtener la localización y mediante ésta, establecer qué código de parada tiene asignado la parada más cercana al usuario.

Si el usuario utiliza el modo manual de búsqueda de parada, el sistema buscará el código de parada una vez que el usuario haya elegido qué parada desea consultar.

La consulta de las próximas llegadas se realiza periódicamente a fin de tener una lista siempre actualizada.

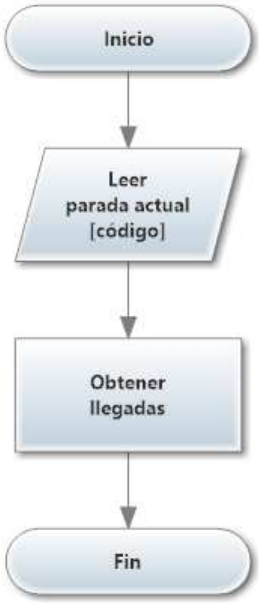

Ilustración 18. Diagrama de flujo: Recuperar información de llegadas

Establecer la parada más cercana

En el modo automático de búsqueda, el sistema comprueba periódicamente si la ubicación del usuario no ha cambiado. En caso de un cambio de ubicación (se establece un umbral mínimo para el disparo de este evento) se recalcula la parada más cercana al usuario.

El sistema contiene una lista de las paradas vigentes en la red de autobuses con su posición geográfica (latitud-longitud). En base a la ubicación recuperada de los servicios de ubicación de Google, se recalcula la parada más cercana al usuario.

Este código de la parada más cercana es utilizado para actualizar la variable del núcleo "código de parada" estableciendo así en el sistema la nueva parada más cercana al usuario, la cual se utiliza periódicamente para consultar la lista de próximas llegadas.

Ilustración 19. Diagrama de flujo: Obtener la parada más cercana

Obtener la ubicación actual

La ubicación del usuario es obtenida cada vez que se requiere encontrar la parada más cercana al usuario. Utiliza los servicios de ubicación de Google y establece un tiempo límite después del cual deja de realizar búsquedas.

Mientras tanto, el módulo de obtención de ubicación intenta encontrar una ubicación mejor a la actual (por precisión). Si encuentra una ubicación con una precisión menor a un umbral definido el cual se considera que es lo suficientemente bueno como para ubicar la parada del usuario sin ningún tipo de error, deja de intentar encontrar mejores ubicaciones.

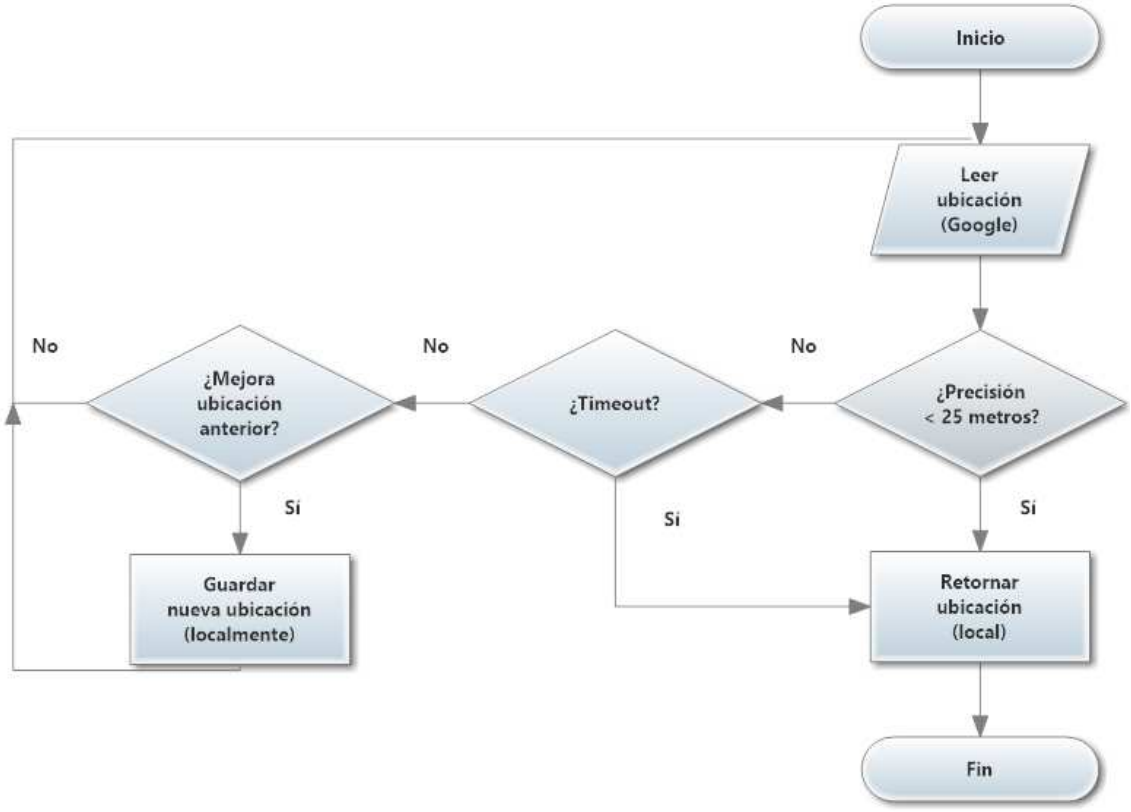

Ilustración 20. Diagrama de flujo: Obtener la ubicación actual

Cálculo de la parada más cercana

El cálculo de la parada más cercana a la ubicación actual del usuario guardada en el sistema, se lleva a cabo utilizando las variables latitud y longitud de dicha ubicación y las mismas variables que determinan la posición de cada parada en la lista de paradas de la red guardadas en el sistema (o en la lista de paradas favoritas si estamos utilizando un modo de búsqueda entre sólo los favoritos)

Por cada parada de la lista de paradas, se calcula la distancia con la ubicación del usuario. Si el último cálculo realizado (distancia entre la parada que se está calculando actualmente y el usuario) es mejor que el último obtenido, se actualiza la parada más cercana al usuario.

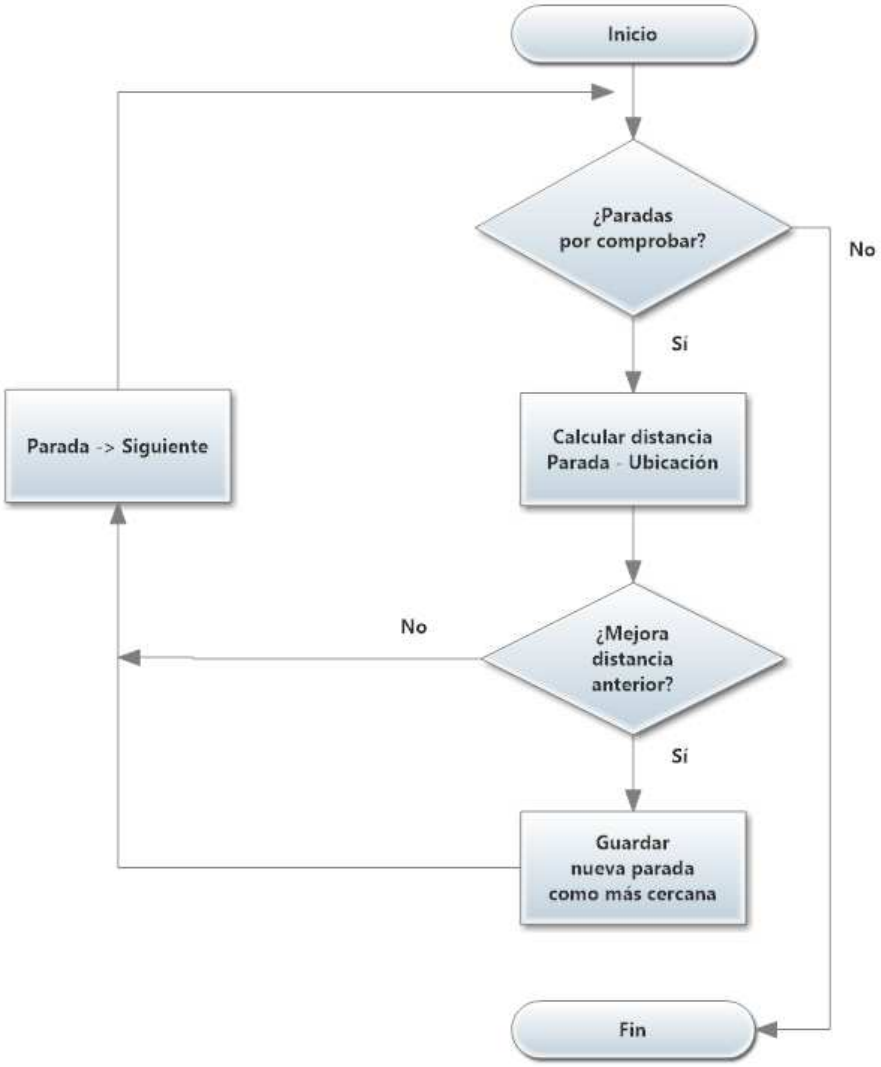

Ilustración 21. Diagrama de flujo: Cálculo de la parada más cercana

Selección del modo de búsqueda

El usuario puede cambiar el modo de búsqueda en todo momento y elegir entre una búsqueda automática (obteniendo la lista de llegadas de la parada más cercana a la ubicación actual en la que se encuentre o más cercana a alguna de su lista de favoritos) o una búsqueda manual.

Si el modo de búsqueda elegido es un modo de búsqueda automático, se lanzará el proceso de recuperación de ubicación y búsqueda de parada más cercana. En caso de ser una búsqueda manual, se preguntará al usuario por la parada para la cual desea recibir la lista de próximas llegadas.

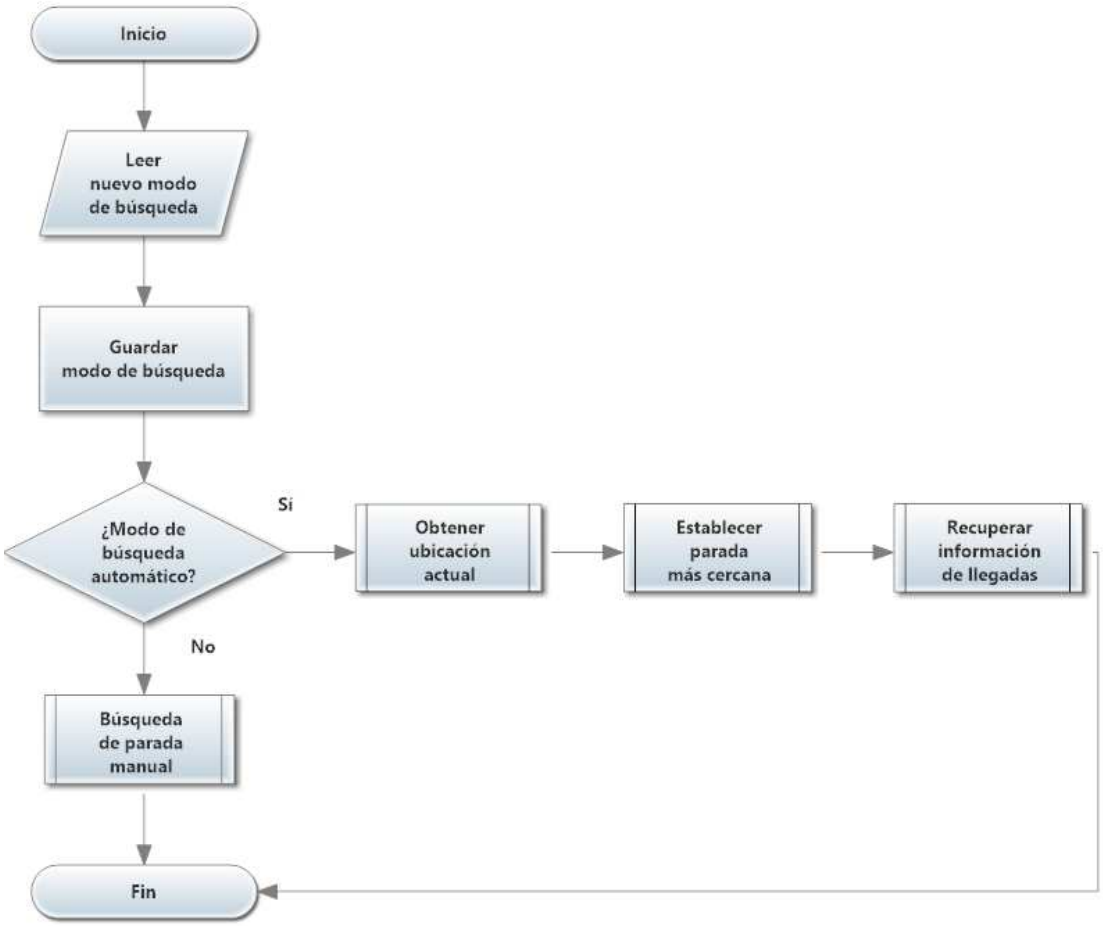

Ilustración 22. Diagrama de flujo: Selección del modo de búsqueda

Búsqueda de parada manual

El modo de búsqueda manual permite al usuario establecer qué parada es la actual en el sistema (en qué parada se encuentra ubicado o de qué parada desearía obtener una lista de llegadas). El usuario puede elegir dicha parada de su lista de favoritos o iniciar una búsqueda mediante patrón.

Puede utilizarse el código de parada directamente (que es la variable con la que el sistema funciona internamente), el cual es visible en las marquesinas y postes de las paradas de la red, pero también puede utilizar el nombre de parada o nombre de la calle donde se encuentra la parada (no es necesario introducir el nombre completo).

Posteriormente el sistema devolverá una lista de paradas coincidentes con el patrón de búsqueda donde el usuario puede elegir una de ellas (el sistema traducirá y almacenará la parada elegida como código de parada para trabajar con esta variable en todos los procesos).

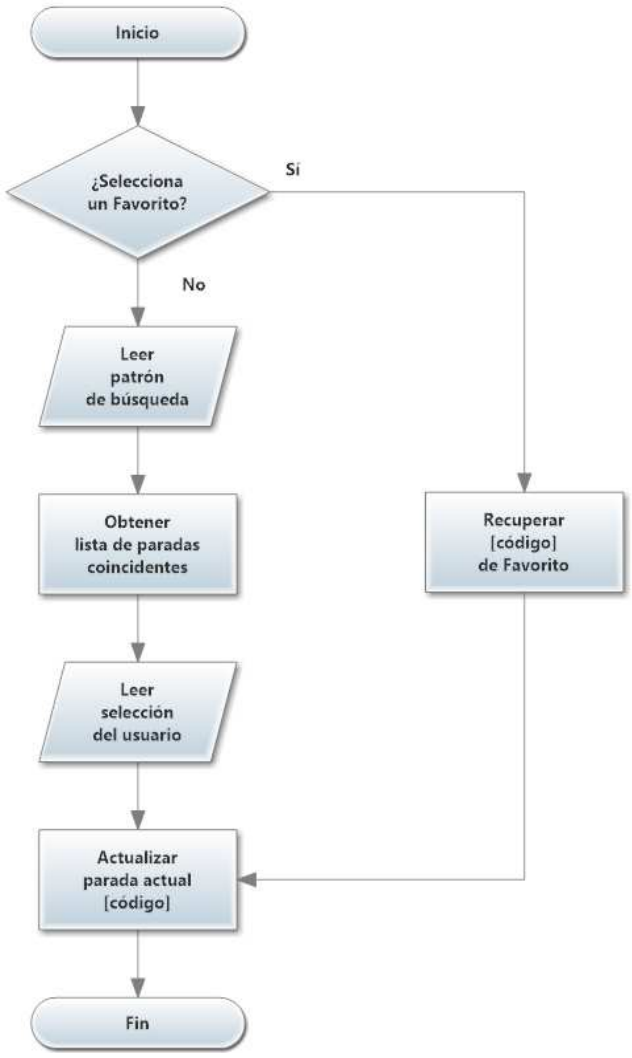

Ilustración 23. Diagrama de flujo: Búsqueda de parada manual

4.4 Conclusiones

Se ha aportado un diseño conceptual del sistema y se ha visto cual debe ser el comportamiento interno del sistema y también como éste debe comunicarse con el exterior para la obtención de los datos que requiere para procesar y presentar al usuario. Se han definido además, una serie de prototipos de la interfaz de la aplicación, dando una idea así de qué es lo que se desea y donde debe ir cada elemento de la aplicación.

El siguiente capítulo aporta detalles de la implementación con total detalle, quedando claro todos los aspectos que cierran la definición del producto final. Se comenta el entorno de desarrollo utilizado, herramientas de apoyo utilizadas, se detalla la aplicación asociada al widget y todos los módulos que interactúan con el núcleo o sistema principal.

5. Implementación

5.1 Introducción

En este apartado se pasará a explicar todos los detalles del proceso de implementación del producto: entorno utilizado, recursos, estructura del código y como se comunican y conectan los distintos módulos que por separado obtienen toda la información necesaria para que el sistema o módulo principal la procese y muestre los horarios de llegada del transporte público en la parte visible del usuario.

5.2 Entorno de desarrollo

Aunque en un principio se contempló la idea de utilizar el nuevo *Android Studio* (IDE diseñado por Google específicamente para el desarrollo de aplicaciones Android), tras algunas pruebas y viendo que ocasionaba algunos problemas con ciertos módulos, finalmente se optó por *Eclipse*, entorno más generalista utilizado hasta el momento para el desarrollo de aplicaciones Android y (hasta la fecha) algo más estable.

Todas las imágenes han sido creadas o tratadas con el software de diseño *Adobe Photoshop CS6*, con distintas resoluciones para tratar de adaptarlas a distintos tamaños de pantalla, con el fin de evitar que dispositivos con pantallas más grandes redimensionen imágenes pequeñas perdiendo así definición. Concretamente se han contemplado los diseños de pantalla Android: *low, medium, high, extra-high, extra-extra-high* y *extra-extra-extra-high*, que van desde una densidad de pantalla de 120 puntos por pulgada a una densidad para los dispositivos más grandes de 640 puntos por pulgada, más que suficiente para los dispositivos existentes a fecha de desarrollo de este proyecto.

5.3 Aplicación

La aplicación de este proyecto está basada en un widget. El widget es la parte visible para el usuario, el cual le muestra la información acerca de los destinos de línea y sus llegadas, siendo transparente para el usuario todo el proceso interno (que es la mayor parte del código) y que se lleva a cabo para el cálculo de su posición, sincronización con sus preferencias o estado anterior de la consulta y enlace de todos estos datos para dar el resultado final.

Además de la parte visible para el usuario (el widget), la parte núcleo del widget se ocupa de las llamadas a las distintas funciones y de reunir y sincronizar todos los datos para devolver un resultado final. Se puede decir que esta parte núcleo sería como la circuitería interna del panel electrónico físico que se encuentra en las marquesinas de la red de autobuses y la parte visible del widget (la interfaz de usuario) como la pantalla electrónica de matriz de puntos de los paneles electrónicos.

El widget tiene asociada una aplicación a pantalla completa. Esta aplicación se lanza al pulsar sobre la pantalla del panel de llegadas (el widget) y ofrece información ampliada que no cabría en el widget de tres líneas. La aplicación ofrece la lista completa de llegadas para ese día de la parada actual incluyendo las líneas nocturnas de las que aún pueden quedar horas para salir de cocheras. Es desde esta aplicación a pantalla completa, desde donde el usuario puede cambiar el modo de búsqueda, eligiendo manualmente una parada específica para consultar sus llegadas, establecer su lista de favoritos, seleccionar de sus favoritos una de las paradas para establecerla como parada actual, volver a la búsqueda automática, etc…

Finalmente, el núcleo del widget hace uso o se apoya en distintos módulos que ofrecen todas las funciones necesarias para llevar a cabo el cálculo de resultados: el módulo de localización, el módulo de conexiones http y el módulo de funciones.

Módulo de localización

Contiene las funciones necesarias para establecer la ubicación del usuario con una considerablemente alta precisión teniendo en cuenta que no es necesario la activación de GPS en el dispositivo. Utilizando el servicio de ubicación de Google, tan sólo es necesario conocer las antenas de telefonía móvil de nuestro entorno y direcciones MAC de los router a nuestro alrededor para establecer nuestra ubicación con un error entre los 20 y los 70 metros, suficiente para localizar sin error la parada actual del usuario.

La manera de conseguir este mínimo error de precisión sin necesitar que el usuario mantenga activado en su dispositivo el modo GPS ha sido a través de un sencillo pero eficiente algoritmo para tratar los valores devueltos por el servicio de ubicación de Google.

El algoritmo obtiene lecturas (latitud, longitud y precisión estimada) cada cierto tiempo durante un máximo de medio minuto. La precisión de cada lectura es comprobada con la anterior para intentar mejorar la ubicación. El algoritmo es cancelado nada más que se encuentra que una lectura tiene una precisión estimada menor a 25 metros, pues se considera que esta es más que suficiente para localizar una parada con total precisión y no se espera poder obtener una ubicación más precisa.

El módulo de localización es totalmente deshabilitado cada vez que se termina una búsqueda de ubicación, mejorando así considerablemente el consumo de batería. Asimismo, contiene un *watchdog* que impide que los servicios de ubicación de Google sigan habilitados en caso de que el usuario haya retirado o desinstalado el widget de su dispositivo móvil y esta acción haya coincidido con una petición (por tanto activación) de los servicios de localización.

Módulo de conexiones http

Mediante la llamada a las funciones de este módulo se obtiene toda la información necesaria de la API para la consulta de los horarios de llegada. Se han utilizado las clases *HttpClient* y *HttpGet* del paquete *org.apache.http*, sencillas y más que suficientes para la tarea requerida, ya que Ayuntamiento de Valencia provee de una serie de servicios web que sólo requieren de unos pocos parámetros pasados por GET.

Las peticiones son no bloqueantes, la interfaz de usuario y resto de procesos pueden seguir su trabajo sin esperar una respuesta de este módulo, lo cual obliga a llevar a cabo una consulta de disponibilidad de resultados de manera periódica. Una vez que el núcleo es informado de la disponibilidad de los resultados, recoge estos, desactiva las conexiones y sigue adelante.

El módulo de conexiones http ha sido extendido de *AsyncTask*, lo cual ha sido necesario para lograr el trabajo requerido de manera no bloqueante como se ha comentado anteriormente. *AsyncTask* proporciona el método *doInBackground* que es donde se realiza el trabajo de peticiones http, realizando todas estas tareas en un segundo plano y manteniendo libre así al hilo principal.

Las distintas peticiones existentes dentro de este módulo de entre las ofrecidas por la API de Ayuntamiento de Valencia son:

1. **Lista completa de paradas dentro de un cuadrante**

   Datos proporcionados: Esquinas superior-izquierda e inferior-derecha (ambas con latitud y longitud) de un cuadrante del que se desea que sean devueltas las paradas existentes dentro de éste.

   Devuelve todas las paradas de la red, incluyendo su código numérico de parada, nombre, calle en la que se encuentra y longitud-latitud de su ubicación.

2. **Lista de paradas que coinciden con un patrón de búsqueda**

   Datos proporcionados: Un patrón de búsqueda que puede incluir tan sólo unas pocas letras y puede referirse tanto al código de parada, nombre de parada o calle donde se encuentra la parada.

   Devuelve las paradas de la red que coinciden con un patrón de búsqueda. El resultado devuelto es un listado de las paradas que coinciden con dicho patrón donde se incluye su código numérico de parada, nombre, calle en la que se encuentra y longitud-latitud de su ubicación.

3. Lista de llegadas dado un código de parada

Datos proporcionados: El código numérico de la parada de la cual se desea obtener las llegadas.

Devuelve las llegadas de todos los autobuses que utilizan la parada proporcionada. La respuesta puede incluir los minutos de espera para el próximo autobús o la hora exacta a la que el autobús tiene previsto el paso por la parada.

Módulo de funciones

Contiene una serie de funciones de utilidad. Todas estas funciones se han incluido en un módulo común y separado, ya que realizan pequeñas funciones de apoyo al núcleo principal.

En éste módulo se encuentran funciones de extracción de los datos devueltos por la API, que suelen ser en formato XML o JSON, funciones de comprobación de conectividad Wi-Fi o datos móviles, funciones de reemplazo de caracteres…

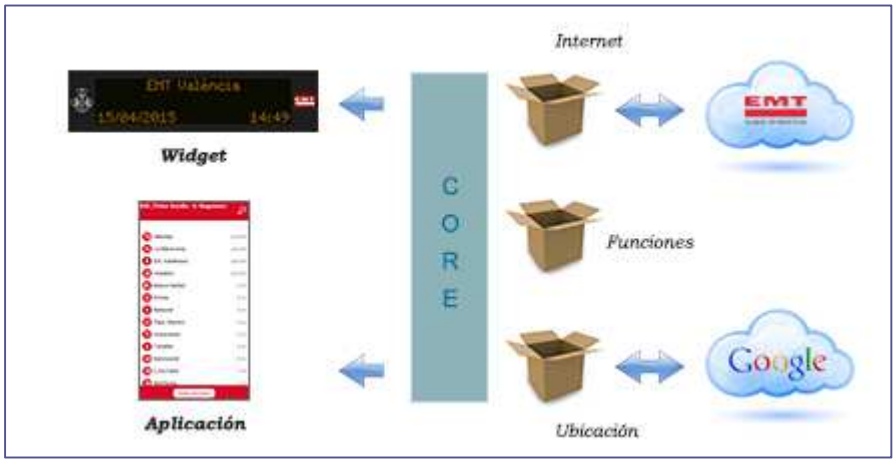

Ilustración 24. Estructura de la aplicación

El núcleo utiliza todos estos módulos de apoyo para llevar a cabo la obtención de resultados y presentación al usuario.

Cada minuto se lleva a cabo un proceso por el cual se obtiene la lista completa de próximas llegadas de todos los autobuses que tienen asociada la parada actual del sistema. Si la parada actual del sistema ha sido establecida de manera manual (búsqueda por patrón o selección a través de favorito), sólo es necesario utilizar el servicio web (módulo de Internet) para la obtención de la lista de llegadas utilizando el código numérico de parada.

Si la parada actual del sistema debe ser establecida de manera automática (y por tanto es cambiante según el desplazamiento del usuario), antes de obtener la lista de llegadas es necesario comprobar si la ubicación actual del usuario coincide con la última ubicación que teníamos de éste. El servicio de ubicación es lanzado y queda trabajando en segundo plano, el núcleo permanece a la escucha y sólo cuando obtiene una localización definitiva, recalcula la parada más cercana al usuario.

El cálculo de la parada más cercana al usuario se lleva a cabo restringiendo la búsqueda a tan sólo las paradas que se encuentran en un cuadrante alrededor de la ubicación recientemente obtenida. Para obtener toda la lista de paradas dentro de un cuadrante se hace uso de otro de los servicios web de la API, comentado anteriormente. Con esta lista reducida de paradas alrededor del usuario, se calcula con cada una de ellas la distancia al usuario, estableciendo como nueva parada del sistema la más cercana de todas ellas.

La utilización de la llamada al servicio web para la obtención de la lista de paradas existente dentro de un cuadrante, se descartó finalmente en la última versión del widget considerando que la lista de paradas de la red de autobuses no varía prácticamente, por lo que se optó finalmente por proveer al widget/aplicación de un fichero de datos con toda la lista de paradas de la red, ahorrando así unos 10 segundos cada vez que el núcleo tiene que recalcular la parada más cercana, al haberse evitar la llamada a este servicio web. A cambio, es necesario que periódicamente los desarrolladores revisen si las paradas de la red no han variado, para subir un fichero de datos de paradas de la red más actualizado con la publicación de cada nueva versión del widget/aplicación.

Se pretende por tanto que el núcleo conserve siempre una lista completa de parada actual y de llegadas actualizada cada minuto (este estado es además salvado en el dispositivo de manera persistente). La lista completa de llegadas es visible en el modo de aplicación (pulsado sobre el widget) mientras que el widget muestra una lista rotativa de llegadas (tres por cada página) de manera similar a como lo hace un panel de llegadas real.

Este diseño modular permite que sea muy sencillo realizar cualquier tipo de cambio sin afectar otras funcionalidades no relacionadas, especialmente si se deben llevar a cabo ajustes en el módulo de comunicaciones, en aquellos casos, por ejemplo, en que el Ayuntamiento de Valencia notifique cualquier tipo de modificación en las llamadas a la API, ampliación de servicios, etc…

5.4 Información persistente

El widget utiliza cierta información almacenada de manera persistente en el dispositivo.

Por una parte está la lista completa de paradas de la red de autobuses, los ficheros de configuración para ajustar algunos parámetros por parte del desarrollador, los ficheros de configuración de los servidores que contiene el formato necesario para las llamadas a los servicios web y los ficheros de idioma para dar soporte a otras lenguas con la traducción de todos los mensajes, etiquetas y menús.

Por otra parte es necesario almacenar cierta información que contiene las preferencias del usuario, la última parada utilizada, la última parada más cercana al usuario, la última posición del usuario o la lista de favoritos establecidos por el usuario entre otros datos usados por el widget/aplicación para su correcto funcionamiento. Para el almacenaje y gestión de este último tipo de información se ha utilizado *SharedPreferences*, que aunque tremendamente simple (y limitada) hace su uso sencillo, rápido, efectivo y más que suficiente para las necesidades.

5.5 Paso de mensajes

Cierta comunicación sólo es posible mediante el paso de mensajes. Tras algunas pruebas utilizando otros métodos para la comunicación entre los distintos módulos del sistema, finalmente para este caso se ha optado también por una solución muy sencilla debido a que no se requería una alta complejidad ni un orden concreto en la llegada de los mensajes, ni tampoco el tiempo de tratamiento de estos mensajes era algo crítico.

Basándonos en la misma clase *SharedPreferences* para el tratamiento de la información persistente, se ha llevado a cabo una adaptación para lograr que los distintos módulos intercambiaran información entre ellos de una manera que de otro modo no hubiese sido posible. Por ejemplo, la creación de una aplicación asociada al widget se lleva a cabo a través de un servicio Android, a través del cual ya no podemos recuperar una referencia a dicha aplicación, al menos desde la parte visible del widget. La solución de utilizar *SharedPreferences*, nos permite escribir información en el dispositivo y que ésta sea leída posteriormente desde cualquier otro módulo dentro del mismo entorno de ejecución.

Con esto hemos conseguido llevar a cabo un paso de mensajes para informar de una nueva lista de llegadas, una nueva selección del usuario y en general de cualquier cambio de estado en cualquiera de otras dos partes del sistema que de otro modo hubiesen quedado incomunicadas.

5.6 Servicio Android

Dado que el widget debe estar actualizando la ubicación del usuario y con ello determinando la parada actual para realizar seguidamente una lectura de las próximas llegadas sin intervención del usuario, todo el desarrollo ha sido basado en un servicio que continuamente está ejecutándose en segundo plano.

Este servicio arranca la primera vez que el usuario coloca el widget en el escritorio de su dispositivo y está en marcha únicamente si la pantalla se encuentra encendida. Cada vez que el usuario bloquea el dispositivo (apaga la pantalla) el servicio entra en un estado pausado. Esto permite reducir el consumo de batería, y puesto que el servicio una vez pausado no puede realizar peticiones de actualización, la ausencia de conexiones http contribuye a esta reducción de consumo tanto de batería como también de datos en este caso. Una vez que el usuario de nuevo desbloquea el dispositivo y entra en modo interactivo de nuevo (enciende la pantalla), de nuevo

arranca el servicio y vuelven a realizarse las peticiones necesarias para mantener el listado de llegadas actualizado a la parada actual del usuario.

El usuario puede tener varias instancias del widget en distintas páginas y áreas de su pantalla de escritorio (varios widgets). Cuando el usuario retira el último widget de su escritorio, el servicio es retirado completamente (detenido y eliminado del sistema).

5.7 Evaluación del sistema

Por último, se llevó a cabo una evaluación final del sistema completo para verificar que todas las funcionalidades se llevaban a cabo de forma correcta y sin errores. Se introdujeron todas las combinaciones posibles de situaciones y desplazamientos del usuario y todos los casos y excepciones con las que se podía encontrar, a fin de encontrar cualquier tipo de incompatibilidad entre los diversos datos recibidos y procesados, o cualquier problema, excepción o resultado no deseado.

Funcionalidad	Acción	Resultado
Respuesta general a distintas acciones sobre el sistema	Selección de opciones, menú, cambios de configuración…	Respuesta satisfactoria: adecuada, sin bloqueos. Comportamiento dentro de esperado.
Detección correcta de la ubicación actual del usuario	Prueba aislada del módulo de ubicación en distintos puntos de la ciudad.	Ubicación correcta con error medio de 45 m., suficiente para determinar la parada.
Determinación correcta de la parada más cercana al usuario	Prueba aislada del módulo de funciones para la extracción de la parada más cercana de la red EMT dada una ubicación.	Parada más cercana determinada de forma correcta el 100% de las pruebas realizadas en distintos puntos de la ciudad.
Obtención correcta de los horarios de llegada en base a la parada solicitada	Prueba aislada del módulo de Internet para la obtención de los horarios de llegada de la red EMT dado un código de parada.	Obtención correcta de los horarios de llegada o de los mensajes informativos de panel fuera de servicio, línea fuera de servicio o línea desviada.
Comportamiento correcto del sistema autónomo de obtención de los horarios de llegada	Prueba conjunta de los módulos de ubicación, parada más cercana y obtención de horarios de llegada en distintos puntos de la ciudad.	Obtención correcta sin ninguna intervención del usuario, de los horarios de llegada (o mensajes informativos) de diversas paradas de la red EMT en distintos puntos de la ciudad.
Ahorro de energía en modo de dispositivo bloqueado y recuperación satisfactoria de un estado de reposo.	Forzado del dispositivo a modo reposo (bloqueo) en distintos escenarios (con transacciones de datos, sin transacciones de datos, durante un proceso de ubicación, otros procesos…)	Recuperación satisfactoria reanudando las peticiones de datos o procesos de ubicación en su caso. Servicio pausado o reanudado correctamente.
Servicio Android	Añadir y retirar instancias del widget en una misma sesión o reiniciando el dispositivo.	Arranque y detención correcta de los servicios asociados.

Tabla 6. Evaluación del sistema

5.8 Capturas de pantalla

Ilustración 25. Captura de pantalla. Widget (I)

Ilustración 26. Captura de pantalla. Widget (II)

Ilustración 27. Captura de pantalla. Aplicación (llegadas)

Ilustración 28. Captura de pantalla. Aplicación (menú)

Ilustración 29. Captura de pantalla. Aplicación (búsqueda manual)

Ilustración 30. Detalle de widget

5.9 Conclusiones

En este capítulo se ha mostrado la fase de implementación con un detalle mayor: entorno de desarrollo, herramientas de diseño gráfico utilizadas, funcionalidades de la aplicación asociada al widget y como ambos interactúan. Se ha visto la estructura modular de la aplicación, cuáles son las funciones de cada uno de estos módulos y cómo finalmente se comunican con el núcleo principal para que éste reúna todos los datos obtenidos a fin de mostrar la información final al usuario. También se han descrito otros detalles asociados al producto, como son el paso de mensajes o los servicios Android asociados.

Por último se ha detallado cómo se han llevado a cabo todas las pruebas necesarias, aisladamente y en conjunto, para determinar el comportamiento correcto del producto en base a los requerimientos que se habían establecido inicialmente.

6. Conclusiones

6.1 Introducción

Este apartado cierra el presente documento exponiendo los problemas y dificultades encontrados durante el desarrollo e implementación del proyecto y las soluciones encontradas.

Se enumeran además, una serie de futuras ampliaciones que sería interesante implementar en el producto.

6.2 Problemas y soluciones

Una vez se tuvo suficiente código y módulos desarrollados como para comenzar a realizar las primeras pruebas, uno de los primeros problemas encontrados fue la inexactitud de la ubicación. Además, para minimizar el gasto de batería al máximo, la frecuencia de consulta de la ubicación había sido fijada entre 2 y 3 minutos, lo cual empeoraba todavía más la precisión de la ubicación.

Se mejoró el algoritmo de obtención de ubicación, no limitándose a recoger tan sólo una muestra, sino a llevar a cabo una serie de muestras periódicas durante un periodo de tiempo y guardando siempre la muestra de mayor precisión encontrada hasta el momento. El hecho de llevar a cabo una serie lecturas durante un corto periodo de tiempo (en lugar de una sola lectura cada cierto tiempo) redundó en una ligera mejora de la precisión sin incrementar el uso de batería. Además, implementando un sistema de "mejor estimación", se logró reducir la frecuencia de búsqueda a tan sólo 1 minuto, reduciendo algo más el consumo de la batería y logrando en ocasiones precisiones similares a las conseguidas con posicionamiento GPS y con la ventaja de no tener que activar éste, ganando en tiempo de adquisición de ubicación y consumo de la batería.

Aún con las mejoras comentadas, el consumo de la batería seguía considerándose alto, por lo que se introdujo una mejora en este sentido, desactivando cualquier conexión de datos y actualización del panel en caso de que la pantalla del usuario se encontrase apagada con el dispositivo bloqueado (o en modo de espera).

Por último, se decidió además que en este estado de reposo se pausase también el servicio asociado, deteniéndose cualquier operación en segundo plano que por otro lado no tenía sentido si el usuario no se encontraba interactuando con su dispositivo y por tanto no consultando el widget.

6.3 Novedades y mejoras aportadas

Llevado a cabo el análisis de las aplicaciones similares actualmente existentes en el mercado para dispositivos con sistema operativo Android, se detectaron algunas carencias que se pretendían subsanar con la creación de este producto.

En primer lugar, *Mi Bus Valencia* es la primera aplicación para la consulta de los horarios de llegadas de los autobuses de Valencia que incluye un widget. La naturaleza del panel electrónico de las marquesinas de la red, es ideal para transportar tal cual a un widget Android por su forma y sus características visuales; al igual que el panel electrónico se encuentra colgado en las marquesinas y se puede consultar de un vistazo, es posible anclar el widget al escritorio de un dispositivo móvil, como si de un panel electrónico de llegadas se tratase, ofreciendo la información que se necesita igualmente de un vistazo.

Además, es también la primera aplicación que no necesita intervención del usuario; este no necesita conocer ni buscar en las marquesinas o postes el código numérico de la parada, tampoco necesita conocer el nombre ni perder el tiempo introduciendo estos datos cuando quiere obtener la lista de llegadas del autobús que está esperando. En algunos de los productos analizados, el hecho de realizar una búsqueda por código de parada termina con un consumo de datos móviles inexplicablemente alto (es posible quizá que en algunas aplicaciones se descargue la lista completa de paradas antes de realizar alguna búsqueda). *Mi Bus Valencia* detecta automáticamente la ubicación del usuario y se encarga por él de averiguar cuál es el código de parada que corresponde a la parada actual del usuario, llevando a cabo inmediatamente después una consulta de las próximas llegadas. Por tanto, no se necesita nunca la intervención del usuario (tan sólo cuando el usuario desea cambiar a una búsqueda manual, que sigue existiendo en esta aplicación como en las que se encuentran ya en el mercado).

El widget, a modo de panel de llegadas, sigue funcionando incluso en ocasiones cuando los paneles reales de las marquesinas se quedan “colgados”, averiados o cuando estos implemente no existen, siempre que los datos sigan estando disponibles en la API proporcionada por el Ayuntamiento de Valencia.

6.4 Ampliaciones futuras

Se desea que el producto esté en continua ampliación introduciendo mejoras y nuevas funcionalidades para el usuario, que siga diferenciándola de las aplicaciones similares ya existentes en el mercado.

Algunas de las ampliaciones y mejoras programadas son:

- Menú lateral.
- Soportar orientación de la pantalla vertical y horizontal en la vista aplicación.
- Información de la parada actual en la vista widget.
- Ampliación del número de Favoritos.
- Control del saldo de la tarjeta bonobús.
- Poder compartir en redes sociales.

7. Bibliografía

Dashboards. *Android Developers*
Fecha de consulta: 21 de abril de 2015
https://developer.android.com/about/dashboards/index.html

App Widgets. *Android Developers API Guides*
Fecha de consulta: 25 de marzo de 2015
https://developer.android.com/guide/topics/appwidgets/index.html

Widget. (2014, 24 de diciembre). *Wikipedia, La enciclopedia libre*.
Fecha de consulta: 25 de marzo de 2015
http://es.wikipedia.org/wiki/Widget

Interfaz de programación de aplicaciones. (2015, 8 de abril). *Wikipedia, La enciclopedia libre*.
Fecha de consulta: 23 de abril de 2015
http://es.wikipedia.org/wiki/Interfaz_de_programación_de_aplicaciones

Servicio web. (2015, 21 de abril). *Wikipedia, La enciclopedia libre*.
Fecha de consulta: 23 de abril de 2015
http://es.wikipedia.org/wiki/Servicio_web

Location Strategies. *Android Developers API Guides*
Fecha de consulta: 4 de mayo de 2015
http://developer.android.com/guide/topics/location/strategies.html

Extensible Markup Language. (2015, 4 de junio). *Wikipedia, La enciclopedia libre*.
Fecha de consulta: 8 de mayo de 2015
http://es.wikipedia.org/wiki/Extensible_Markup_Language

XML. (2015, 5 de junio). *Wikipedia, La enciclopedia libre*.
Fecha de consulta: 8 de mayo de 2015
http://en.wikipedia.org/wiki/XML

JSON. (2015, 5 de mayo). *Wikipedia, La enciclopedia libre*.
Fecha de consulta: 8 de mayo de 2015
http://es.wikipedia.org/wiki/JSON

Supporting Multiple Screens. *Android Developers API Guides*
Fecha de consulta: 17 de mayo de 2015
http://developer.android.com/guide/practices/screens_support.html

Supporting Different Densities. *Android Developers API Guides*
Fecha de consulta: 17 de mayo de 2015
http://developer.android.com/training/multiscreen/screendensities.html