UNIVERSIDAD DE VALLADOLID

E.T.S.I. TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Tecnologías híbridas en el desarrollo de aplicaciones móviles. Desarrollo de una aplicación móvil como apoyo a una clínica pediátrica.

Autora:

Dña. Julia Santo Domingo Gómez

Tutora:

Dra. Dña. Míriam Antón Rodríguez

Valladolid, 19 de Mayo de 2015

TÍTULO:	**Tecnologías híbridas en el desarrollo de aplicaciones móviles. Desarrollo de una aplicación móvil como apoyo a una clínica pediátrica**
AUTORA:	**Dña. Julia Santo Domingo Gómez**
TUTORA:	**Dr. Dña. Míriam Antón Rodríguez**
DEPARTAMENTO:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTA:	**Dra. Dña. Míriam Antón Rodríguez**
VOCAL:	**Dr. D. David González Ortega**
SECRETARIO:	**Dr.D. Mario Martínez Zarzuela**
SUPLENTE:	**Dr.D. Francisco Javier Díaz Pernas**
SUPLENTE:	**Dra. Dña. Mª Ángeles Pérez Juárez**

FECHA:

CALIFICACIÓN:

Resumen

Actualmente las aplicaciones móviles forman parte de nuestra vida cotidiana y día tras día nuevos desarrolladores y empresas buscan crear su propia aplicación. Existen diferentes tipos de desarrollo de aplicaciones móviles que ofrecen ventajas que hay saber priorizar adecuadamente ya que toda ventaja suele conllevar inconvenientes que se pueden omitir. Este Trabajo Fin de Grado analiza los distintos tipos y se adentra en una elección lógica que muchos desarrolladores toman.

Después de la elección, se desarrollará una aplicación móvil haciendo uso de tecnologías híbridas, orientada a la telemedicina, concretamente al campo de la pediatría, que será destinada a una pyme, para evaluar a continuación las ventajas y dificultades encontradas al implementar esta solución.

Las aplicaciones *mHealth (Mobile Health)* se encuentran en continuo auge y son un factor importante en la asistencia sanitaria. Esta rápida evolución está directamente relacionada con el constante avance de las técnicas médicas que requieren en muchos casos de un apoyo telemático.

Debido por tanto a la evolución de las nuevas tecnologías, se ha realizado en este Trabajo Fin de Grado una aplicación que se pondrá en funcionamiento y que tiene como objetivo que pacientes y personal sanitario puedan beneficiarse del intercambio de datos e información entre ellos.

Palabras clave: m-health, PhoneGap, framework, jQueryMobile, JavaScript, phpMyAdmin

Abstract

Nowadays mobile applications are part of our daily lives and everyday new developers and companies look for their own mobile application. There are different types of mobile applications that offer plenty of advantages that we have to know how to adequately prioritize as every advantage entails inconveniences that are usually forgotten. This report aims to analyze the different types and explores the logic choices many developers take.

After my choice, I will develop a mobile application using hybrid technologies orientated towards telemedicine, more specifically in the area of pediatric care and orientated to the small and medium business.

Furthermore mobile health applications are in continuous growth. Moreover they are very important in the health care. This fast evolution is directly linked with the constant advances of the medical techniques that require telematics support in many cases.

Due to the evolution of new technologies I have developed an application that will work with the goal of exchanging data between patients and medical personnel and in this way benefiting both.

Key words: m-health, PhoneGap, framework, jQueryMobile, JavaScript, phpMyAdmin

Agradecimientos

A mi padre y a mi madre, por apoyarme, por ayudarme y por quererme tanto cada día. Gracias por haberme dado tanto, por confiar en mí y por haber estado siempre a mi lado.

A Ángela, por ser la mejor hermana, gracias a tus ánimos has hecho que cada día sea un poco más fuerte. Siempre me tendrás a tu lado.

A mis grandes y queridos amigos de teleco, gracias por haber hecho de éstos, unos años tan maravillosos, sin vosotros no hubiera sido lo mismo.

A mi tutora Míriam, por sus buenos consejos, su ayuda y su paciencia, desde el primer día hasta hoy.

A Syltec y a Rocío, por haber confiado en mí para realizar las prácticas en su empresa.

A mis amigos de la "resi" Blanca de Castilla, a mis amigas de Burgos y a todo el resto de mi familia, por los buenos momentos y porque siempre he podido contar con vosotros.

Gracia una vez, a todos los que habéis hecho que mi paso por Valladolid quede siempre en mi corazón.

ÍNDICE DE CONTENIDOS

1. **INTRODUCCIÓN** *15*

1.1 Introducción 15

1.2 Motivación y objetivos 17

1.3 Fases y métodos 17

1.4 Estructura de la memoria. 18

2. **APLICACIONES MÓVILES EN TELEMEDICINA** *20*

2.1 ¿Qué es el eHealth? 21

2.2 ¿Qué es el mHealth? 23

2.3 ¿Qué supone el mHealth en la vida de las personas? 25

2.4 Tipos de aplicaciones móviles de mHealth 26

2.5 Las aplicaciones móviles y la tecnología al servicio del negocio 27

3. **ESTUDIO DE LAS DIFERENTES TECNOLOGÍAS** *29*

3.1 Tipos de aplicaciones 30

3.1.1 Aplicaciones móviles nativas 31

3.1.1.1 Android 33

3.1.1.2 Apple iOS 38

3.1.1.3 Windows Phone 40

3.1.1.4 Blackberry RIM 42

3.1.1.5 Comparativa entre ambos 42

3.1.2 Aplicaciones web móvil 43

3.1.3 Desarrollo de aplicaciones híbridas 44

3.1.3.1 phoneGap 45

3.1.3.2 jQuery Mobile 48

3.1.3.3 LungoJS 49

3.1.3.4 Xamarin 50

3.1.3.5 Rhodes 50
3.1.3.6 Sencha Touch 50
3.2 Elección a partir de una comparativa entre ambos tipos de aplicaciones 51
3.3 Lenguajes de programación del lado del cliente y otros 53
3.3.1 HTML, HTML5 54
3.3.2 CSS, CSS3 56
3.3.3 JavaScript 60
3.3.4 AJAX 62
3.3.5 jQuery 63
3.4 Lenguajes de programación del lado del servidor 64
3.4.1 PHP 64
3.4.2 ASP 67
3.4.3 Servlets y JSP 68
3.4.4 Elección del lenguaje de programación del lado del servidor 68
3.5 Bases de datos 69
3.5.1 Bases de datos orientadas a objetos 70
3.5.2 Bases de datos relacionales 70
3.5.2.1 SQL 72
3.5.2.2 MySQL 72
3.6 Elección del lenguaje del lado del servidor y la base de datos 73
3.6.1 phpMyAdmin 74
3.7 Servicios WEB 75
3.7.1 Introducción a los servicios WEB 75
3.7.2 Arquitecturas comunes para servicios WEB 77
4. **DESCRIPCIÓN TÉCNICA DE LA APLICACIÓN PEDIATRÍA** *80*
4.1 Estructura de la base de datos 80
4.2 Relaciones entre las tablas de las bases de datos 84

4.3 Casos de uso y diagramas de actividad de las distintas funcionalidades de la aplicación. 87

4.4 Estructura de ficheros 100

4.5 Conexión de la aplicación móvil con el servidor 102

5. MANUAL DE USUARIO 103

5.1 Escritorio: Icono de la aplicación 104

5.2 Pantalla inicio de la aplicación: Menú principal 105

5.3 Pantalla ¿Quién somos? 108

5.4 Pantalla Pide tu cita 109

5.5 Pantalla Lactancia: Preguntas frecuentes FAQ 115

5.6 Pantalla Calendario de vacunas (Castilla y León) 117

5.7 Pantalla Consulta tu informe (Consulta tus últimas visitas al pediatra) 118

5.8 Pantalla Test de Denver 121

5.9 Pantalla Percentiles (Talla y peso) 122

5.9.1 Pantalla Percentiles: Talla niños 124

5.9.2 Pantalla Percentiles: Talla niñas 125

5.9.3 Pantalla Percentiles: Peso niños 126

5.9.4 Pantalla Percentiles: Peso niñas 127

5.10 Pantalla Ubicación (¿Dónde estamos?) 128

5.11 Pantalla Envío Mensaje (Email) 129

5.12 Opción Compartir: 131

6. PRESUPUESTO ECONÓMICO 133

7. CONCLUSIONES Y LÍNEAS FUTURAS 136

7.1 Conclusiones 136

7.2 Líneas futuras 137

BIBLIOGRAFÍA 139

ANEXOS TÉCNICOS.....*144*

ANEXO 1: PhoneGap 144

ANEXO 2: Publicar aplicaciones..... 146

ÍNDICE DE ILUSTRACIONES

Ilustración 1. Aplicaciones médicas: 3ª categoría de la *AppStore* de mayor crecimiento (De la Serna & Mugarza, 2014). 21

Ilustración 2. Estadísticas de las búsquedas en Google del término *mHealth* frente al término eSalud 23

Ilustración 3. ¿A quién van dirigidas las aplicaciones relacionadas con la salud? (De la Serna & Mugarza, 2014). 24

Ilustración 4. Estimación del crecimiento de la salud móvil en los próximos años (De la Serna & Mugarza, 2014). 27

Ilustración 5. Mercado aplicaciones móviles en 2015 29

Ilustración 6. Mercado aplicaciones móviles, según la empresa Vision Mobile en su informe en revista Developer Economics (6ª Edición) 30

Ilustración 7. Porcentaje de desarrolladores que usan cada plataforma como su plataforma predeterminada 31

Ilustración 8. Arquitectura de Android 34

Ilustración 9. JAVA: lenguaje multiplataforma 35

Ilustración 10. Creación de archivos *.dex* a partir de archivos *.class* 36

Ilustración 11. Arquitectura de iOS 38

Ilustración 12. Arquitectura Windows Phone 41

Ilustración 13. Empaquetamiento aplicaciones 46

Ilustración 14. Empaquetamiento de las aplicaciones en PhoneGap 47

Ilustración 15. Acceso a recursos por sistema operativo mediante *PhoneGap*. 48

Ilustración 16. Sistemas operativos soportados por *jQuery Mobile* 49

Ilustración 17. Comparación entre los distintos tipos de aplicaciones 51

Ilustración 18. Ventajas e inconvenientes según la orientación de cada aplicación 52

Ilustración 19. Icono HTML5 56

Ilustración 20. CSS3 57

Ilustración 21. Ejemplo uso: border-radius + shadows, con herramienta web 59

Ilustración 22. CSS3 Media *queries*. Responsividad. 60

Ilustración 23. Popularidad de los lenguajes de programación del lado del cliente 62

Ilustración 24. Comparación gráfica del modelo tradicional de aplicación web y del nuevo modelo propuesto por AJAX 63

Ilustración 25. Popularidad de los lenguajes de programación del lado del servidor. 64

Ilustración 26. Evolución de PHP 65

Ilustración 27. Relación de la extensión de distintos lenguajes de programación con su tráfico 66

Ilustración 28. Agentes fundamentales de los servicios web 76

Ilustración 29. Estructura de los mensajes SOAP 78

Ilustración 30. Base de datos de la aplicación PEDIATRÍA 81

Ilustración 31. Relación BBDD I 85

Ilustración 32. Relación BBDD II 86

Ilustración 33. Relación BBDD III 87

Ilustración 34. Diagrama UML Funcionalidades de la aplicación 88

Ilustración 35. Diagrama de actividad Pide tu cita 92

Ilustración 36. Diagrama de actividad Consulta tu informe 95

Ilustración 37. Diagrama de actividad Enviar mensaje 99

Ilustración 38. Icono de la aplicación Pediatría 104

Ilustración 39. Splash Screen: Imagen que se muestra al abrir la aplicación Pediatría 105

Ilustración 40. Menú principal de la aplicación 106

Ilustración 41. Pantalla ¿Quién somos? 109

Ilustración 42. Pantalla Inicio sesión 110

Ilustración 43. Pantalla Pide tu cita 111

Ilustración 44. Pantalla selección fecha y hora en el formulario para la reserva de cita 113

Ilustración 45. Pantalla de confirmación de cita 114

Ilustración 46. Solicitud de cita coincidente con otra ya existente 115

Ilustración 47. Pantalla Lactancia preguntas frecuentes (FAQ) 116

Ilustración 48. Pantalla calendario de vacunas castilla y león 117

Ilustración 49. Pantalla inicio de sesión 119

Ilustración 50. Pantalla Consulta tu informe 120

Ilustración 51. Pantalla Test de Denver 121

Ilustración 52. Pantalla Menú percentiles 123

Ilustración 53. . Pantalla Percentiles longitud (cm) niños 125

Ilustración 54. Pantalla Percentiles longitud (cm) niñas 126

Ilustración 55. Pantalla Percentiles peso (kg) niños 127

Ilustración 56. Pantalla Percentiles peso (kg) niñas 128

Ilustración 57. Pantalla Ubicación 129

Ilustración 58. Pantalla Envío mensaje electrónico 130

Ilustración 59. Confirmación de envío de mensaje electrónico para el usuario. Recepción correcta del mensaje en el buzón de entrada del pediatra 131

Ilustración 60. Opción Compartir la aplicación 132

Ilustración 61. Subida de un fichero .zip a PhoneGapBuild sin errores 145

Ilustración 62. Visualización de la página web *PhoneGapBuild* una vez concluida la subida del fichero .zip con la aplicación comprimida. 145

ÍNDICE DE TABLAS

Tabla 1. Principales lenguajes de programación por plataforma 32

Tabla 2. Ventajas y desventajas aplicaciones móviles nativas 33

Tabla 3. Ventajas y desventajas aplicaciones web 44

Tabla 4. Ventajas y desventajas aplicaciones híbridas 45

Tabla 5. Tabla de la BBDD: citas 82

Tabla 6. Tabla de la BBDD: datosusuario 83

Tabla 7. Tabla de la BBDD: informe 84

Tabla 8. Caso de uso ¿Quién somos? 89

Tabla 9. Caso de uso Inicia sesión 90

Tabla 10. Caso de uso Pide tu cita 91

Tabla 11. Caso de uso Lactancia: Preguntas frecuentes FAQ 93

Tabla 12. Caso de uso Calendario de vacunas (Castilla y León) 93

Tabla 13. Caso de uso Consulta tu informe 94

Tabla 14. Caso de uso Test de Denver 96

Tabla 15. Caso de uso Percentiles 96

Tabla 16. Caso de uso Ubicación 97

Tabla 17. Caso de uso Enviar Mensaje 98

Tabla 18. Caso de uso Limpiar formulario 100

Tabla 19. Base de datos de los usuarios. *User* = julia, Idusuario = 2 120

Tabla 20. Base de datos : Informes del usuario con Idusuario = 2 121

Tabla 21. Estimación presupuesto económico 134

ÍNDICE DE FOTOGRAFÍAS

Fotografía 1. Pantalla Menú en LG 9 Optimus 106

Fotografía 2. Pantalla Menú en Nexus 5 (izquierda) y Samsung GT S7580 (derecha) 107

Fotografía 3. Menú vertical en Tablet Sunstech 107

Fotografía 4. Menú horizontal en Tablet Sunstech 108

Fotografía 5. Selección fecha cita en el calendario en Nexus 5 horizontal 111

Fotografía 6. . Selección fecha cita en el calendario en LG 9 Optimus 112

Fotografía 7. Pantalla Lactancia en LG 9 Optimus 116

Fotografía 8. Pantalla Calendario de vacunas en Nexus 5 118

Fotografía 9. Pantalla Test de Denver en Samsung GT S7580 122

Fotografía 10. Pantalla Percentiles en Tablet Sunstech 124

Fotografía 11. Pantalla Envío Mensaje en Samsung GT S7580 (izquierda) y en LG 9 Optimus (derecha) 130

CAPÍTULO 1

1. INTRODUCCIÓN

1.1 Introducción

Actualmente la sociedad hace uso de las tecnologías de la información y comunicación (TIC) diariamente. Las personas utilizan estas tecnologías obteniendo así una mejor calidad de vida. Posiblemente podamos decir que una gran parte de nuestra sociedad "necesita" de estas herramientas tecnológicas, tanto para su trabajo, como para su vida personal.

Desde la aparición del primer transistor a mediados del siglo pasado, la industria tecnológica ha experimentado avances impresionantes, cuya consecuencia más obvia son las llamadas TIC. Las TIC han supuesto un cambio en el antiguo modelo de vida, y hoy en día la mayor parte de los sectores de la sociedad quieren ofrecer al usuario un enfoque adecuado para potenciar el funcionamiento de sus actividades. Es así como ofertantes y usuarios pueden beneficiarse de todas las ventajas que las nuevas tecnologías ofrecen.

En la década de los 90 empezaron a surgir los primeros móviles con tecnología GSM. A día de hoy, nadie se imagina un mundo sin teléfono móvil. En los últimos veinte años, la idea del teléfono móvil como instrumento para realizar llamadas, se ha quedado obsoleto a la vista de las múltiples funcionalidades para las que actualmente se están diseñando los mismos.

En concreto, los últimos años han estado marcados por un gran desarrollo de los dispositivos móviles y táctiles, tanto a nivel *hardware* con la aparición de *smartphones* y *tablets* que se aprovechan de nuevas y mejores pantallas y baterías, como a nivel de *software* con la irrupción de *Android*, *iOs*, *Symbian* y demás sistemas operativos móviles. El *smartphone* o teléfono inteligente es un pequeño y potente ordenador en miniatura que se caracteriza por su potente conectividad. A pesar de la limitación de tamaño de pantalla, este dispositivo está consolidado actualmente entre toda la población.

Existe un abanico infinito de facilidades ofrecidas por los *smartphones* , como por ejemplo, agenda personal conectada a nuestro ordenador personal, aplicaciones con las que reservar

hora en el gimnasio, juegos por internet, aplicaciones que miden la distancia recorrida, gestión de nuestra cuenta bancaria a través de la aplicación del banco…

Investigando sobre las diferentes posibilidades existentes en el mercado de las aplicaciones móviles, se optó por desarrollar una que estuviera relacionada con la medicina. Esta decisión, que personalmente me gustaba, surgió en parte debido a los numerosos proyectos relacionados con la medicina que la Escuela de Ingenieros de Telecomunicación de Valladolid tiene actualmente y también gracias a la empresa Sylfo Tecnoconsulting S.L.U. (SYLTEC), dónde realicé mis prácticas, curriculares, e inicié este proyecto.

En la actualidad disponemos de herramientas tecnológicas que permiten, allí donde nos encontremos y en cualquier momento, acceso a la red y consecuentemente acceso a todo tipo de información y aplicaciones relativas a la salud. Podemos ofrecer servicios a para ciudadanos, pacientes y profesionales, mediante la utilización de la tecnología disponible para los dispositivos móviles

La ingeniería ha sabido dar un paso adelante introduciéndose en el difícil campo de la medicina. Muchos de los grandes avances médicos que benefician a toda la población se deben al esfuerzo que ingenieros y médicos consiguen trabajando conjuntamente, con el objetivo de conseguir y dar respuesta de forma más eficaz y de mayor calidad a los múltiples problemas que van surgiendo en este ámbito.

El desarrollo del conocimiento en áreas tales como la genética y la biología molecular está parcialmente basado en el avance de la ciencia informática y sus aplicaciones en el estudio de dichas áreas.

Se ha hecho un progreso muy significativo que ha generado una enorme cantidad de datos, para cuyo análisis es necesaria la especialización de ordenadores y herramientas afines, que son capaces de ordenarlos y gestionarlos de forma más rápida y resolutiva.

Concretamente, el Departamento de Teoría de la Señal y Comunicaciones, e Ingeniería Telemática de la Universidad de Valladolid es uno de los grupos españoles más activos en la bibliografía internacional sobre salud móvil. Un ejemplo es la aplicación conocida como Heartkeeper, para la autogestión de las enfermedades del corazón. Además de desarrollar un elaborado método de medida de la calidad de experiencia del usuario de aplicaciones móviles para la salud, han realizado también estudios de coste-efectividad y han creado una guía de recomendaciones para desarrolladores de aplicaciones móviles.

Por todos estos motivos y puesto que los aplicativos referidos a la *salud móvil* generan un interés creciente, se pensó que una aplicación *m-health* dedicada exclusivamente a la Pediatría, era una buena idea con la que trabajar en el presente TFG.

A pesar de comprobar que existe una gran variedad de iniciativas y de herramientas para la mejora de la continuidad asistencial, para la colaboración y la toma de decisiones de los profesionales y su comunicación con los pacientes, se observó que ninguna de las

relacionadas directamente con la pediatría, era tan concreta y fácil de usar como la que se pretende diseñar en este TFG.

1.2 Motivación y objetivos

Una vez descrita la contextualización en la que se ha diseñado este trabajo fin de grado, puede describirse el fin último con el que se ha realizado:

Diseñar una aplicación móvil híbrida mediante el uso del *framework* de *jQuery Mobile* como herramienta de apoyo para los usuarios de un Clínica Pediátrica.

Para llegar a este objetivo final, se han establecido objetivos parciales para guiarnos en el desarrollo de este trabajo fin de grado. Los objetivos parciales han sido los siguientes:

- Investigar sobre el panorama actual de la salud móvil para llegar a comprender como el *m-health* está afectando diariamente a la vida de las personas.
- Realizar una comparativa entre las diferentes tecnologías móviles existentes, observando las ventajas y desventajas de cada una de ellas.
- Argumentar y justificar un criterio propio para decidir qué tecnología móvil es más apropiado usar en este TFG.
- Realizar un ejemplo de una aplicación móvil orientada a la pediatría mediante una tecnología móvil híbrida. Justificar los motivos por los que se ha llevado a cabo el desarrollo de ésta mediante el *framework* de *jQuery Mobile*.
- Conseguir que la aplicación móvil PEDIATRÍA desarrollada, se caracterice por su escalabilidad, su facilidad de uso y el dinamismo de su contenido.

Personalmente creo que en este trabajo de fin de grado me centro en realizar un estudio del arte de la salud móvil actual y de las diferentes tecnologías y plataformas móviles existentes hasta el día de hoy. Con esto, he querido desarrollar una aplicación móvil como una herramienta de apoyo a una Clínica Pediátrica que facilitara la comunicación entre los clientes y personal sanitario , de la que ambas partes podrán beneficiarse.

1.3 Fases y métodos

Para conseguir los objetivos propuestos , se han seguido los siguientes pasos en el desarrollo del proyecto:

- Estudio del panorama actual en el mercado de las aplicaciones móviles orientadas a la medicina, m-*health*.
- Documentación. Realización de una comparativa entre las diferentes tecnologías móviles y servicios web.
- Análisis. Se estudian las posibilidades de desarrollar la aplicación PEDIATRÍA mediante el *framework* de *jQuery Mobile* y *PhoneGapBuild*.

- Desarrollo de la nueva aplicación: Programación de la aplicación, tanto de la lógica de negocio como el diseño de las interfaces que se mostrará al usuario. Ha sido la fase del proyecto que más tiempo ha requerido , debido a la falta de experiencia en la programación del lenguaje utilizado para el desarrollo de esta aplicación. Antes de comenzar con el desarrollo de la aplicación como tal, hubo una fase de documentación y aprendizaje de los lenguajes que se iban a utilizar.

  Esta fase previa al desarrollo se realizó con mucha dedicación e ilusión , pues lo que estaba estudiando tenía un objetivo definido en mi TFG y se realizaba para conseguir desarrollar con éxito la aplicación buscada.
- Pruebas. Se realizan correcciones a posibles fallos en el desarrollo.
- Conclusiones y líneas futuras para mejorar y ampliar la aplicación.

1.4 Estructura de la memoria.

Esta memoria se ha estructurado en siete capítulos en los que se pretende abarcar los ámbitos más importantes del desarrollo de un proyecto.

En el presente capítulo se hace una introducción al tema tratado en este trabajo de fin de grado y se detallan los pasos que se han seguido en la desarrollo de este proyecto.

En el capítulo dos se empezará definiendo lo que es la telemedicina o *e-Health* para introducir con ello el tema de la salud móvil, o lo que es denominado más comúnmente *m-Health*. Se describirá este nuevo modelo de aplicaciones móviles y se establecerá una visión global del panorama actual de las aplicaciones móviles orientadas a la medicina.

El siguiente apartado trata de las distintas tecnologías móviles existentes actualmente. Se comienza comparando los tres tipos de aplicaciones móviles, nativas, híbridas o *weba*pps, estableciendo una serie de ventajas y desventajas para cada tipo. A su vez se van a detallar para cada tipo de aplicación las tecnologías móviles más comunes.

Una vez descritas dichas tecnologías, se describirá el proceso por el cual se ha elegido el modelo de aplicación híbrida como la mejor opción para este trabajo de fin de grado.

A continuación comienzan los capítulos en los que se describirá la aplicación PEDIATRÍA desarrollada en este proyecto. Primeramente se detallará la aplicación a nivel interno con detalles más funcionales y después se describirá a nivel de usuario. Para concluir se hará un análisis económico de ésta.

Como se ha comentado previamente, en el cuarto capítulo se realiza un análisis funcional de la aplicación móvil orientada a la pediatría que se ha desarrollado como aplicación *m-health* de ejemplo para este proyecto. Mediante la ayuda de diagramas UML, tablas con los casos de uso y diagramas de actividad - se describirán todas las funcionalidades que caracterizan esta aplicación móvil.

En el capítulo cinco se describe un manual de usuario de la aplicación PEDIATRÍA. Con éste se pretende enseñar y facilitar al usuario el manejo y uso de las funcionalidades que esta aplicación le ofrece. Siguiendo los pasos de este manual el usuario podrá beneficiarse más rápidamente de las ventajas de uso de esta aplicación.

El capítulo sexto trata de dar a conocer los gastos que supone la realización de una aplicación móvil como la diseñada en este trabajo fin de grado. Por este motivo y para acercar este TFG lo más posible a un proyecto real, se ha llevado a cabo la realización de un presupuesto económico, en el que se ha buscado ajustar los gastos a la realidad , de la manera más precisa posible.

Para concluir esta memoria, en el capítulo siete se describen las conclusiones obtenidas de este proyecto, valorando si se ha conseguido alcanzar los objetivos propuestos en el capítulo uno. Además de una opinión personal, también se explican algunas ideas que podrían llevarse a cabo en un futuro para mejorar la aplicación, optimizarla y dotarla de más prestaciones para seguir beneficiando tanto al personal sanitario como a los usuarios finales.

CAPÍTULO 2

2. APLICACIONES MÓVILES EN TELEMEDICINA

No cabe ninguna duda de que las nuevas tecnologías de la información y la comunicación se han abierto camino en el mundo de la medicina, irrumpiendo con fuerza, llegando no sólo para quedarse sino además para cambiar el paradigma de la atención sanitaria en su conjunto.

La salud móvil, dos palabras que unidas poca gente pronunciaba hace muy pocos años (se hablaba de telemedicina y de '*eHealth'*), se está transformando en uno de los pilares básicos de la sanidad en todas partes, incluidos los países en vías de desarrollo. La premisa que más se adapta a esta nueva tecnología podría definirse como: '*Everywhere, everytime, everyone',* o lo que es lo mismo, *'Todo el mundo, en cualquier sitio a cualquier hora'* (De la Serna & Mugarza, 2014).

El mercado actual del *mobile health*, o lo que es lo mismo, salud móvil, es un sector en pleno proceso de expansión del que se espera que genere un volumen de negocio de más de 6.000 millones de euros en este año 2015. Hoy en día el *mobile health*, también conocido como mHealth está generando un alto interés en todas las partes involucradas en el ámbito de la salud; desde los profesionales de este sector a la sociedad en general, nadie quiere perderse los beneficios de esta nueva forma de interacción con los servicios sanitarios. La implantación del mHealth supondrá importantes beneficios sociales y económicos e impulsará un cambio en el sector sanitario.

Ilustración 1. Aplicaciones médicas: 3ª categoría de la *AppStore* de mayor crecimiento (De la Serna & Mugarza, 2014).

Lo cierto es que la telemedicina va a permitir incrementar la esperanza de vida saludable mediante un mayor control de la salud de las personas, no sólo de los enfermos crónicos, sino también mediante la prevención de factores de riesgo.

Este capítulo se centrará en dar una serie de definiciones de los nuevos términos que han surgido a raíz de la telemedicina, y tratará de hacer una distinción en los diferentes usos que pueden tener las aplicaciones móviles en el mHealth, mobile Health o mSalud.

Para finalizar se hará un estudio del mercado actual de las aplicaciones mHealth, que destacan por su calidad, utilidad médica y contribución a la mejora de la salud, y cuya gran importancia ha servido como referencia para la realización de este TFG.

2.1 ¿Qué es el eHealth?

De acuerdo a la definición de la OMS, la telemedicina, *eHealth* o eSalud se define como "El suministro de servicios de atención sanitaria, en los que la distancia constituye un factor crítico, por profesionales que apelan a las tecnologías de la información y de la comunicación con objeto de intercambiar datos para hacer diagnósticos, preconizar tratamientos y prevenir enfermedades y heridas, así como para la formación permanente de los profesionales de atención de salud y en actividades de investigación y evaluación, con el fin de mejorar la salud de las personas y de las comunidades en que viven".

Esto puede interpretarse como la aplicación de la tecnología a la salud para mejorar las herramientas de los actores que intervienen en el proceso sanitario con el fin de mejorar la calidad de la atención de los pacientes (Jiménez Díaz, 2014).

Por lo tanto la eSalud aplica las Tecnologías de Información y Comunicación en los aspectos que afecten al cuidado de la salud, incluyendo entre ellas el diagnóstico, el seguimiento de los pacientes y la gestión de las organizaciones implicadas en estas actividades.

De esta manera, los profesionales tienen la oportunidad de mejorar su acceso al historial clínico de los pacientes, prescribir electrónicamente diferentes recetas, así como consultar las principales publicaciones médicas especializadas. No obstante los pacientes, también adquieren ventajas en esta nueva dimensión; tienen la oportunidad de acceder a la atención médica con independencia del lugar en el que se encuentren, fomentando así la prevención, el bienestar y la vida independiente y autónoma de las personas. Se demuestra por tanto, que la eSalud supone un ahorro de costes y un aumento en la eficacia del sistema sanitario (Com Salud, 2014).

Principales mejoras que ha supuesto para la sociedad el *eHealth*:

- Registros electrónicos de salud: Historias clínicas fácilmente accesibles por los profesionales sanitarios.
- Control de parámetros fisiológicos y biométricos de manera electrónica
- Sistemas de información de pacientes y del hospital: Combinación de información clínica y administrativa, para mejorar la seguridad del paciente, la eficiencia clínica, la gestión del centro y su personal, etc.
- Asistencia y cuidado de pacientes crónicos o personas mayores dependientes con la aplicación de la telemedicina
- Sistemas de apoyo a las decisiones, que ofrezcan orientaciones diagnósticas y terapéuticas.
- Rehabilitación remota mediante dispositivos tecnológicos

Con ayuda de la herramienta *Google Trends*, que Google ofrece a todos los usuarios de manera gratuita, se ha realizado un estudio de cómo ha ido evolucionando desde el año 2005 hasta hoy (Abril de 2015) la búsqueda en Google de los términos eSalud (*eHealth*) y *mHealth*.

Como puede apreciarse en la ilustración inferior, las búsquedas sobre eSalud, empiezan aparecer sobre el 2005 y esta tendencia va aumentando hasta finales del año 2012. Es a partir de dicho año, cuando empieza a ganar popularidad entre la sociedad el nuevo término sobre salud móvil. Desde entonces, se realizan más búsquedas en Google sobre *mHealth* que sobre *eHealth*.

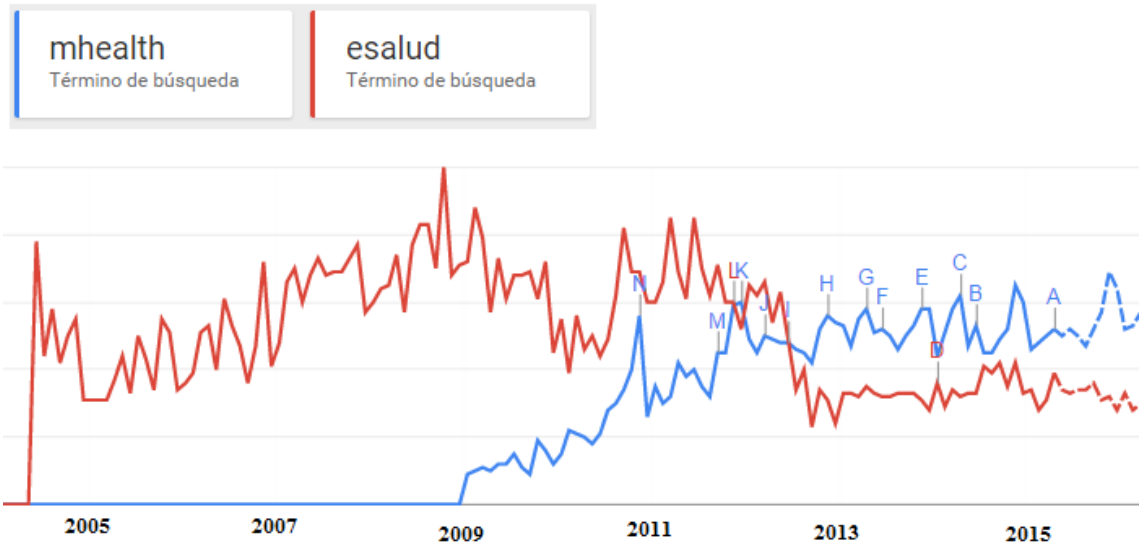


Ilustración 2. Estadísticas de las búsquedas en Google del término *mHealth* frente al término eSalud

2.2 ¿Qué es el mHealth?

Según la OMS, se define la *mHealth* (del inglés, *mobile health*), salud móvil o mSalud como "la práctica de la medicina y la salud pública soportada por dispositivos móviles como teléfonos móviles, dispositivos de monitorización de pacientes, asistentes personales digitales y otros dispositivos inalámbricos." Además concluye: "la mHealth es un componente de la *eHealth*" (Cabo Diez, 2015).

Esta definición incluye dos aspectos esenciales: la tecnología, que se refiere no sólo a la movilidad de los dispositivos sino la conectividad inalámbrica de los mismos (algunos autores hablan de salud inalámbrica - *wireless health*); y la gestión de los datos de salud, que puede basarse en nuevos paradigmas, como el papel de las redes sociales o la trazabilidad de personas o el impulso de la promoción de vida saludable. Los fines de esa gestión de datos de salud pueden ser asistenciales (provisión de servicios médicos), de salud pública, docentes, de investigación, de gestión de calidad, económicos, etc.

Los productos relacionados con mHealth se basan en telefonía móvil, *smartphones*, tabletas, asistentes personales (PDA), dispositivos de monitorización de pacientes u otros dispositivos móviles inalámbricos. En este concepto se incluyen también las aplicaciones ("apps") dedicadas a estilos de vida o promoción de vida saludable, las cuales también pueden conectarse a dispositivos médicos o sensores (brazaletes, relojes), sistemas de asesoramiento personalizado, sistemas de recordatorio de medicación, etc. En general, son soluciones que permiten el acceso a información útil donde y cuando son necesarias (Sociedad española de informática de la salud, 2015).

El mHealth pretende compartir el conocimiento y hacerlo además de forma ilustrativa, fácilmente comprensible y a través de una herramienta tecnológica absolutamente versátil, de fácil manejo y usabilidad, es algo que conforma un espacio de comunicación abierto, que permite ya no sólo entender, sino, además, compartir experiencias en el entorno que más preocupa al ser humano, el de su propia salud (De la Serna & Mugarza, 2014).

Ahora pacientes y médicos están más conectados y existe una mayor preocupación por la salud.

Ilustración 3. ¿A quién van dirigidas las aplicaciones relacionadas con la salud? (De la Serna & Mugarza, 2014).

La evolución de las telecomunicaciones y de Internet facilitó la transmisión de datos e imágenes a distancia. Es precisamente este hecho el que transforma el concepto de salud, vigente hasta ahora, en el de 'salud móvil', es decir, la salud a distancia (deja de ser fundamental donde es el médico, la enfermera, el paciente o el historial médico, y mediante herramientas como un móvil o una tableta, el proceso se optimiza y se personaliza -en tiempo, recursos y calidad- en las fases de prevención, educación, diagnóstico y tratamiento).

Las soluciones de mHealth potencian la eficiencia, la eficacia y la universalización de la atención sanitaria, la mejora de la toma de decisiones médicas, la reducción de las hospitalizaciones y la optimización del tiempo de que disponen los médicos y, por tanto, contribuyen a mejorar el sistema sanitario y la atención final al paciente.

2.3 ¿Qué supone el mHealth en la vida de las personas?

El concepto mHealth debe ser global, por ejemplo para la Unión Europea, la salud móvil es una de las prioridades de la Agenda Digital para Europa, contemplando las redes de comunicaciones inalámbricas, sistemas de telecomunicaciones, la integración con estaciones médicas móviles y con otros dispositivos conectados por cables, el desarrollo de aplicaciones y middleware, la historia clínica móvil, la seguridad en las comunicaciones, etc. (Sociedad española de informática de la salud, 2015).

El ámbito sanitario está evolucionando hacia un modelo basado en la medicina personalizada. Gracias al desarrollo de la tecnología, el sector tiene la posibilidad de transformarse para ofrecer servicios mucho más individualizados, participativos y preventivos. El estudio pone de manifiesto que la implantación de la mHealth supondrá importantes beneficios sociales y económicos e impulsará un cambio en el sector sanitario.

Algunos cambios que la sociedad debe entender y aceptar ante la nueva era del mHealth según el Dr. José Luis de la Serna (De la Serna & Mugarza, 2014).

1. Nuevas relaciones médico-paciente. Los profesionales de la salud haciendo uso en su mayoría de *smartphones* y *tablets* como herramienta de trabajo, podrán tener un contacto más directo con sus pacientes, lo que facilitará la comunicación y el seguimiento del tratamiento
2. Aplicaciones para la mejora de la gestión sanitaria. Petición de citas, consultas entre especialistas, el acceso al historial médico, etc.
3. Monitorización. Gracias a la incorporación de sensores cada vez es más fácil registrar parámetros vitales que pueden ayudar a detectar determinados tipos de comportamientos irregulares en el cuerpo humano, que facilitarán el diagnostico de ciertas enfermedades.
4. Almacenamiento inteligente de datos: '*big data*', que, en el caso de la salud, trata precisamente de cómo integrar la cantidad de datos provenientes de dispositivos, redes sociales, aparatos médicos, registros en papel, etc., respetando siempre la privacidad, y estructurarlos de forma eficaz para predecir, prevenir y personalizar enfermedades.
5. Empoderamiento del paciente. El paciente se convierte ahora en un elemento activo en el seguimiento de su enfermedad. Es decir, cada persona ha de ser consciente de la importancia que el autocuidado puede desempeñar en su salud, comer bien, hacer ejercicio, dormir las horas necesarias, así como estar informado y participar de forma activa en el abordaje de su enfermedad.
6. Cambio de hábitos. Se intentará fomentar la implicación y motivación del paciente haciendo que modifique comportamientos potenciales que redunden en beneficios para la salud. Existen ya numerosas aplicaciones que incentivan hábitos de vida más saludables, por ejemplo, las que controlan la actividad física, la dieta o los cigarrillos consumidos por el usuario.

2.4 Tipos de aplicaciones móviles de mHealth

Dentro del campo del *mobile health* se pueden encontrar numerosas aplicaciones que cubrirán muchas de las necesidades que los usuarios demandan hoy en día. Estas aplicaciones son utilizadas con un sinfín de propósitos atendiendo cada usuario a su interés particular.

El hecho de que haya tantas aplicaciones sobre mHealth, muchas de ellas parecidas entre sí, puede ser debido a que éstas se vayan creando según profesionales o usuarios vayan encontrando carencias en otras aplicaciones que hayan utilizado. Estos motivos llevan a desarrolladores a seguir mejorando sus aplicaciones para conseguir abarcar el mayor número de usuarios posibles.

Cada desarrollador, elegirá que tipo de aplicación desea construir. Éstas pueden diferenciarse atendiendo, por ejemplo, al criterio recomendado por el Dr. Fernando Mugarza, en su informe (De la Serna & Mugarza, 2014):

• Aplicaciones informativas: las que tienen como principal función aportar información completa y detallada sobre alguna patología determinada o área de especialización médica, ya sea en formato texto, imagen o vídeo.

• Educación y sensibilización: las que también aportan información actualizada sobre alguna enfermedad pero quieren ir más allá, facilitando la educación activa por parte del paciente o público al que va dirigido, siguiendo el modelo de 'paciente experto' promovido por la Universidad de Stanford al que hemos hecho referencia en uno de los apartados anteriores.

• Registro y monitorización: las que se centran en el registro de parámetros físicos y el seguimiento de determinada actividad o comportamiento por parte del usuario.

• Ayuda al diagnóstico: las que facilitan el proceso de identificación de una determinada enfermedad o alteración médica, aportando datos de valor para el profesional sanitario.

• Seguimiento de tratamiento: las que sirven de apoyo al paciente para mejorar su adherencia al tratamiento o tener un control sobre cómo está llevando el mismo.

• Gestión y utilidades: las que aportan información útil relacionada con la gestión sanitaria, citas médicas, localización de centros y profesionales de la salud, etc.

Hemos planteado tres áreas bien diferenciadas a la hora de mostrar las apps en la infografía, atendiendo a la siguiente clasificación:

• Aplicaciones con contenido más científico dirigidas fundamentalmente a profesionales de la salud.

• Aplicaciones que tienen un amplio contenido médico pero que están concebidas para servir de apoyo a pacientes y familiares/cuidadores en el proceso de una enfermedad o problema médico.

• Aplicaciones con contenido más generalista que facilitan la labor de la prevención primaria y el cuidado de la salud, dirigidas a la sociedad en general

2.5 Las aplicaciones móviles y la tecnología al servicio del negocio:

Como ya se dijo en la introducción, las aplicaciones móviles de salud, la llamada 'mHealth', podrían generar a finales de 2015 un volumen de negocio de 6.000 millones de euros en España, debido a que se espera que un tercio de los usuarios de *'smartphones'* tenga instalada, al menos, una de estas aplicaciones, según *The App Date*, con el impulso del Observatorio Zeltia y la colaboración de la Cátedra conjunta Zeltia-Universidad Rey Juan Carlos de Madrid, Wake App Health y Red de Innovación (RDi).

Ilustración 4. Estimación del crecimiento de la salud móvil en los próximos años (De la Serna & Mugarza, 2014).

Según estimaciones recientes, hay disponibles en el mercado mundial 97.000 aplicaciones de mHealth, de las cuales aproximadamente el 70% están dirigidas directamente a los consumidores mientras que el 30% se enfocan al segmento del profesional sanitario. La funcionalidad más común a nivel general es la de aportar información (39,8%), seguida de la proveer de instrucciones de uso (21,4%) y registrar o capturar datos del usuario (18,7%).

Estas aplicaciones están creciendo a un ritmo vertiginoso, existiendo actualmente 97.000 en el mercado, lo que las sitúa en la tercera categoría con mayor crecimiento, sólo por detrás de las aplicaciones de juegos y de utilidades. Además, se estima que su presencia crecerá un 23% anual en los próximos 5 años y que en 4 años los ingresos aumentarán un 511% (Europa press, 2014).

De todas ellas, según el último estudio del IMS Institute for Healthcare Informatics, recogido por The App Date en su informe, el 70% están dirigidas al público en general, a través de los segmentos de bienestar y ejercicio físico, y el 30% están ideadas para los profesionales

sanitarios y los pacientes. La funcionalidad más común a nivel general es la de aportar información (39,8%), seguida de la proveer de instrucciones de uso (21,4%) y registrar o capturar datos del usuario (18,7%).

En el caso concreto de las aplicaciones disponibles en español analizadas en el nuevo estudio, el 24% son informativas, el 22% están dedicadas a la monitorización de parámetros físicos y el 18% las que facilitan el seguimiento del tratamiento.

A continuación se enumeran algunas de las mejoras que han surgido gracias a la implementación de programas de *mHealth* según el informe *Socio-economic impact of mHealth, an assessment report for the European Union de PricewaterhouseCoopers* (PWC, 2013):

- Ciudadanos y pacientes pueden ganar en calidad de vida gracias a la prevención ya sistemas de monitorización remota. Únicamente en el ámbito europeo se calcula que 141 millones de pacientes podrían mejorar su calidad de vida

- Los centros de salud pueden mejorar su gestión y ser más eficientes. Con un ahorro estimado de 42 millones de días laborables de doctores, podrían tratarse hasta 126 millones de pacientes adicionales

- Las administraciones públicas, que pueden encontrar nuevas soluciones para dar un servicio más satisfactorio y rápido a los ciudadanos y a la vez ahorrar gastos. Se estima que de cara al 2017, la UE podría ahorrarse 76.000 millones de euros

- Las empresas contarán con nuevas oportunidades de desarrollo y negocio. Para las iniciativas privadas supondría un ahorro de 23 billones de euros.

Actualmente la Unión Europea, a través del nuevo programa de investigación e innovación Horizonte 2020, está financiado proyectos de mHealth, enfocados sobre todo a la personalización de la atención sanitaria (PHC), el impulso al papel del ciudadano, a través de autogestión de su salud y de la enfermedad, la promoción de la salud y la prevención de enfermedades (Sociedad española de informática de la salud, 2015).

CAPÍTULO 3

3. ESTUDIO DE LAS DIFERENTES TECNOLOGÍAS

El mercado de las aplicaciones móviles no para de crecer. Cada día salen al mercado nuevas y novedosas aplicaciones, que no hacen sino que satisfacer las necesidades de los millones de usuarios en todo el mundo que continuamente están conectados a sus dispositivos móviles.

Estos dispositivos están controlados por sistemas operativos, los cuáles detallaremos más adelante siguiendo una distinción entre aquellos que se usan para aplicaciones móviles nativas, hibridas o para aplicaciones web-móviles.

En este proyecto, en el que describiré una aplicación móvil enfocada al mundo de la medicina, en concreto para el campo de la pediatría, explicaré que ventajas y desventajas tiene el desarrollar una aplicación según un sistema operativo u otro y haré un recorrido por todas ellas para al final, según un criterio propio, exponer los motivos que me han llevado a programar mi aplicación con una tecnología híbrida.

Puesto que en el mundo, hay ya casi tantos teléfonos móviles como personas, la mayoría de las empresas no quieren dejar pasar la oportunidad de unirse a este negocio y crear una aplicación que dé respuesta a sus necesidades. Sin embargo, la mayoría desconocen qué tipos de aplicaciones móviles existen y cuál es la mejor opción según las necesidades (Prieto Cortera, 2015).

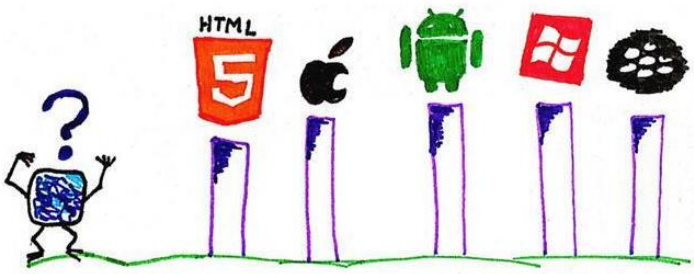


Ilustración 5. Mercado aplicaciones móviles en 2015

3.1 Tipos de aplicaciones

Existen varios sistemas operativos móviles en el mercado mundial, el cual está dominado claramente por Android, como se puede ver en la siguiente figura, sistema operativo de Google. Su principal competidor es el sistema operativo iOS de Apple, aunque bastante lejos de él. También existen otros sistemas con mucha menor penetración en el mercado como son *BlackBerry*, *Symbian*, *Windows Phone* o *Linux* (Ortiz, 2013).

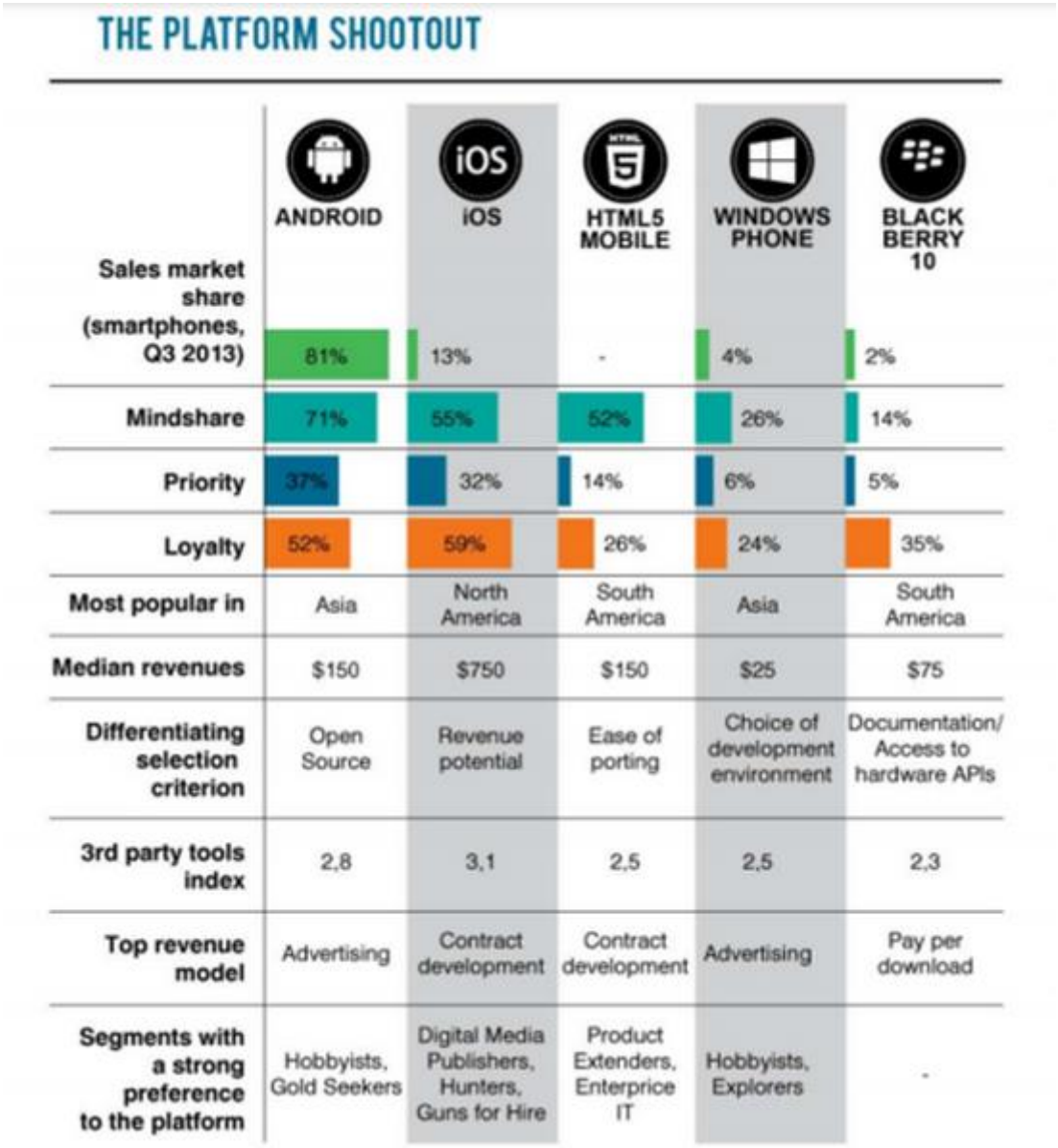


Ilustración 6. Mercado aplicaciones móviles, según la empresa Vision Mobile en su informe en revista Developer Economics (6ª Edición)

La empresa *VisionMobile* realiza desde hace tiempo su informe *Developer Economics*, y en su sexta edición, que estudia el primer trimestre de 2014, se analizan las tendencias del mercado en cuanto al interés de los desarrolladores por el segmento de la movilidad.

Fijándonos en la figura anterior, el estado del desarrollo de aplicaciones móviles parece bastante claro: Android e iOS se repartieron el 94% de las ventas de software en *smartphones* en el cuarto trimestre de 2013 según ese estudio. De ese porcentaje el 71% se dedicó a Android, mientras que el 55% desarrolla en iOS. Como se puede comprobar, parte de los desarrolladores trabaja de forma paralela en ambas plataformas móviles.

Lo interesante no es tanto lo que se da hoy en día como las tendencias que deja entrever ese estudio: por ejemplo, que iOS es la plataforma preferida en países desarrollados, y los que solo se dedican a dicha plataforma (el 59% de los desarrolladores encuestados) son más en porcentaje que los que solo se dedican a Android (el 52%) (Pastor, 2014).

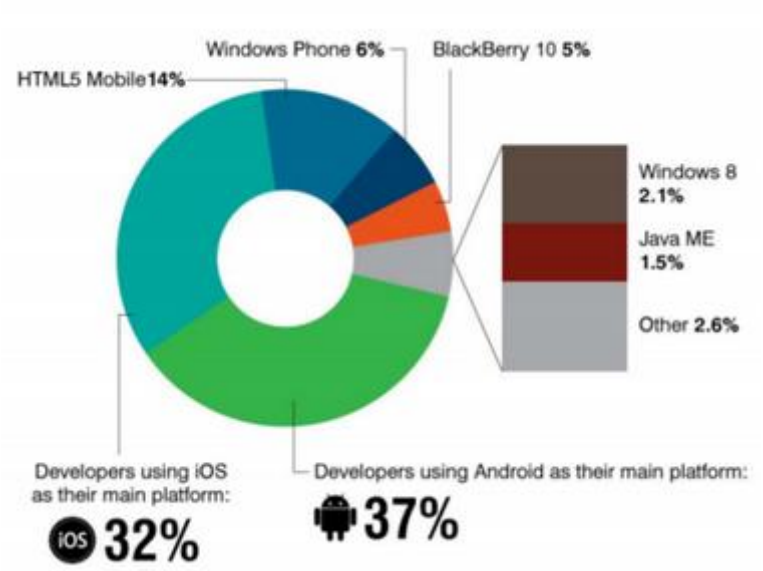


Ilustración 7. Porcentaje de desarrolladores que usan cada plataforma como su plataforma predeterminada

Según la encuesta realizada a principios de este año 2015, por la *Vision Mobile*, como se observa en la ilustración anterior, el 37% de los desarrolladores priorizan Android, por un 34% en la anterior edición, mientras que iOS se ha mantenido estable con un 32% de interés en priorizar para la plataforma de Apple. En el caso de HTML5, la prioridad se sitúa en un 14%, mientras que en el tercer trimestre de 2013 era del 17%: la falta de herramientas de desarrollo maduras parece haber sido la causa de ese descenso.

3.1.1 Aplicaciones móviles nativas

Una aplicación nativa es la que se desarrolla de forma específica para un determinado sistema operativo, llamado *Software Development Kit* o *SDK*. Cada una de las plataformas, *Android*, *iOS*

o *Windows Phone*, tienen un sistema diferente, por lo que si se pretende que una aplicación (*App*) esté disponible en todas las plataformas se deberán de crear varias apps con el lenguaje del sistema operativo seleccionado.

Operating System	Manufacturer	Programmming Language
Symbian OS	Symbian Foundation	C++
BlackBerry OS	RIM	Java
iPhone OS	Apple	Objective C
Windows Phone	Microsoft	C#
Android	Google	Java

Tabla 1. Principales lenguajes de programación por plataforma

Por ejemplo:

- Las apps para *iOS* se desarrollan con lenguaje *Objective*-C
- Las apps para Android se desarrollan con lenguaje *Java*
- Las apps en *Windows Phone* se desarrollan en .Net

Cuando se habla de desarrollo móvil casi siempre se refiere a aplicaciones nativas. La principal ventaja con respecto a los otros dos tipos (aplicaciones híbridas o aplicaciones Web – *WebApp*), es la posibilidad de acceder a todas las características del hardware del móvil: cámara, GPS, agenda, dispositivos de almacenamiento y otras muchas. Esto hace que la experiencia del usuario sea mucho más positiva que con otro tipo de aplicaciones.

La descarga e instalación de estas aplicaciones se realiza siempre a través de las tiendas de aplicaciones (app store de los fabricantes). Esto facilita el proceso de marketing y promoción lo que es vital para dar visibilidad a una aplicación.

Está claro que si el coste no es un obstáculo a la hora de programar, o si se tiene la certeza de que la aplicación será rentable, la mejor opción será siempre el desarrollo de una aplicación nativa para cada plataforma (*iOS*, *Android* y *Windows Phone*) (Lance Talent, 2015).

Ventajas	Inconvenientes
• Acceso completo al dispositivo • Mejor experiencia del usuario • Visibilidad en APP Store • Envío de notificaciones o "avisos" a los usuarios • La actualización de la app es constante	• Diferentes habilidades / idiomas / herramientas para cada plataforma de destino • Tienden a ser más caras de desarrollar • El código del cliente no es reutilizable entre las diferentes plataformas

Tabla 2. Ventajas y desventajas aplicaciones móviles nativas

3.1.1.1 Android

Google compró al desarrollador original de Android, Android inc., en 2005. El sistema es desarrollado a partir de ese momento por la *Open Handset* Alliance. Liderada por Google, es una alianza comercial de 84 compañías que buscan desarrollar estándares abiertos para dispositivos móviles. Entre ellas, se encuentra las principales compañías del sector tecnológico, a excepción de Apple, Microsoft, Nokia y algunos competidores más. Varios operadores móviles de renombre internacional, como Telefónica, Vodafone y T-Mobile, también forman parte de la alianza (Open Handset Alliance, 2012).

Android es una pila de software para dispositivos móviles que incluye un sistema operativo, middleware y diversas aplicaciones. El Android SDK aporta las herramientas y APIs necesarias para empezar a desarrollar aplicaciones para la plataforma Android usando el lenguaje de programación Java. En la siguiente ilustración se puede apreciar un diagrama en el que se muestra la arquitectura de Android.

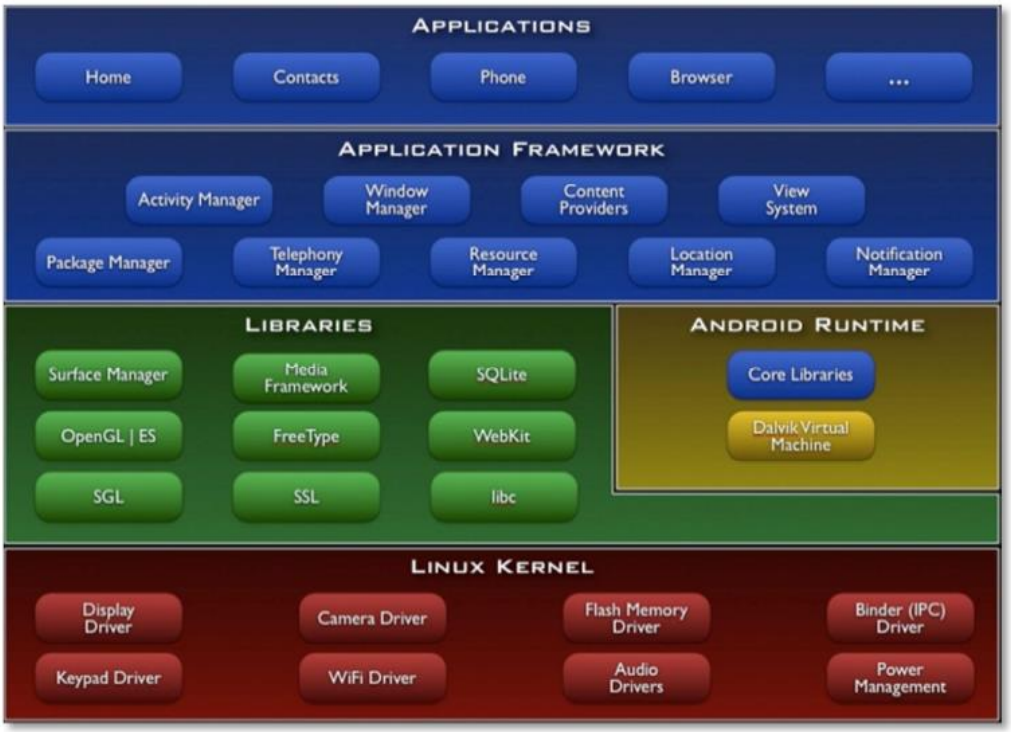


Ilustración 8. Arquitectura de Android

Antes de estudiar la arquitectura de *Android* se harán algunas aclaraciones sobre Java. *Java* es un lenguaje orientado a objetos que fue diseñado para funcionar de forma independiente a la arquitectura sobre la que trabaja. Para ello, tal y como se puede observar en la ilustración siguiente, la compilación del código fuente (archivos .java) genera archivos de clases (archivos *.class*), es decir *bytecode*, los cuales serán ejecutados por la máquina virtual JVM (*Java Virtual Machine*), utilizada como abstracción entre el *hardware* de la máquina y los programas *Java*. Los distintos archivos *class* suelen ser compilados en un único archivo JAR (*Java Archive*). En *Android* sucede algo parecido (Android Developers, 2014).

La *Open Handset Alliance* optó por diseñar un sistema similar teniendo en cuenta las limitaciones de los dispositivos móviles, en principio una baja capacidad de almacenamiento y procesamiento y poca potencia de cálculo.

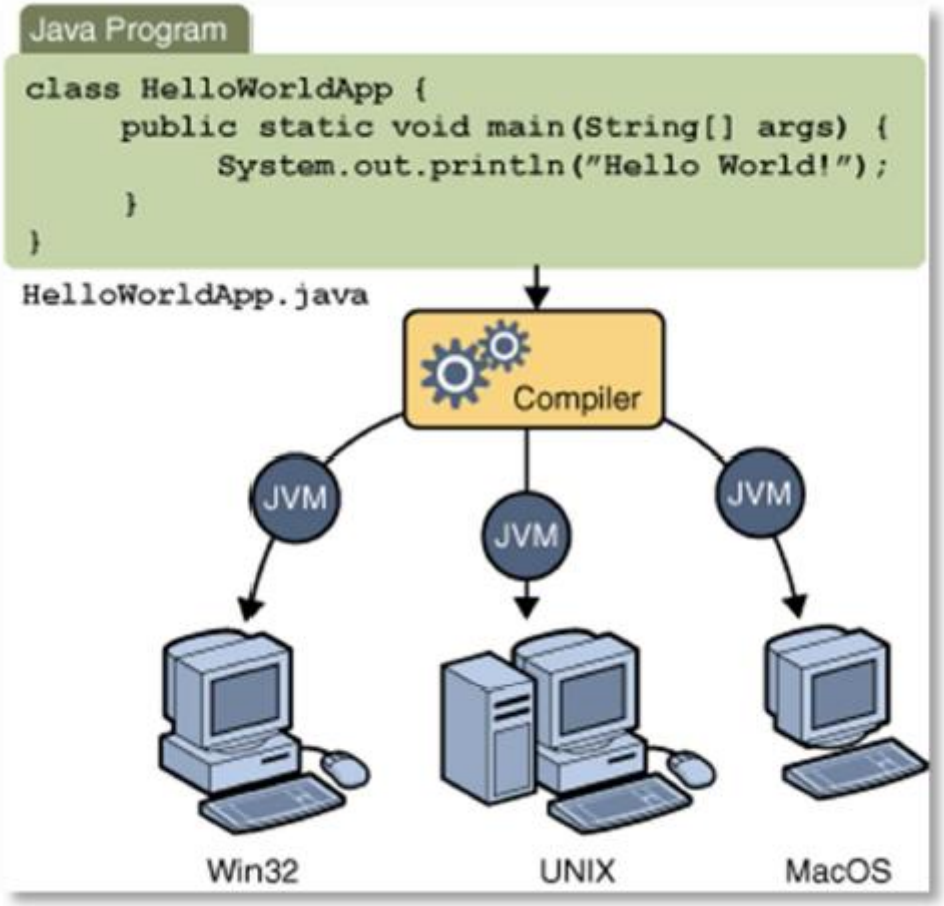


Ilustración 9. JAVA: lenguaje multiplataforma

La máquina virtual empleada por Android recibe el nombre de DVM (*Dalvik Virtual Machine*). Los programas generalmente son escritos en Java y compilados a *bytecode*. Posteriormente son convertidos a archivos compatibles con la DVM (*archivos*.dex) antes de su instalación en el dispositivo. Estos archivos son comprimidos en archivos APK (*Application Package*), los cuales pueden ser instalados en dispositivos Android compatibles (a la hora de crear una aplicación es necesario especificar la versión para la que está diseñada por lo que el terminal donde se instalará ha de tener una versión de Android igual o superior). En la ilustración siguiente se puede observar cómo se crean los archivos con extensión *dex* a partir de los archivos de clase class. La principal diferencia radica en la forma de *empaquetar* la información. Los archivos de clase son transformados en un único archivo *.dex* en el que se intenta despreciar la información repetida. Es más que evidente el enfoque de Android hacia dispositivos con memoria pequeña y almacenamiento similar.

Otro factor importante y a tener en cuenta es que *Android* implementa algunas de las *APIs* de *Java* (Oracle and/or its affiliates, 2014) , pero no todas ellas (Android Developers, 2014).

Consecuentemente, *Android* y *Java* no son el mismo sistema, aunque tengan muchos puntos en común.

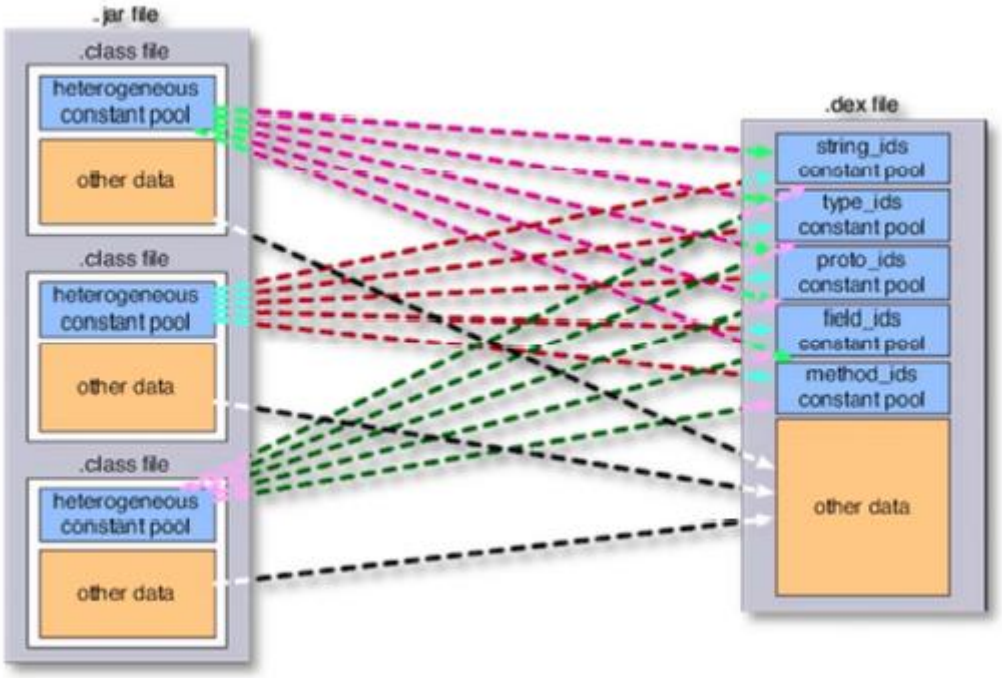


Ilustración 10. Creación de archivos *.dex* a partir de archivos *.class*

A continuación se exponen las principales características de la arquitectura Android:

• *Applications*: Android se entrega conjuntamente con una serie de aplicaciones que incluyen un cliente de email, un programa gestor de SMS, calendario, mapas, explorador y gestor de contactos entre otros. Todas ellas están escritas en *Java*.

• *Application Framework*: los desarrolladores de *Android* tienen acceso a las mismas *APIs* del *framework* usadas por las aplicaciones del núcleo. La arquitectura está diseñada con el objetivo de simplificar la reutilización de componentes. Por debajo de todas las aplicaciones hay un conjunto de servicios y sistemas entre los que se incluyen:

- Conjunto de *Views* que pueden ser utilizadas para construir la aplicación en cuestión: listas, tablas, cajas de texto…
- Proveedores de contenido que permiten compartir información entre aplicaciones, como por ejemplo la destinada a gestionar los contactos.
- Un *Resource Manager* capaz de proporcionar acceso a recursos como cadenas de texto, imágenes, *layouts*…
- Un *Notification Manager* que permite gestionar las notificaciones en pantalla.
- Un *Activity Manager* que gestiona el ciclo de vida de las aplicaciones.

• *Libraries*: *Android* incluye un conjunto de librerías escritas en C/C++ que son usadas por el sistema *Android*. Algunas de estas librerías son: *System* C *library*, *Media libraries*, *Surface Manager*…

• *Android Runtime*: cada aplicación de *Android* es un proceso diferenciado que tiene su propia instancia de la máquina DVM (*Dalvik Virtual Machine*). La máquina virtual *Dalvik* se sustenta en el *Linux Kernel* para proporcionar funcionalidades como la gestión de *threads* y de la memoria a bajo nivel.

• *Linux Kernel*: Android se basa en la versión 2.6 de Linux para servicios del núcleo de sistema como seguridad, gestión de memoria, gestión de drivers…El núcleo Linux también es usado como capa de abstracción entre el *hardware* y el resto del *software*.

Por su parte, las principales ventajas de *Android* son las siguientes:

• Uso del lenguaje de programación *Java*. Este lenguaje se encuentra ampliamente extendido y su comprensión si se desconoce es fácil para desarrolladores con conocimientos de programación orientados al objeto.

• *Software* libre: gracias a la licencia *Apache* en la que se basa *Android*, cualquier programador puede realizar modificaciones de las partes más internas de cualquier programa. Además, cualquier persona puede realizar un programa para dispositivos *Android* sin necesidad de cuotas anuales como sucedía con *iOS*.

• Accesibilidad: la participación de terceros en el desarrollo del sistema operativo conlleva la aparición y creación de multitud de *APIs*.

• Al igual que sucedía con *iOS*, existe una gran comunidad de desarrolladores gracias a la cual es sencillo encontrar información para programadores noveles en la red.

En cuanto a las desventajas se pueden citar las siguientes:

• Fragmentación: *Android* ha sido criticado múltiples veces a causa de la inmensa variedad de terminales que soportan el sistema operativo. Este hecho no es en sí mismo negativo, pero la diferencia de *hardware* existente entre sus terminales y la influencia de las operadoras móviles y de los fabricantes en el ritmo de llegada de las actualizaciones de *Android* a los terminales, hacen que exista una diversidad enorme de versiones del sistema operativo utilizados por los clientes. Existen numerosas y distintas versiones del sistema operativo de *Google*. Esto supone un gran inconveniente para los desarrolladores de *software*, obligados a revisar sus aplicaciones cada vez que *Android* saca al mercado una nueva versión.

• Aplicaciones: Google no supervisa las aplicaciones que se suben a Google Play, por lo que suele haber un mayor número de aplicaciones no demasiado útiles en el mercado.

• La gestión multitarea. El sistema de *Google,* a diferencia de *iOS*, sí que implementa un multitarea real de cara al usuario. Si una aplicación pasa a segundo plano, ésta se sigue

ejecutando en el procesador del terminal. El fallo consiste en que si el usuario tiene muchas tareas abiertas, el dispositivo se ralentizará.

En cuanto al entorno de desarrollo, programar para *Android* tiene bastantes menos requisitos que hacerlo para *iOS*. Prueba de ello es la posibilidad de elección del sistema sobre el que se quiere programar (*Unix, Windows, Mac OS,…*) y del entorno de desarrollo a utilizar. Existen varias opciones, entre las que la más destacada es sin duda alguna *Eclipse*.

3.1.1.2 Apple iOS

*iOS (*anteriormente denominado iPhone OS) es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone, siendo después usado en el *iPod touch* e *iPad*. Es un derivado de *Mac OS* X, que a su vez está basado en Darwin (Sistema Operativo). *El iOS tiene 4* capas de abstracción: la capa del núcleo del sistema operativo, la capa de *"Servicios Principales",* la capa de *"Medios de comunicación"* y la capa de *"Cocoa Touch"*

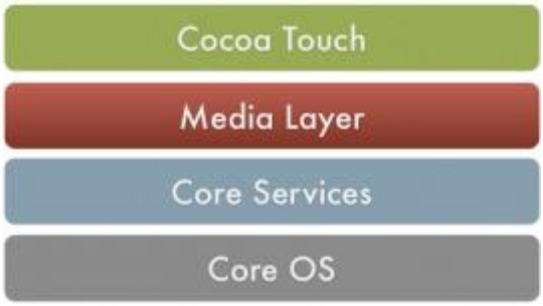


Ilustración 11. Arquitectura de iOS

A día 30 de marzo de 2015, el 78% de los dispositivos *iPhone*, *iPod touch* o *iPad* tienen instalado actualmente la versión *iOS* 8, según el blog Actualidad iPhone (Actualidad iPhone, 2015).

Originalmente *Apple* desarrolló el sistema operativo *iOS* para su teléfono inteligente *iPhone*. *Apple* reveló por primera vez la existencia de *iOS*, por aquel entonces llamado *iPhone OS*, en la *MacWorld Conference* el 9 enero del 2007, aunque el sistema no tuvo nombre oficial hasta marzo del 2008, fecha en la que salió la primera versión beta del *iPhone SDK (Start Development Kit)*. Como se ha mencionado el sistema original estaba diseñado para funcionar en los *iPhones* de *Apple*, pero lo emplean otros de sus productos como son el *iPod Touch*, el *iPad* y *Apple TV*. *Apple* no permite la instalación de iOS en hardware de terceros.

En junio del 2010 *Steve Jobs, CEO de Apple*, anunció que *iPhone OS* pasaría a llamarse *iOS*. *iOS* está basado en el sistema operativo *Mac OS X* , que a su vez está basado en *Darwin BSD* y por lo tanto es un sistema operativo *Unix*. En concreto ambos sistemas operativos comparten el mismo núcleo *Mach/FreeBSD*, y utilizan como lenguajes de programación principales *C* y *Objective-C* (Macprogramadores, 2013).

El sistema Unix es el utilizado en publicaciones de Linux, así que *iOS*, *OS* X y *Linux*, guardan más similitudes de las que nos podemos imaginar, tan solo que los dos primeros son sistemas operativos propiedad de *Apple* y cerrados al uso en dispositivos de la propia compañía, mientras que *Linux* es un código abierto y valido para multitud de dispositivos, abierto a implementaciones y al uso e inclusión en los dispositivos y marcas que lo consideren.

El lenguaje de programación C es el predecesor de *Objetive*-C y ha sido el principal lenguaje estructurado y utilizado en gran cantidad de entornos diferentes. C nació a principios de los años 70 en AT&T *Bell Laboratories* a manos de Dennis Ritchie.

Las aplicaciones nativas para *OS X* e *iOS* se programan con el lenguaje *Objective-C*. Se trata de un lenguaje orientado a objetos, muy dinámico y en constante mejora por parte de *Apple*. También cuenta con otras características que lo hacen muy robusto, como es su compatibilidad con *C/C++* (Guapu Technologies, S.L., 2013).

La interfaz de usuario de *iOS* está basada en el concepto de manipulación directa, usando gestos multitáctiles (eventos *multi-touch*). Los elementos de control consisten de deslizadores, interruptores y botones. La respuesta a las órdenes del usuario es inmediata y provee de una interfaz fluida. La interacción con el sistema operativo incluye gestos como deslices, toques, pellizcos, los cuales tienen definiciones diferentes dependiendo del contexto de la interfaz. Se utilizan acelerómetros internos para hacer que algunas aplicaciones respondan a sacudir el dispositivo o rotarlo en tres dimensiones (Petrazzini, 2012).

Para entender mejor la arquitectura de *iOs* se debe comenzar por saber que la capa *Cocoa Touch* es la de más alto nivel y la de *Core OS* la de más bajo nivel. En general, las dos capas superiores ofrecen servicios y tecnologías más sofisticadas que el resto de capas. Los detalles de cada una de ellas son:

- Cocoa Touch: es un conjunto de *frameworks* orientados a objetos que permiten el desarrollo de aplicaciones nativas para *iOS*. El lenguaje utilizado es por supuesto *Objective*-C. Cabe destacar la presencia de los siguientes servicios (Rodríguez Lorenzana, 2012).
    - Eventos *multi-touch.*
    - *Multi-tasking.*
    - Acelerómetro y giroscopio.
    - Jerarquía de vistas.
    - Localización e internalización: *framework* que permite adoptar diferentes idiomas y regiones sin necesidad de realizar cambios en el código del programa.
    - Soporte para cámara.
- Media*:* contiene servicios orientados a multimedia como:

- *OpenAl (Open Audio Library).*
- Mezcla de audio y grabación.
- Reproducción de video.
- Formatos de archivo de imágenes
- *Quartz*: *framework* para manipular gráficos 2D.
- *Core Animation: framework* para la visualización de datos.
- *OpenGL: framework* para manipular gráficos 3D.

❖ Core Services*:* recopila servicios básicos como pueden ser:

- *Networking*.
- Base de datos *SQLite*.
- *Core Location: framework* que permite un fácil acceso al GPS.
- *Threads.*
- *Core Motion.*

❖ Core OS: actúa como núcleo del sistema operativo. Controla el sistema de memoria virtual, los hilos, los ficheros del sistema, la red e interprocesa la comunicación con los marcos de la capa *Core Services*. En general rodea el entorno del *kernel*, los controladores y las interfaces básicas del sistema operativo (Universidad Carlos III, 2014).

- *TCP/IP*.
- *Sockets*.
- Gestión de la batería.
- *File System*: sistema de archivos.
- Seguridad

3.1.1.3 Windows Phone

Windows Phone (abreviado *WP*) es un sistema operativo móvil desarrollado por Microsoft, como sucesor de Windows Mobile. A diferencia de su predecesor está enfocado en el mercado de consumo en lugar de en el mercado empresarial. Con *Windows Phone*; Microsoft ofrece una nueva interfaz de usuario que integra varios de sus servicios propios como *OneDrive*, *Skype* y *Xbox* Live en el sistema operativo. Compite directamente contra *Android* de *Google* e *iOS* de *Apple*.

Debido a la evidente fragmentación de sus sistemas operativos, Microsoft anunció en enero de 2015 que dará de baja a *Windows Phone*, para enfocarse en un único sistema más versátil denominado *Windows* 10, disponible para todo tipo de plataformas (teléfonos inteligentes, tabletas y computadoras). Actualmente el sistema operativo *Windows Phone* 8, es el tercero en el ranking de los más usados por detrás de *Android* e *iOS*.

Próximamente podrán empezar las migraciones al nuevo sistema operativo *Windows* 10.

El hecho de que Windows *Phone* 8 comparta núcleo con Windows 8 hace que sus modelos de aplicación tengan similitudes y nos sea sencillo compartir código entre ambas plataformas. Todo esto nos da una idea del potencial con el que contaremos a la hora de desarrollar nuestras aplicaciones en Windows *Phone*.

Para poder desarrollar en este sistema operativo, no es necesario tener una cuenta propia para descargarse el *SDK*, pero sí se requiere para desbloquear nuestro terminal móvil y para publicar nuestras apps en el *Store* de *Windows Phone*. Existen las siguientes opciones para adquirir una cuenta de desarrollador (Microsoft, 2015):

- Incluida si tienes una suscripción MSDN.
- Gratis para estudiantes con la suscripción *Dreamspark*.
- $99 al año para desarrolladores individuales.

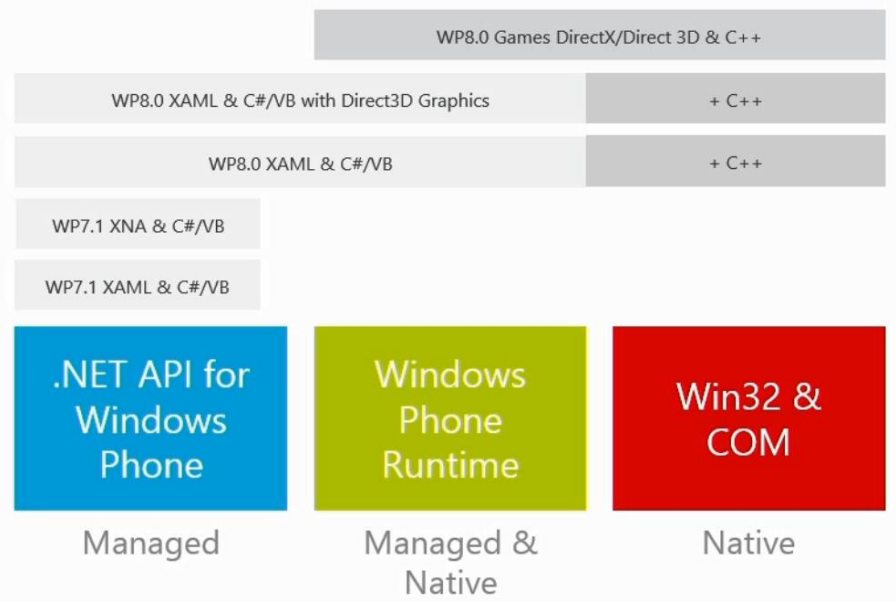


Ilustración 12. Arquitectura Windows Phone

Windows Phone 8 proporciona las *APIs* que se muestran en la ilustración anterior:

- .NET API para *Windows Phone*: Las aplicaciones dirigidas únicamente a Windows Phone 8 utilizan, además de *.NET API* para *Windows Phone*, el *Windows Phone Runtime*.

- *Windows Phone Runtime*: esta API es una versión reducida de *WinRT*, el conjunto de APIs usadas en Windows 8. El Windows *Phone Runtime*, además, añade nuevas clases con funcionalidad específica para el dispositivo móvil como la síntesis de voz, personalización de la pantalla de bloqueo, información detallada del teléfono y muchas más. Si nuestra intención es desarrollar una aplicación conjuntamente en Windows *Phone* y Windows 8, la mejor opción será usar el Windows *Phone Runtime* para una mayor posibilidad de compartir código.

- *Win32* & *COM*: principalmente es de interés para desarrolladores que usan código nativo. Provee clases para uso de redes a bajo nivel, *APIs* de cámara y otras funciones avanzadas que normalmente no se necesitarán en nuestras aplicaciones.

3.1.1.4 Blackberry RIM

El BlackBerry OS es un sistema operativo móvil de código cerrado desarrollado por BlackBerry, antigua *Research* In *Motion* (RIM); para los dispositivos BlackBerry. El sistema permite multitarea y tiene soporte para diferentes métodos de entrada adoptados por RIM para su uso en computadoras de mano, particularmente la *trackwheel*, *TrackBall*, *touchpad* y pantallas táctiles. Su desarrollo se remonta la aparición de los primeros *handheld* en 1999.

El SO *BlackBerry* está claramente orientado a su uso profesional como gestor de correo electrónico y agenda. Desde la cuarta versión se puede sincronizar el dispositivo con el correo electrónico, el calendario, tareas, notas y contactos de *Microsoft Exchange Server* además es compatible también con Novell *GroupWise* (Hernandez Vargas, 2014).

Se caracteriza por ser un sistema operativo atractivo y algo diferente a Android e iOS, además, sus iconos son sencillos. Casi todas sus aplicaciones piden permisos para sincronizarse con BlackBerry Messenger, y es que esta aplicación es el centro del teléfono. El *kernel* del sistema operativo *Blackberry* está basado en Java y posee una arquitectura *ARM*. Se caracteriza por ser bastante ligero y funcionar con fluidez. Mediante dos pestañas permita separar la vida personal de la laboral, la seguridad en este dispositivo es una prioridad.

Uno de los inconvenientes que actualmente ha hecho que el uso de este sistema operativo haya disminuido tanto es que su tienda de aplicaciones, *BlackBerry World*, no cuenta con todas las aplicaciones que a los usuarios les gustaría. Además de esto, sus escasas actualizaciones y el retraso en sacar nuevos dispositivos móviles, en comparación con otras tecnologías ha hecho que hoy en día, el porcentaje de dispositivos que usan *Blackberry* sea de apenas un 10% (Gruman, 2015).

3.1.1.5 Comparativa entre ambos

Se ha observado que aunque iOS 8 goza de una presencia casi absoluta entre los usuarios de Apple, es cierto que su ritmo de crecimiento es menor que en anteriores ocasiones. Se trata de un sistema que no empezó con buen pie y debido a la merma de rendimiento en los dispositivos más antiguos, muchos son los que siguen en *iOS* 7 sin intención de actualizar a corto plazo.

A pesar de esta situación negativa dentro del ecosistema de *Apple*, llama la atención que Android *Lollipop*, con casi un año de antigüedad, tiene una tasa de adopción del 3,3%, siendo Android 4.4 *KitKat* la más utilizada con casi un 41%. También es cierto que los dispositivos que no tienen acceso a la tienda de aplicaciones de Google no cuentan en esta estadística (gama *Kidle Fire*, dispositivos con tiendas de terceros, etc.) pero aun así, es muy difícil mejorar esa cifra de *Lollipop*.

Windows *Phone* continúa quedándose corto en la selección de apps y las que tiene son más limitadas que las que las versiones para *Android* o *iOS*.

Windows *Phone* 8.1 es la mejor actualización que hemos visto del sistema operativo móvil de Microsoft, gracias a mejoras en su diseño y funciones modernas. Sin embargo, una de las novedades más importantes, *Cortana*, el asistente virtual por el cuál más se caracteriza la nueva actualización, sigue quedándose por detrás de Android o iOS.

3.1.2 Aplicaciones web móvil

Una aplicación web o *webapp* es la desarrollada con lenguajes muy conocidos por los programadores, como es el HTML, *Javascript* y *CSS*. La principal ventaja con respecto a las nativas, vistas en el apartado anterior, es la posibilidad de programar independiente del sistema operativo en el que se usará la aplicación. De esta forma se pueden ejecutar en diferentes dispositivos sin tener que crear varias aplicaciones.

Las aplicaciones web se ejecutan dentro del propio navegador web del dispositivo a través de una URL. Por ejemplo en Safari, si se trata de la plataforma iOS. El contenido se adapta a la pantalla adquiriendo un aspecto de navegación APP.

Cuando me planteé el hecho de crear la aplicación me surgió la duda de que tecnología escoger, pero lo que mi tutora y yo tuvimos claro es que no escogeríamos una *webapp* puesto que hoy en día no todo el mundo considera estas *webapps* realmente como puras aplicaciones móviles. Existen grandes diferencias con una aplicación nativa o híbrida como por ejemplo las mostradas en la siguiente tabla, y una más y es que no necesita instalación por lo que no pueden estar visibles en app store y la promoción y comercialización debe realizarse de forma independiente.

Las apps web móviles serán una mejor opción si nuestro objetivo fuera adaptar la web a formato móvil (Lance Talent, 2015).

Ventajas	Inconvenientes
• El mismo código base reutilizable en múltiples plataformas • Proceso de desarrollo más sencillo y económico • No necesitan ninguna aprobación externa para publicarse (a diferencia de las nativas para estar visibles en app store) • El usuario siempre dispone de la última versión • Pueden reutilizarse sitios "responsive" ya diseñados	• Requiere de conexión a internet • Acceso muy limitado a los elementos y características del hardware del dispositivo • La experiencia del usuario (navegación, interacción..) y el tiempo de respuesta es menor que en una app nativa • Requiere de mayor esfuerzo en promoción y visibilidad

Tabla 3. Ventajas y desventajas aplicaciones web

3.1.3 Desarrollo de aplicaciones híbridas

Una aplicación híbrida es una combinación de las dos anteriores, se podría decir que recoge lo mejor de cada una de ellas. Las apps híbridas se desarrollan con lenguajes propios de las *webabpp*, es decir, HTML, *Javascript* y *CSS* por lo que permite su uso en diferentes plataformas, pero también dan la posibilidad de acceder a gran parte de las características del hardware del dispositivo. La principal ventaja es que a pesar de estar desarrollada con HTML, Java o CSS, es posible agrupar los códigos y distribuirla en app store.

PhoneGap es uno de los *frameworks* más utilizados por los programadores para el desarrollo multiplataforma de aplicaciones híbridas (Lance Talent, 2015).

A su vez dan la posibilidad de acceder a las características del hardware del dispositivo, es decir que se puede utilizar una gran parte o todas las utilidades y sensores del dispositivo.

Además de las ventajas y desventajas que se observan en la tabla inferior, puntualizaré las tres ventajas que más me han gustado sobre este tipo de aplicaciones:

• Son aplicaciones multiplataforma, permiten que el código fuente se pueda ejecutar en diferentes plataformas.

• Estas aplicaciones permiten acceder a todos los recursos del móvil, como pueden ser la cámara o el sistema GPS.

• No es necesaria conexión a internet para ejecutar la aplicación a excepción de partes concretas de la aplicación que requieran dicha conexión.

Ventajas	Inconvenientes
• Es posible distribuirla en las tiendas de iOS y Android. • Instalación nativa pero construida con JavaScript, HTML y CSS • El mismo código base para múltiples plataformas • Acceso a parte del hardware del dispositivo	• Experiencia del usuario más propia de la aplicación web que de la app nativa • Diseño visual no siempre relacionado con el sistema operativo en el que se muestre

Tabla 4. Ventajas y desventajas aplicaciones híbridas

El proceso de desarrollo para las aplicaciones híbridas es más complicado que para las aplicaciones web. Del mismo modo que para el desarrollo de aplicaciones web, se crean archivo *HTML*, *CSS* y *JavaScript* a ejecutar en un navegador. Pero también al igual que para las aplicaciones nativas, el código generado se compila en un ejecutable. Ambos códigos son compilados y forman un paquete que será compartido mediante los *market* places (Glera Aransay, 2013).

La mayor parte de la infraestructura es web y la comunicación con las herramientas del terminal se lleva a cabo mediante comunicadores o herramientas multiplataforma como *PhoneGap* que es uno de los *frameworks* más utilizados por los programadores para el desarrollo de aplicaciones híbridas.

Hoy día existen muchos *frameworks* y tecnologías basadas en HTML5 y *Javascript* que permiten programar interfaces de uso en los móviles que casi igualan la experiencia de usuario de aplicaciones nativas. A continuación se van a mencionar algunos de los múltiples *frameworks* que existen hoy en día.

3.1.3.1 phoneGap

PhoneGap es un interesante *framework* para el desarrollo de aplicaciones móviles propiedad de Adobe, capaz de permitirnos crear simultáneamente una misma aplicación móvil para varios sistemas operativos. Está basado en el proyecto *open source* de Apache *Cordova*. *PhoneGap* fue liberado como un proyecto de la fundación Apache y renombrado como *Cordova*. Fue creado por *Nitobi* y ésta fue comprada por Adobe. *PhoneGap* es una librería creada en JavaScript, la cual agrupa las principales interacciones con el hardware de los teléfonos móviles utilizando los estándares HTML5, CSS3 y JavaScript (García Echegaray, 2014).

Phonegap trata de solventar el problema de decidir la plataforma con la que trabajar cuando se comienza a crear una aplicación móvil. Que además también supone elegir el lenguaje de programación, ya que cada plataforma utiliza un lenguaje.

En un principio fue desarrollado por *Nitobi* y su uso era gratuito. En 2011 Adobe anunciaba la adquisición de *Nitobi*, por lo tanto *PhoneGap* pasaba al control de Adobe. De modo que ha sido integrado en las últimas versiones de Dreamweaver. Se armó un gran revuelo por el temor de que se abandonara la gratuidad, pero el código fue entregado a la Fundación Apache y así *PhoneGap* continuaba siendo un software libre.

Actualmente este proyecto en la Fundación Apache recibe el nombre de "*Apache Cordova*" aunque se sigue manteniendo *PhoneGap* como una especie de marca comercial y por ello se sigue conociendo al *framework* como *PhoneGap*.

Phonegap permite convertir casi todos los códigos basados en tecnologías web (HTML, CSS y *JavaScript*) en código con apariencia de nativo, preparado para compilar en el SDK. Evitando al programador el largo proceso de aprendizaje de lenguajes específicos de programación y facilitando el desarrollo y mantenimiento de la aplicación. El producto final de cada aplicación es un archivo binario *(IPA, APK, XAP, etc.*) listo para ser distribuido en las diferentes tiendas o *markets* propios de cada sistema (Adobe Systems Inc, 2015).

Ilustración 13. Empaquetamiento aplicaciones

PhoneGap empaqueta las aplicaciones web dentro de una aplicación nativa, de cualquiera de las plataformas soportadas (*Android*, iOs, *Blackberry*, *Windows Phone*, *Web Os*, *Symbian*), de modo que parece una aplicación nativa, pero no es así, se trata de una aplicación híbrida.

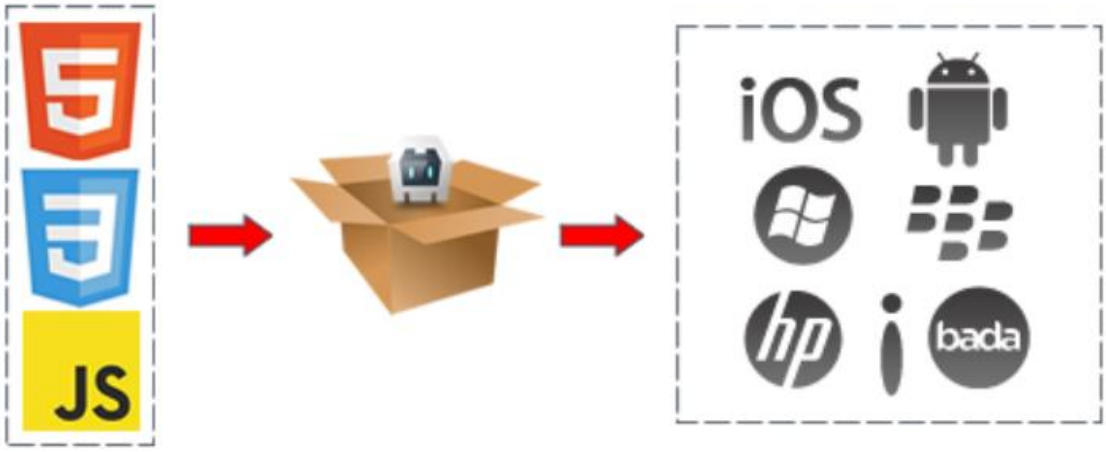


Ilustración 14. Empaquetamiento de las aplicaciones en PhoneGap

PhoneGap consiste en un conjunto de APIs (interfaz de programación de aplicaciones), basadas en JavaScript que permiten al desarrollador acceder a los elementos nativos del dispositivo, como la cámara, el acelerómetro, los contactos, el GPS, etc. El uso de estas librerías se combina con un *framework* de desarrollo web móvil como puede ser *jQuery Mobile* o *Sencha Touch*, de este modo es posible desarrollar aplicaciones que accedan a partes nativas del dispositivo utilizando únicamente HTML, CSS y JavaScript (Glera Aransay, 2013).

Estas *APIs* permiten desarrollar la aplicación sin utilizar ningún lenguaje específico de cada plataforma como Java u Objetive C. Es por esto que *PhoneGap* es cada vez más popular entre los desarrolladores y sobre todo entre los primerizos que se estrenan en el diseño e implementación de aplicaciones móviles, debido a sus grandes ventajas, su facilidad y comodidad (Adobe Systems Inc, 2015).

En este TFG, me he decantado por el uso de *PhoneGap* puesto que era la tecnología que me enseñaron a usar en la empresa donde realicé las practicas curriculares del grado, y tras trabajar con ella durante los 3 meses que duró la práctica, encontré que sus ventajas, tanto de uso como de distribución, iban a facilitar mucho la labor de desarrollo de mi aplicación.

	iPhone / iPhone 3G	iPhone 3GS and newer	Android	Blackberry OS 6.0+	Blackberry 10	Windows Phone 8
Accelerometer	✓	✓	✓	✓	✓	✓
Camera	✓	✓	✓	✓	✓	✓
Compass	X	✓	✓	X	✓	✓
Contacts	✓	✓	✓	✓	✓	✓
File	✓	✓	✓	✓	✓	✓
Geolocation	✓	✓	✓	✓	✓	✓
Media	✓	✓	✓	X	✓	✓
Network	✓	✓	✓	✓	✓	✓
Notification (Alert)	✓	✓	✓	✓	✓	✓
Notification (Sound)	✓	✓	✓	✓	✓	✓
Notification (Vibration)	✓	✓	✓	✓	✓	✓
Storage	✓	✓	✓	✓	✓	✓

Ilustración 15. Acceso a recursos por sistema operativo mediante *PhoneGap*.

3.1.3.2 jQuery Mobile

El *framework* utilizado para desarrollar el código base de la aplicación de pediatría es *jQuery* Mobile. Permite que las aplicaciones creadas tengan la misma apariencia independientemente del terminal mediante el que se acceda. No se trata de un *framework* desarrollado desde cero, se puede entender como un *pluging* para *jQuery* ya que se basa en dicho *framework* JavaScript. Del mismo modo que *Jquery*, esta versión móvil también se basa en el lema expuesto en página web: "*Write Less*, *Do More*", lo que quiere decir:" Escribe menos, haz más" (The jQuery Foundation, 2015).

No solo consiste en un *framework* para desarrollar código JavaScript que pueda ser ejecutado en varios navegadores, la novedad es que presente varias herramientas que facilitan el proceso de desarrollar aplicaciones web. Las características básicas de *jQuery Mobile* son las siguientes (Glera Aransay, 2013):

- Es simple de utilizar, desarrollando poco código es posible crear aplicación muy usables y atractivas.
- *Jquery* Mobile hace uso de las últimas tecnologías web: HTML5, CSS3 y JavaScript.
- Presenta herramientas CSS que permiten que por ejemplo los formularios o las listas se estilicen de forma automática.

- Está preparado para los dispositivos táctiles.
- Presenta muchos automatismos, por ejemplo siempre que pueda realizar conexiones Ajax de forma automática y las transiciones entre páginas también son automáticas.
- Es un *framework* con un tamaño relativamente pequeño, aproximadamente unos 12Kb de las bibliotecas JavaScript, 6Kb de CSS y algunas imágenes.
- También proporciona una serie de temas para personalizar la aplicación.
- Compatible con varios sistemas operativos:

Ilustración 16. Sistemas operativos soportados por *jQuery Mobile*

Una herramienta muy útil que posee *jQuery Mobile* y que he usado para darle el estilo que más me gustaba a mi aplicación de pediatría ha sido el *themeroller* (jQuery Mobile, 2015).

Una herramienta que permite probar diferentes temas y colores en distintos estilos y simular como quedarían en nuestra futura aplicación. Me resulto muy interesante puesto que es muy sencilla de usar y facilita mucho la elección del fondo, tipografías, botones, etc.

Después de haber elegido los tres diseños que más gustan para nuestra aplicación, esta herramienta permite descargar la plantilla que hemos creado, obteniendo un archivo CSS que introduciremos en nuestro código para que genere el tema creado.

3.1.3.3 LungoJS

Lungo.JS es un *framework* para el desarrollo de aplicaciones móviles que aplica lo mejor de las capacidades de HTML5, CSS3 y *Javascript* para desarrollar rápidamente aplicaciones poderosas (CreativeCommons, 2014).

Las características que ofrece son:

- Diseño y desarrollo de aplicaciones para iOS, Android, Blackberry y WebOS.
- Diseñado para tomar ventaja de las características de los dispositivos móviles actuales.
- Captura eventos como el Pase, Toque, Doble Toque.
- Distribuir su aplicación en “Tiendas Móviles” o en sitios web.
- No hay necesidad de imágenes, todo es vectorial.
- Implementar las características de HTML5 como WebSQL, Ubicación Geográfica, Historia, Dispositivos de Orientación y más.
- No requiere servidor web.

- Totalmente personalizable.
- Construcción de aplicaciones, juegos, catálogos

A diferencia de otros *frameworks* como *jQuery Mobile* o Sencha se centra en dispositivos móviles más recientes no manteniendo compatibilidad para dispositivos que dentro de unos meses no tendrán demasiado uso. Esto lo hace bastante ligero y nos permite centrarnos fácilmente en este tipo de desarrollo. La curva de aprendizaje es realmente rápida, sobre todo echando un vistazo a las explicaciones y vídeos que el autor ha puesto a nuestra disposición en la red.

3.1.3.4 Xamarin

Xamarin, consiste en una implementación libre de la plataforma de desarrollo .NET para dispositivos Android, iOS y GNU/Linux. Es decir, con *Xamarin* se puede evitar tener que utilizar *Java* para una aplicación para Android.

Xamarin permite generar una aplicación para iOS (.APP) y para Android (.APK), la cual ya sí correrá de forma nativa. Gracias a esto, surge una de las grandes ventajas de *Xamarin*: la reutilización de código. Se basa en utilizar C# y .NET para la compilación de aplicaciones nativas para Android o iOS.

Uno de los detalles más importantes es que *Xamarin* nos proporciona acceso total a la API estándar de Android (Zamora, 2015).

El SDK de *Xamarin* es una abstracción de las librerías nativas, por lo que al desplegar la aplicación en una plataforma concreta, se genera código 100% nativo. No obstante, no todo el código de la aplicación desarrollada puede ser compartido para todas las plataformas, la interfaz de usuario debe ser específica para cada plataforma. En este caso, las plataformas soportadas son iOS, Android y Windows Mobile.

3.1.3.5 Rhodes

Rhodes es un *framework* de código libre para el desarrollo de aplicaciones multiplataforma basado en Ruby.

Sirve para la creación de forma rápida de aplicaciones nativas para la mayoría de sistemas operativos de los *smartphones* actuales. Soporta GPS, geolocalización y captura de imágenes con la cámara. Permite una amplia variedad de soportes: BlackBerry, Android, iPhone, Windows Mobile, etc. Es una buena alternativa para quienes ya manejen el lenguaje Ruby (Ortiz de Zárate, 2011).

3.1.3.6 Sencha Touch

Sencha Touch es un conjunto de librerías (*framework*) que permite crear rápidamente aplicaciones web móviles para: iPhone, Android y BlackBerry, utilizando *APIs* de HTML5 y una singular estructura estilo JSON.

Sencha Touch quiebra las líneas entre aplicaciones web o nativas liberando una gran experiencia que lo usuarios disfrutan usando las aplicaciones móviles. Usando HTML5 y CSS3 *Sencha Touch* gana ventajas sobre la aceleración de hardware para liberar experiencia en las aplicaciones sin importar que navegador o que dispositivo móvil se esté usando.

Un gran inconveniente de *Sencha Touch* es que necesita una licencia dual. Aun así, *Sencha Touch* funciona perfectamente junto a *PhoneGap*, por lo que de esta manera se pueden distribuir las aplicaciones en la App Store o en Android Marketplace del uso de *PhoneGapBuild*. Gracias a esto se puede hacer uso de la API nativa del dispositivo para acceder a la lista de contactos, la cámara entre otras (Meléndez, 2012).

3.2 Elección a partir de una comparativa entre ambos tipos de aplicaciones.

La decisión que he tomado para la realización de una aplicación móvil orientada al campo de la pediatría ha sido la del diseño de una aplicación híbrida o multiplataforma mediante el *framework* de *jQuery Mobile* y *PhoneGap*. He escogido *PhoneGap* puesto que es uno de los *frameworks* más populares que soporta varias plataformas. Además permite la integración de otros *frameworks* como *jQuery Mobile*.

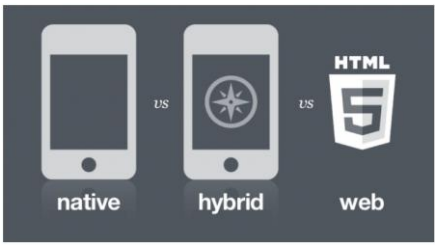


Ilustración 17. Comparación entre los distintos tipos de aplicaciones

Esta disposición la he llevado a cabo tras hacer una comparación de las diferentes tecnologías que actualmente existen en el mercado para el diseño e implementación de aplicaciones móviles y comprobar que una aplicación de este tipo, abarca un mercado mucho más extenso, a través del cual podremos llegar a un número mayor de personas, lo que equivale a más solidez, más competitividad y una mejor y más activa oferta en el mercado mundial de las aplicaciones móviles (Lance Talent, 2015).

Una aplicación híbrida recoge lo mejor de las nativas y de las web. Entre las ventajas más influyentes de este tipo de aplicaciones, destacaré:

- Uso de los recursos del dispositivo y del sistema operativo
- El costo de desarrollo puede ser menor que el de una nativa
- Son multiplataforma: Esto equivale directamente al ahorro de tiempo
- Permite distribución a través de las tiendas de su respectiva plataforma.

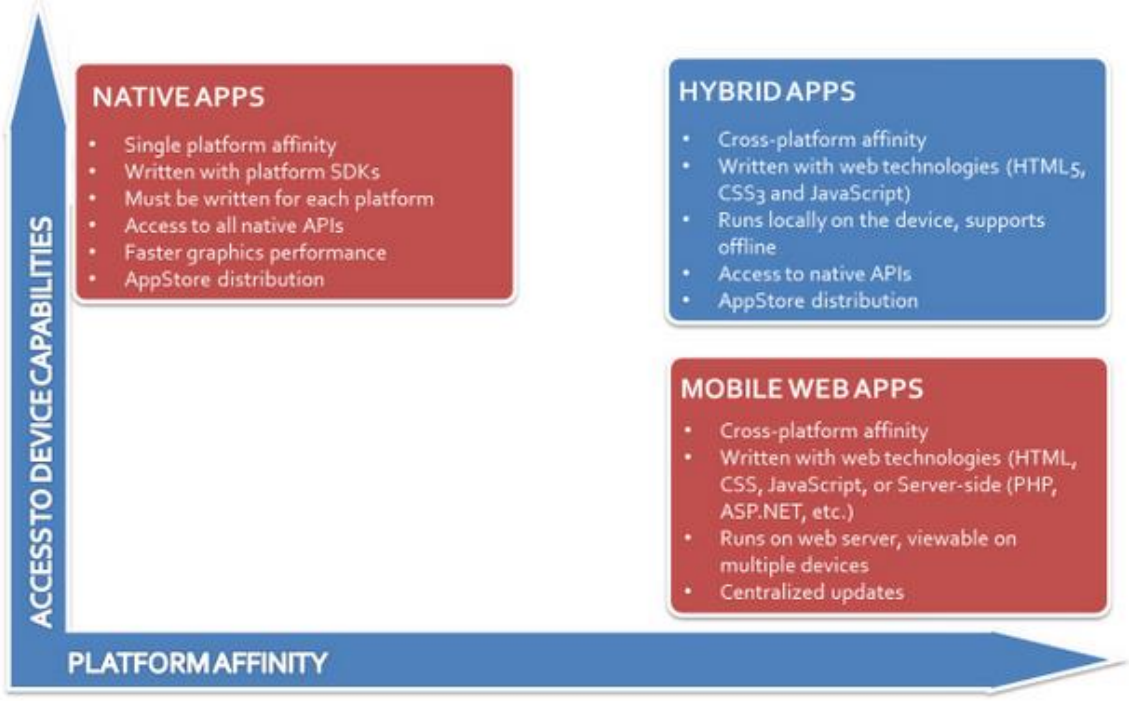


Ilustración 18. Ventajas e inconvenientes según la orientación de cada aplicación

La posibilidad de hacer una aplicación web, lo descarté inicialmente debido a que mayormente, la única ventaja que tienen es que son muy sencillas de implementar; pero a partir de ahí, en mi opinión comienzan las desventajas. Algunas son tan importantes como las que nombro a continuación (Pimienta, 2014):

- No pueden ser publicadas en plataformas para su distribución
- No utilizan los recursos del sistema ni del dispositivo de manera optima

Debido a estas dos carencias tan influyentes, descarte la posibilidad de diseñar una aplicación de este tipo.

Por último, explicaré los motivos que me llevaron también a descartar la implementación de una aplicación nativa. Este tipo de aplicaciones requieren de mayor tiempo de implementación, puesto que no son multiplataforma y deberá desarrollarse un código distinto para cada sistema operativo. Algunas otras desventajas que he encontrado relevantes han sido:

1. Solo pueden ser utilizadas por un dispositivo que cuente con el sistema para el cual fue desarrollada.
2. Requiere de un costo para distribuirla en una tienda, y dependiendo el sistema, para el uso del entorno de desarrollo.
3. Necesitan aprobación para ser publicadas en la plataforma.
    - El usuario deberá actualizar manualmente la aplicación desde los *market places*.

- A la hora de publicar la aplicación el desarrollador se enfrentará a los procesos de validación de los diferentes *market places*, algunos más duros que otros.
- Al tener que desarrollar específicamente para cada plataforma el tiempo de desarrollo y el coste se incrementarán.

Tras realizar la comparativa entre los tres tipos de aplicaciones, hemos observado que los tres presentan tanto ventajas como desventajas en los distintos aspectos estudiados.

Habitualmente, no existe una mejor o peor tecnología, pero habitualmente existirá a priori una solución que se acerque más a los requisitos que el desarrollador desea para su aplicación. Por este motivo, cada uno encontrará en cada solución la forma más adecuada de conseguir cubrir las necesidades de los potenciales usuarios a los que quiera llegar.

Como en este caso la función de la aplicación es facilitar el día a día de los padres de los niños y niñas que acuden al pediatra, se quiere que todos los usuarios la puedan utilizar independientemente del terminal que dispongan. Esto es posible gracias a la naturaleza multiplataforma y multidispositivo del lenguaje basado en web utilizado para desarrollar el código fuente de la aplicación.

Ha sido precisamente debido al inconveniente nombrado en el anterior punto 1," Solo pueden ser utilizadas por un dispositivo que cuente con el sistema para el cual fue desarrollada", por el cual a mi parecer, para una aplicación con la que deseo llegar al máximo número posible de usuarios, y para la cual dispongo de un tiempo limitado para su desarrollo, he concluido que es más rentable realizar una aplicación híbrida. Detallaré esto con más detalle en el capítulo dedicado al coste económico.

Además, se ha visto que una de las ventajas de las aplicaciones híbridas es la posibilidad de que sean instaladas en el dispositivo y disponer de un acceso directo en él. En este caso resultará muy útil a los usuarios ya que seguramente utilicen la aplicación en varias ocasiones y para ello simplemente tendrán que pulsar el botón de acceso directo, sin necesidad de memorizar la dirección e introducirla en el navegador. Gracias a este tipo de aplicación los usuarios podrán hacer uso de esta sin necesidad de una conexión a Internet, a excepción de aquellas funciones específicas que si requieran Internet.

3.3 Lenguajes de programación del lado del cliente y otros

En este apartado se complementa el análisis de las tecnologías usadas en el proyecto mediante una revisión de los lenguajes de programación del lado del cliente y otras tecnologías empleadas como AJAX.

3.3.1 HTML, HTML5

HTML es un lenguaje de descripción de hipertexto compuesto por una serie de comandos, marcas, o etiquetas, también denominadas *tags* que permiten definir la estructura lógica de un documento web y establecer los atributos del mismo (color del texto, contenidos multimedia, hipervínculos, etc).

Se puede resumir en que HTML es un lenguaje que permite crear páginas web y para ello hace uso de una serie de comandos o etiquetas que indican o marcan qué contenido se debe mostrar y de qué forma (Gauchat, 2013).

Estos comandos van siempre incluidos entre los signos < y > y se insertan en el propio texto que forma el contenido de la página. Especifican las distintas partes de la página, esto es, su estructura, y su formato. Adicionalmente, dan la posibilidad de insertar contenidos especiales como pueden ser imágenes, vídeos o sonidos, entre otros.

HTML5, actualización de HTML, es un término de marketing que agrupa las nuevas tecnologías web: HTML5, CSS3 y nuevas capacidades de JavaScript. La versión anterior, HTML4 carece de características necesarias para la creación de aplicaciones modernas.

El uso fuerte de JavaScript ha ayudado a mejorar, gracias a *frameworks* como *jQuery*, *jQuery* UI, *Sproutcore*, entre otros. Flash en especial ha sido usado en reemplazo de HTML para desarrollar web apps que superarán las habilidades de un navegador: Audio, vídeo, webcams, micrófonos, datos binarios, animaciones vectoriales, componentes de interfaz complejos, entre muchas otras cosas. Ahora HTML5 es capaz de hacer esto sin necesidad de *plugins* y con una gran compatibilidad entre navegadores.

Así pues, HTML 5 es una nueva versión de diversas especificaciones, entre las que se encuentran:

- HTML 4
- XHTML 1
- CSS Nivel 2
- DOM Nivel 2 (DOM = *Document Objetc Model*)

A la par, HTML 5 pretende proporcionar una plataforma con la que desarrollar aplicaciones web más parecidas a las aplicaciones de escritorio, donde su ejecución dentro de un navegador no implique falta de recursos o facilidades para resolver las necesidades reales de los desarrolladores. Para ello se están creando unas *APIs* que permitan trabajar con cualquiera de los elementos de la página y realizar acciones que hasta hoy era necesario realizar por medio de tecnologías accesorias.

Estas API, que tendrán que ser implementadas por los distintos navegadores del mercado, se están documentando con minuciosidad, para que todos los Browsers, creados por cualquier compañía las soporten tal cual se han diseñado. Esto se hace con la intención que no ocurra lo que viene sucediendo en el pasado, que cada navegador hace la guerra por su parte y los que acaban pagándolo son los desarrolladores y a la postre los usuarios, que tienen muchas posibilidades de acceder a webs que no son compatibles con su navegador preferido (Álvarez, 2012).

Novedades en HTML5:

HTML 5 incluye novedades significativas en diversos ámbitos. Como se decía, no sólo se trata de incorporar nuevas etiquetas o eliminar otras, sino que supone mejoras en áreas que hasta ahora quedaban fuera del lenguaje y para las que se necesitaba utilizar otras tecnologías (Álvarez, 2012).

- Estructura del cuerpo: La mayoría de las webs tienen un formato común, formado por elementos como cabecera, pie, navegadores, etc. HTML 5 permite agrupar todas estas partes de una web en nuevas etiquetas que representarán cada uno de las partes típicas de una página.
- Etiquetas para contenido específico: Hasta ahora se utilizaba una única etiqueta para incorporar diversos tipos de contenido enriquecido, como animaciones Flash o vídeo. Ahora se utilizarán etiquetas específicas para cada tipo de contenido en particular, como audio, vídeo, etc.
- *Canvas*: es un nuevo componente que permitirá dibujar, por medio de las funciones de un API, en la página todo tipo de formas, que podrán estar animadas y responder a interacción del usuario. Es algo así como las posibilidades que nos ofrece Flash, pero dentro de la especificación del HTML y sin la necesidad de tener instalado ningún *plugin*.
- Bases de datos locales: el navegador permitirá el uso de una base de datos local, con la que se podrá trabajar en una página web por medio del cliente y a través de un API. Es algo así como las Cookies, pero pensadas para almacenar grandes cantidades de información, lo que permitirá la creación de aplicaciones web que funcionen sin necesidad de estar conectados a Internet.
- *Web Workers*: son procesos que requieren bastante tiempo de procesamiento por parte del navegador, pero que se podrán realizar en un segundo plano, para que el usuario no tenga que esperar que se terminen para empezar a usar la página. Para ello se dispondrá también de un API para el trabajo con los *Web Workers.*
- Aplicaciones web Offline: Existirá otro API para el trabajo con aplicaciones web, que se podrán desarrollar de modo que funcionen también en local y sin estar conectados a Internet.
- Geolocalización: Las páginas web se podrán localizar geográficamente por medio de un API que permita la Geolocalización.

- Nuevas *API*s para interfaz de usuario: temas tan utilizados como el "*drag & drop*" (arrastrar y soltar) en las interfaces de usuario de los programas convencionales, serán incorporadas al HTML 5 por medio de un API.
- Fin de las etiquetas de presentación: todas las etiquetas que tienen que ver con la presentación del documento, es decir, que modifican estilos de la página, serán eliminadas. La responsabilidad de definir el aspecto de una web correrá a cargo únicamente de CSS.

Ilustración 19. Icono HTML5

Como se puede ver, existirán varios API con los que se trabajará para el desarrollo de todo tipo de aplicaciones complejas, que funcionarán online y offline. HTML5 ha sido un proyecto muy ambicioso que necesitó mucho tiempo para ser implementado.

3.3.2 CSS, CSS3

CSS u hojas de estilo en cascada (en inglés *Cascading Style Sheets*) es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML. El W3C (*World Wide Web Consortium*) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores. Es muy potente y flexible y está directamente conectado con HTML (Oros Cabello, 2002).

CSS permite generar un patrón de estilo que es aplicable a todos los documentos de una web, lo que supone un claro ahorro en el tiempo de desarrollo y en el de mantenimiento de una página. Además, esto permite organizar mejor el contenido al tener separada la estructura de una página de su diseño.

Las hojas de estilo facilitan el control de la presentación o estilo de las páginas web, y su posterior revisión y actualización en caso de ser necesarias, separando para ello la información relativa a la presentación de la relativa a la estructura. Además, las hojas de estilo representan

un avance importante para los autores de páginas web, al darles un mayor rango de posibilidades para controlar la apariencia de sus páginas (Navajas Ojeda, 2012).

Ilustración 20. CSS3

En los entornos científicos en los que la Web fue concebida, existía una mayor preocupación por la estructura de los documentos que por su presentación. A medida que la Web era descubierta por un espectro más amplio de usuarios de distintas procedencias, se vio la necesidad de superar las limitaciones estilísticas de HTML. Aunque las intenciones eran buenas, mejorar la presentación de las páginas web, algunas técnicas para conseguirlo tenían efectos secundarios negativos. Entre estas técnicas, que dan buenos resultados en algunas circunstancias, pero no siempre ni en todos los casos, se incluyen:

- Conversión del texto en imágenes, cuando el texto incorpora tipos de fuentes poco habituales.
- Utilización de imágenes para controlar el espacio en blanco.
- Utilización de tablas para la organización de las páginas.

Las hojas de estilo resuelven este problema al tiempo que superan el limitado rango de mecanismos de presentación de HTML. Con las hojas de estilo es más fácil, por ejemplo, especificar la cantidad de espacio entre líneas, los colores a utilizar para el texto y el fondo, el tamaño y tipo de las fuentes y otros muchos detalles relativos al estilo o presentación. Se puede afirmar que HTML informará al navegador de cómo se estructura el documento mientras que CSS le informará de cómo debe hacerse.

En principio, se podría usar cualquier sintaxis para la información de estilo en los documentos cuya estructura se especifique mediante HTML. Los autores pueden especificar la sintaxis de la información de estilo utilizada mediante el elemento <META> que permite especificar la sintaxis usada por defecto para la información de estilo incorporada en el documento. Por ejemplo, para especificar que el lenguaje de estilo por defecto es CSS (Hojas de Estilo en Cascada, *Cascading Style Sheets*), los autores deberían especificar en la cabecera del documento lo que aparece a continuación:

<META HTTP-EQUIV="content-style-type" CONTENT="text/css">

Los clientes deberían determinar el lenguaje de estilo por defecto siguiendo los siguientes pasos:

- Si algún elemento <META> especifica *content-style-type* como valor del atributo HTTP-EQUIV, el último de ellos en el flujo del documento determina el lenguaje de estilo por defecto.
- De otro modo, el lenguaje de estilo por defecto es "*text/css*".

Si se emplean hojas de estilo en cascada, existen tres formas de añadir información de estilo a las páginas web:

- Añadiendo instrucciones de estilo a elementos concretos, mediante el atributo STYLE.
- Incluyendo las instrucciones de estilo en la cabecera de un documento HTML concreto, mediante el elemento <STYLE>. Esto permite controlar la apariencia de todo el documento HTML mediante un conjunto de instrucciones de estilo.
- Enlazando un documento HTML, o varios, o todos los documentos HTML que componen un sitio web, mediante el elemento <LINK>, a una hoja de estilo externa que contenga la definición del estilo. De esta forma, se puede modificar la apariencia de una, múltiples o todas las páginas web de un sitio web, revisando un solo fichero.

En el diseño de un sitio web se pueden utilizar uno o más métodos simultáneamente.

Dejando ahora de un lado los aspectos más técnicos, la antigua versión de este lenguaje, CSS2.1, reunía muchas características que permitían lograr bien los objetivos marcados. Pero pronto surgió CSS3, última versión a día de hoy, para extender el conjunto de elementos estéticos disponibles, dotando al navegador web de la capacidad de controlar casi cualquier aspecto de presentación (Antón Rodríguez & Pérez Juárez, 2014).

La implementación CSS3 se ha realizado de forma que esta versión es completamente compatible con las versiones anteriores de CSS, por lo que no resulta necesario modificar estilos ya diseñados. Esto supone que las diferentes versiones de navegadores que soporten CSS3, soportarán también CSS2.1.

Por el contrario, debe tenerse en cuenta que no todas las versiones de navegadores usadas actualmente implementan de forma completa los estilos propuestos por CSS3. Además, a la hora de aplicar los nuevos estilos propuestos por CSS3, es importante tener en cuenta que buena parte de ellos aún se encuentran en fase de desarrollo, por lo que no hay establecido un estándar para su uso. Esto quiere decir que cada navegador los aplica a veces de manera diferente, incluso con algunos matices en los parámetros que emplean. Para poder utilizar estos nuevos estilos en periodo de experimentación es necesario emplear lo que se conoce como *vendor prefixes*, que son prefijos que se añaden delante de la propiedad CSS que se vaya a aplicar. Existe uno para cada navegador, aunque algunos de ellos coinciden debido a que diferentes navegadores utilizan el mismo motor de renderizado. Los principales *vendor prefixes* son los siguientes:

- *Android: -webkit-.*
- *Chrome: -webkit-.*
- *Firefox: -moz-.*

- *Internet Explorer: -ms-.*
- *iOS: -webkit-.*
- *Opera: -o-.*
- *Safari: -webkit-.*

Ejemplos de propiedades CSS3 en fase de experimentación para las que es necesario utilizar los *vendor prefixes* son los que se muestran a continuación:

- *webkit-column-count:3;*
- *moz-box-sizing: border-box;*

CSS3 se divide en módulos que incluyen las características de la versión anterior CSS2.1, así como otras características nuevas propuestas en CSS3. Algunos de los principales módulos propuestos son los siguientes (Antón Rodríguez & Pérez Juárez, 2014):

- Fondos y Bordes – Backgrounds and Borders –
- Efectos de Texto – Text Effects –
- Transformaciones 2D / 3D – 2D/3D Transformations –
- Animaciones – Animations –
- Diseño con Múltiples Columnas – Multiple Column Layout –
- Interfaz de Usuario – User Interface –

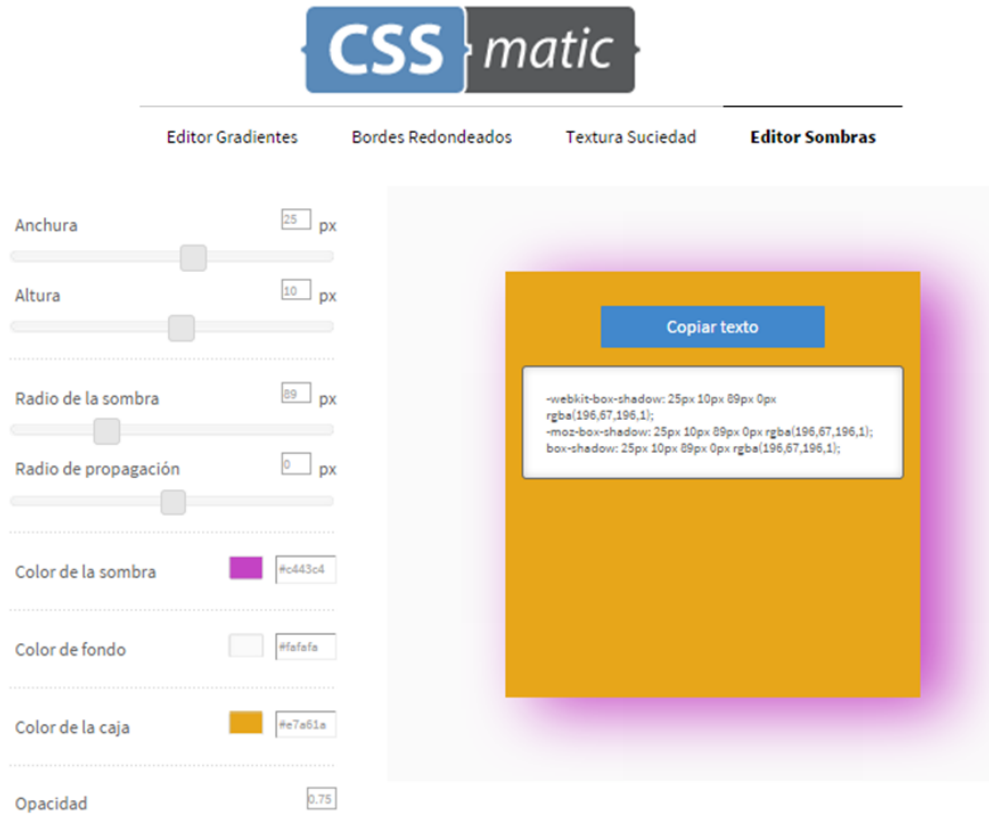


Ilustración 21. Ejemplo uso: border-radius + shadows, con herramienta web

En la imagen anterior se observa una herramienta conseguida de la página web *http://www.cssmatic.com/es/box-shadow*, de la cual han sido sacadas ideas para la aplicación móvil que he realizado. Como puede verse es muy útil para hacerse una idea de cómo pueden aplicarse las distintas propiedades de CSS3 a una determinada imagen.

Por último, se presentan, muy brevemente, algunos de los más destacados ejemplos concretos de las nuevas características que ofrece CSS3:

- Define estilos que controlan la redondez de las esquinas de muchos elementos de HTML y otros tipos de complicadas transformaciones en los bordes.
- Introduce la capacidad de agregar sombras paralelas tanto a contenedores como segmentos de texto HTML.
- Introduce poderosos métodos para controlar la opacidad y el color de contenido HTML añadiendo espacios de color con soporte para HSL (*Hue, Saturation, Lightness)*, HSLA (*HSL* con *Alpha*), RGB (*Red, Green, Blue*) y RGBA (RGB con *Alpha*).
- Añade el soporte de media *queries*, que permiten a diseñadores y desarrolladores crear estilos que se adapten a dimensiones específicas de pantalla.

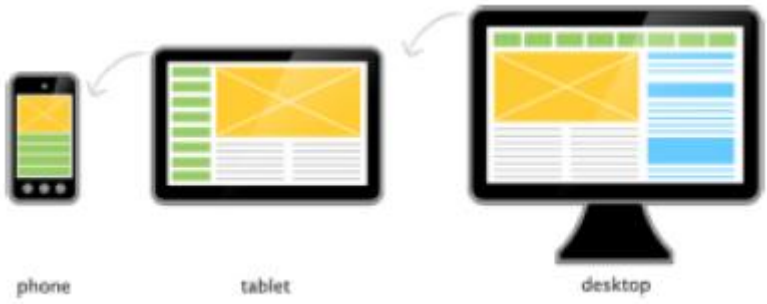


Ilustración 22. CSS3 Media *queries*. Responsividad.

- Añade nuevos *selectors*: reglas condicionales para aplicar hojas de estilo a los elementos HTML.
- Inclusión de *transformations*: estilos que controlan la transformación visual de un objeto en particular (2D y 3D).
- Implementa animación y transiciones, que dotan de movimiento a los elementos y de cambios de estado visuales.

3.3.3 JavaScript

JavaScript es un lenguaje interpretado basado en scripts que se introducen directamente en el código *HTML*. El código se envía desde el servidor al cliente y es este el que lo interpreta al cargar la página. Debido a que necesita utilizar *HTML* a través de distintos manejadores de

eventos para realizar cualquier acción, *JavaScript* no puede utilizarse para crear aplicaciones independientes, no entendidas en un contexto web o que requieran un navegador.

Las principales características de este lenguaje son:

- Es un lenguaje interpretado.
- No necesita compilación.
- Multiplataforma.
- Lenguaje de alto nivel.
- Admite programación estructurada.
- Basado en objetos.
- Maneja la mayoría de los eventos que se pueden producir sobre la página web.
- No se necesita ningún kit o entorno de desarrollo.

A diferencia de *Java*, *JavaScript* no dispone de elementos para crear interfaces de usuario propias para los programas y tiene que utilizar para ello los formularios de *HTML* a través de los denominados manejadores de eventos (Charte Ojeda, 2004).

Al igual que *HTML*, *JavaScript* es un lenguaje interpretado por el cliente y, por tanto, no existe restricción alguna en el servidor. Cada evento a ser procesado no requiere comunicación con el servidor lo que permite, por ejemplo, realizar un filtrado de la información recogida mediante un formulario antes de enviarla a una base de datos para ser almacenada.

JavaScript tampoco depende de otras condiciones de contorno como el Sistema Operativo (S.O.) pues, como ya se ha comentado, es un lenguaje que interpreta el cliente, con lo que los scripts funcionarán en todo S.O. que cuente con un cliente preparado para ello (Antón Rodríguez & Pérez Juárez, 2014).

Los bloques de código JavaScript pueden insertarse en los documentos HTML de varias formas:

- Mediante el uso de la etiqueta <SCRIPT> de HTML
- Mediante el empleo de manejadores de eventos
- Mediante el empleo del URL JavaScript

Un guión *JavaScript* se compone de sentencias, instrucciones o proposiciones que son órdenes que se dan al intérprete del lenguaje y que finalizan con un punto y coma.

En cuanto a la popularidad del lenguaje, basta con observar el porcentaje de páginas web que usan cada uno de los principales lenguajes de programación del lado del cliente, donde JavaScript aplasta a tus competidores con el 88.3% de uso a día de hoy (abril de 2015).

© W3Techs.com	usage	change since 1 March 2015
1. JavaScript	88.8%	+0.4%
2. Flash	11.4%	-0.2%
3. Silverlight	0.1%	
4. Java	0.1%	

percentages of sites

Ilustración 23. Popularidad de los lenguajes de programación del lado del cliente

3.3.4 AJAX

AJAX (Asynchronous Javascript And Xml, JavaScript asíncrono y *XML*) es un término que surgió en 2005 de la mano de Jesse James Garret para referirse a un enfoque para crear interfaces web altamente interactivas a partir de una serie de técnicas. Si bien Microsoft incorporó por primera vez este concepto en el objeto llamado *XMLHTTP*, no sería hasta años más tarde, de la mano de sus servicios e interfaz, cuando se popularizaría dicha técnica ya conocida como *AJAX*.

AJAX usa *XHTML* y *CSS* para conformar la estructura y el diseño, JavaScript como lenguaje de programación, el modelo DOM (*Document Object Model*, modelo de objetos del documento) para trabajar con la estructura del sitio y *XML* como formato, aunque no único, de transporte de los datos hacia y desde el servidor. Además de esto se necesita un lenguaje del lado del servidor como *PHP*, para controlar la lógica de servidor y el acceso a la base de datos. Es, por tanto, un conjunto de tecnologías más que una tecnología en sí misma (Firtman, 2008).

La principal ventaja de *AJAX*, que es a su vez la base de dicho enfoque, es su capacidad para mostrar contenido a partir de una consulta *HTTP* mientras un usuario de la aplicación está viendo el resto de la página. Así, el servidor web responde con un mensaje que JavaScript interpreta y a partir del cual actualiza el contenido de la página o cambia su estilo, por ejemplo (Eguiluz, 2015).

Al contrario que ocurre en el modelo clásico de interacción cliente-servidor en una página web, donde era el propio navegador el que iniciaba las peticiones hacia el servidor y procesaba su respuesta, en el modelo AJAX se tiene una capa intermedia que gestiona esta comunicación. A esta capa intermedia *Garret* la denominó motor AJAX.

Un motor *AJAX* no es más que un objeto o función en JavaScript a la que se llama cada vez que se quiere realizar una petición de información al servidor, de forma que, por ejemplo, un enlace en la página llama al motor *AJAX* en lugar de llevar a otra página, y es el motor *AJAX* el que planifica y ejecuta la petición al servidor de forma síncrona. Dicho motor *AJAX*, al recibir la respuesta del servidor, analiza los datos de esta y realiza cambios en la interfaz de la aplicación del cliente según la información devuelta por el servidor. Puesto que este proceso asíncrono conlleva una menor transferencia de información que en el modelo de página web tradicional,

esto es, si fuera necesario recibir toda la página para poder una parte menor de contenido, la interfaz de usuario se actualiza más rápido, aportando una mejor experiencia de usuario.

En la siguiente ilustración se observa que en la imagen de la izquierda se muestra el modelo tradicional de las aplicaciones web. La imagen de la derecha muestra el nuevo modelo propuesto por AJAX (Firtman, 2008):

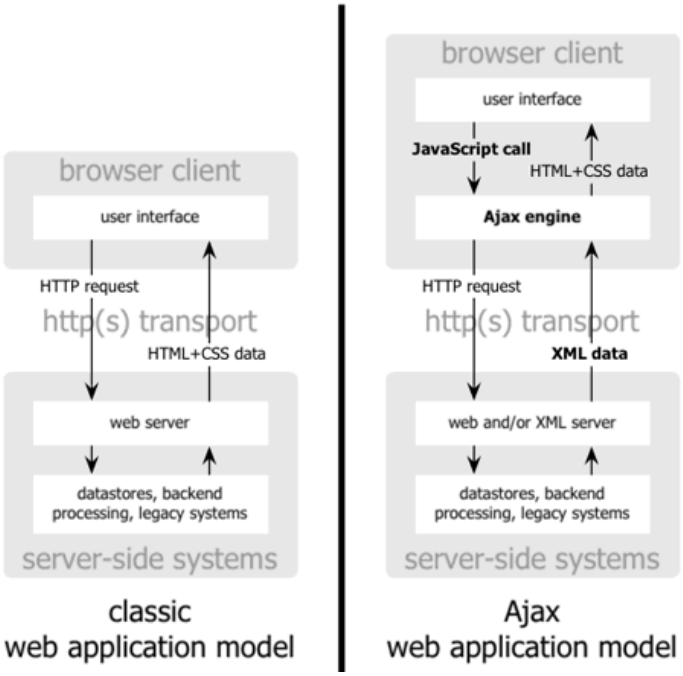


Ilustración 24. Comparación gráfica del modelo tradicional de aplicación web y del nuevo modelo propuesto por AJAX

3.3.5 jQuery

jQuery es una biblioteca de JavaScript de código abierto que permite simplificar la manera en que se interactúa con los documentos HTML, en que se manipula el árbol de elementos DOM, el manejo de eventos, el desarrollo de animaciones y la agregación de interacción con la tecnología AJAX. Hace que sea mucho más simple debido a un API fácil de usar que funciona a través de una multitud de navegadores. Con una combinación de versatilidad y capacidad de ampliación, *jQuery* ha cambiado la forma en que millones de personas escriben *JavaScript*.

jQuery es una biblioteca de JavaScript muy rápida, potente y rica en funciones que vio la luz por primera vez en agosto de 2006, si bien la idea inicial ya fue planteada mucho antes y desarrollada por su creador, John Resig (Franklin, 2013) .

El mejor resumen de lo que *es jQuery* lo podemos encontrar en el lema de su propia página web: “La librería JavaScript para escribir menos y hacer más”. Es una forma de convertir el desarrollo de la parte de cliente de una aplicación web en algo mucho más divertido, rápido y

sencillo, facilitando la interacción con los elementos del árbol de documento, el manejo de eventos, el uso de animaciones, etc.

Gracias a las funciones aportadas por esta biblioteca se consigue reducir el tiempo de desarrollo del código así como el espacio que ocupa este. Para hacer uso de la librería *jQuery* se puede descargar el script desde la página web oficial (http://jquery.com/) e incluirla en el código de este proyecto (The jQuery Foundation, 2015).

3.4 Lenguajes de programación del lado del servidor

Se denominan lenguajes de programación del lado del servidor a aquellos que se ejecutan en el servidor produciendo una salida que es la que le llega al cliente. Existen diferentes opciones a utilizar en esta categoría como son PHP (*PHP: Hypertext Preprocessor*), ASP (*Microsoft Active Server Pages*) o JSP (*Java Server Pages*).

© W3Techs.com	usage	change since 1 March 2015
1. PHP	82.0%	-0.1%
2. ASP.NET	17.0%	
3. Java	2.9%	+0.1%
4. ColdFusion	0.7%	
5. Ruby	0.6%	

percentages of sites

Ilustración 25. Popularidad de los lenguajes de programación del lado del servidor.

En la figura anterior, se observa que lenguajes de programación del lado del servidor son más populares actualmente. Este ranking ha sido obtenido del sitio: *W3Techs-World WideWeb Technology Surveys*: Web especialista en encuestas sobre las tecnologías web (W3techs, 2015).

3.4.1 PHP

PHP (*PHP Hypertext Preprocesor*) es un lenguaje de programación interpretado, cuyos comandos se ejecutan en el servidor y permiten la creación de documentos HTML dinámicos. Su sintaxis es similar a la de otros lenguajes como C, Perl, Java o JavaScript (Antón Rodríguez & Pérez Juárez, 2014).

PHP es un producto de código abierto, lo que quiere decir que se puede acceder a su código, usarlo, modificarlo y distribuirlo de forma gratuita sin que suponga coste alguno, al contrario de lo que ocurre con los productos comerciales. Esta es, sin duda, una de las principales ventajas de PHP, ya que la comunidad de usuarios que desarrollan mejoras para el sistema es

enorme, lo que proporciona a PHP una vitalidad y capacidad de adaptación que no poseen otros lenguajes de programación. Toda la evolución de PHP puede seguirse en el sitio web www.php.net.

PHP se encuentra disponible para muchas plataformas incluyendo *Windows*, *Unix* o *Linux* y con la ventaja de que las aplicaciones desarrolladas en PHP se pueden transportar de una plataforma a otra sin necesidad de modificaciones, es decir que PHP presenta una portabilidad elevada. A pesar de ser en la actualidad un lenguaje multiplataforma, debe comentarse que su entorno nativo es Unix/Linux. Con respecto al servidor web también existirían diferentes opciones, ya que podría usarse por ejemplo IIS (Internet Information Server) de Microsoft o Apache, este último con la ventaja de que, al igual que PHP, es también un producto de libre distribución.

Una de las características más potentes y destacables de PHP es su soporte para una gran cantidad de bases de datos. Escribir una página web con acceso habilitado a una base de datos es increíblemente simple utilizando una de las extensiones específicas (por ejemplo, para *MySQL*), o utilizar una capa de abstracción como PDO (PHP *Data Objects*), o conectarse a cualquier base de datos que soporte el estándar de Conexión Abierta a Bases de Datos por medio de la extensión ODBC (*Open DataBase Connectivity*) (Gutierrez Gallardo, 2009).

PHP fue concebido en 1994 por Rasmus Lerdorf con la intención de crear un contador para averiguar el número de visitas que recibía su CV virtual. Sin embargo, con el tiempo ha sido adoptado por otros desarrolladores que lo han transformado y convertido en la herramienta que es hoy en la actualidad. Una herramienta utilizada ya en más de 200 millones de dominios de Internet, número que sigue creciendo y cuya evolución responde a la siguiente gráfica:

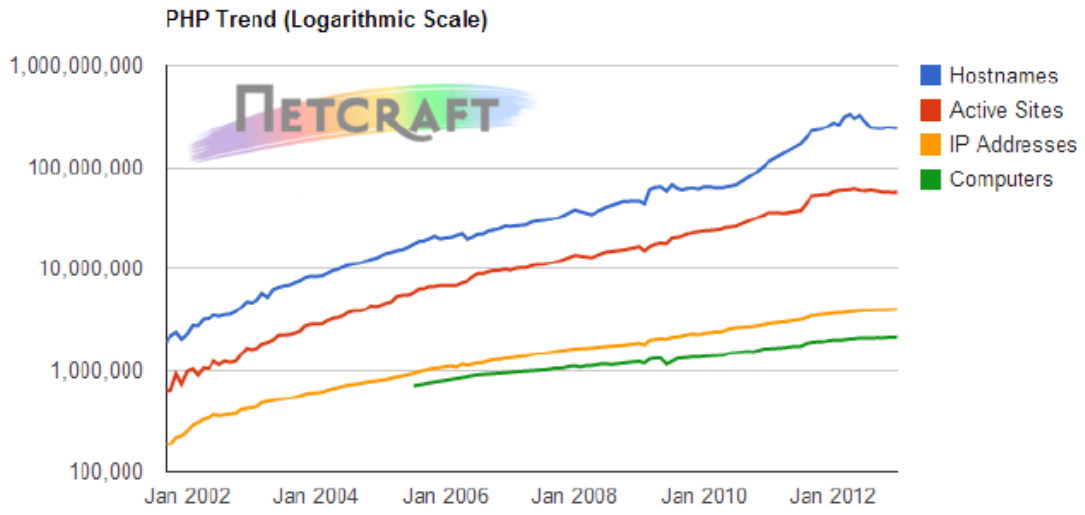


Ilustración 26. Evolución de PHP

PHP es utilizado por el 82.0% (abril de 2015) de los sitios web de los que se conoce el lenguaje del lado del servidor utilizado. En la siguiente figura se puede ver la gráfica

comparativa con otros de los principales lenguajes de programación de lado del servidor. Esta información ha sido obtenida del sitio: *W3Techs-World WideWeb Technology Surveys:* Web especialista en encuestas sobre las tecnologías web (W3techs, 2015).

En la ilustración inferior se puede observar el uso dado a cada lenguaje de programación del lado del servidor considerando el número de sitios web desarrollados en cada lenguaje y su tráfico.

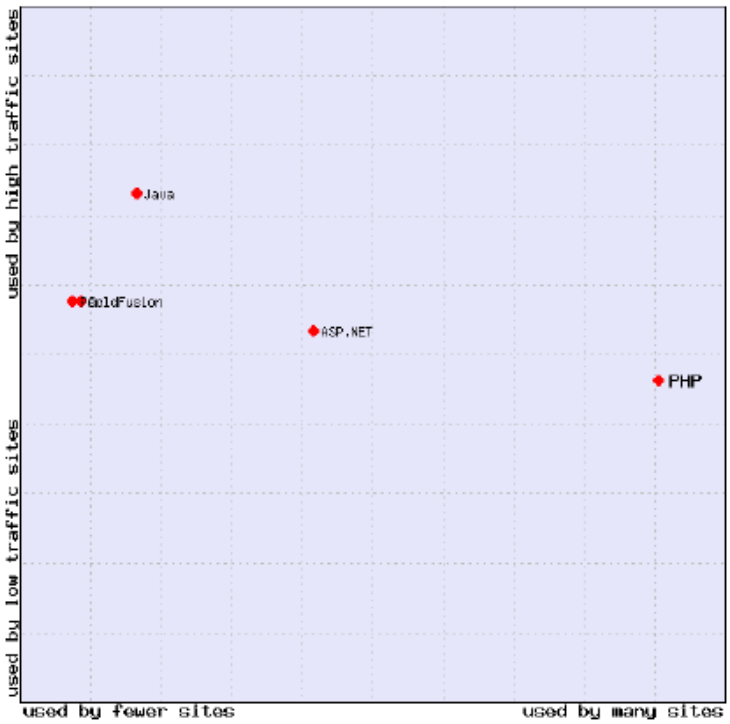


Ilustración 27. Relación de la extensión de distintos lenguajes de programación con su tráfico

Una tecnología que aparezca cerca de la esquina inferior derecha se utiliza en muchos sitios web aunque con un tráfico medio. Una tecnología situada cerca de la esquina superior izquierda se utiliza en pocos sitios web, pero principalmente con un tráfico alto. Las mejores posiciones se encuentran cerca de la esquina superior derecha. En concreto, PHP se utiliza en muchos sitios web del tráfico medio. Esta información ha sido obtenida del sitio: *W3Techs-World WideWeb Technology Surveys*: Web especialista en encuestas sobre las tecnologías web (W3techs, 2015).

Algunos ejemplos de sitios web que emplean PHP son los siguientes:

- *phpMyAdmin*
- *SquirrelMail*

- *OScommerce*
- *Magento eCommerce*
- *Moodle*
- *Drupal*
- *Joomla*
- *Gallery*
- *MediaWiki*
- *Facebook*
- *Tuenti*
- *WordPress*
- *Elgg*
- *Digg*
- *Sourceforge*
- *MyYearbook*

Por último, comentar el aspecto funcional del lenguaje. El código PHP se puede insertar en un documento HTML, siendo así el proceso de generación de una página web dinámica con PHP el que se describe a continuación:

- Cuando un usuario hace clic desde su navegador en un enlace correspondiente a un documento HTML que contiene código PHP, el navegador realiza la solicitud al servidor web correspondiente.
- El servidor web localiza entonces el documento, detecta que contiene código PHP y pone en funcionamiento el intérprete del lenguaje.
- Dicho intérprete ejecuta el código PHP y genera un resultado, generalmente en forma de página web, que se devuelve al navegador para que éste se encargue de su visualización.

A modo de resumen de PHP, se puede concluir que es un lenguaje de programación gratuito, interpretado (necesita intérprete), sencillo de aprender y que permite realizar diseños escalables y orientados a objetos.

3.4.2 ASP

ASP (*Active Server Pages*) es la tecnología diseñada por Microsoft para facilitar la creación de sitios web con una mayor sencillez que la empleada en la programación CGI (*Common Gateway* Interface).

ASP requiere un servidor Web de *Microsoft*. Para utilizar la tecnología *ASP* sobre otros servidores, por ejemplo servidores Unix, se necesita un software intérprete (*Chilisoft*, *Instant ASP*).

El núcleo de funcionamiento de ASP es una aplicación ISAPI (*Internet Server API*). Una aplicación ISAPI es una DLL de *Windows* que se ejecuta en el mismo espacio de direcciones que el servidor Web y que puede soportar varias peticiones simultáneas.

ASP no es realmente un lenguaje como tal, siendo el lenguaje usado en realidad para programar ASP es Visual Basic Script o JavaS*cript* (versión *Microsoft* de *JavaScript*).

Como resumen de la tecnología ASP se puede decir que es sencillo de aprender, que permite realizar diseños escalables y orientados a objetos pero, como principales características negativas, es propietario de *Microsoft* y no se puede ejecutar en cualquier plataforma, sino únicamente sobre servidores con sistema operativo *Windows* (Cobo, Gómez, Pérez, & Rocha, 2011).

3.4.3 Servlets y JSP

Los *servlets* y JSP (*Java Server Pages*) son dos métodos de creación de páginas Web dinámicas en servidor usando el lenguaje *Java*. Se trata de tecnologías desarrolladas por la empresa *Sun Microsystems*.

Las JSP se diferencian de otras tecnologías del lado del servidor como las ASP en dos aspectos principalmente: por un lado los JSP y *servlets* se ejecutan en una máquina virtual Java, lo cual permite que, en principio, se puedan usar en cualquier tipo de ordenador, siempre que tenga instalado esa máquina virtual. Por otro lado, un programa JSP se compila a un programa en Java la primera vez que se invoca, y del programa en Java se crea una clase que se empieza a ejecutar en el servidor como un *servlet*. De esta manera los *servlets* no se ejecutan cada vez que se recibe una petición, sino que persisten de una petición a la siguiente, lo que permite realizar operaciones como la conexión a bases de datos o manejo de sesiones de una manera más eficiente (Cobo, Gómez, Pérez, & Rocha, 2011).

Un JSP es una página web con etiquetas especiales y código Java incrustado, mientras que un servlet es un programa que recibe peticiones y genera a partir de ellas una página web. En ambos casos se necesita un programa servidor que se encargue de recibir las peticiones, distribuirlas entre los *servlets* y realizar las tareas de gestión propias de un servidor web. Estos programas suelen llamarse contenedores de *servlets* o *servlets engines*, y, entre otros, podrían citarse como ejemplos *Resin*, BEA *Weblogic*, *JRun* de Macromedia, Lutris Hendirá, o, quizás el más popular y conocido: *Toncat* .

A modo de resumen de JSP, cabe destacar que es gratuito, puede ser utilizado en cualquier plataforma y permite realizar diseños escalables y orientados a objetos.

Sin embargo, su aprendizaje puede resultar complejo frente a las alternativas PHP y ASP.

3.4.4 Elección del lenguaje de programación del lado del servidor

¿Por qué he elegido PHP frente a otros competidores como son Perl, ASP o JSP?

- Alto rendimiento. PHP es muy eficiente. Mediante el uso de un único servidor, puede servir millones de accesos al día. Los indicadores comparativos de rendimiento publicados por Zend Technologies muestran que PHP supera ampliamente a sus competidores en esta faceta (Zend Technologies , 2015).
- Integración de bases de datos. PHP dispone de una conexión propia a todos los sistemas de base de datos. Puede conectarse directamente a las bases de datos de *MySQL, PostgreSQL, mSQL, Oracle, dbm, filePro, Hyperwave, Informix, Internase y Sybase,* entre otras. Esto se debe a que PHP utiliza ODBC (*Open Database Connectivity Standard*).
- Bibliotecas incorporadas. Como se ha diseñado para su uso en la Web, PHP incorpora una gran cantidad de funciones integradas para realizar útiles tareas relacionadas con la Web. Puede generar imágenes GIF al instante, establecer conexiones a otros servicios de red, enviar correos electrónicos, trabajar con cookies y generar documentos PDF, todo con unas pocas líneas de código (Gutierrez Gallardo, 2009).
- Coste. PHP es un lenguaje de programación gratuito y, por tanto, todo el mundo puede utilizarlo sin ningún coste, frente a otros lenguajes cuyo software es necesario comprar para su utilización.
- Aprendizaje de PHP. Es un lenguaje relativamente sencillo de aprender, especialmente si se conocen previamente otros lenguajes de programación ya que tiene influencias de muchos de ellos, principalmente de C y Perl.
- Portabilidad. PHP está disponible para una gran cantidad de sistemas operativos diferentes. Se puede escribir código PHP en todos los sistemas operativos gratuitos del tipo Unix, como Linux y *FreeBSD*, versiones comerciales de Unix, como Solaris e IRIX o en las diferentes versiones de Microsoft Windows. El código funcionará sin necesidad de aplicar ninguna modificación a los diferentes sistemas que ejecute PHP.
- Código fuente. Se dispone de acceso al código fuente de PHP. A diferencia de los productos comerciales y de código cerrado, si se desea modificar algo o agregar un elemento al programa, se puede hacer con total libertad. No se necesita esperar a que el fabricante publique parches, ni es necesario preocuparse porque el fabricante cierre sus puertas o decida abandonar el producto (Gutierrez Gallardo, 2009).

3.5 Bases de datos

Un SGBD (Sistema Gestor de Bases de Datos) es un sistema computacional que facilita la gestión de las bases de datos.

Una base de datos podría definirse como una colección de datos interrelacionados que son almacenados en un soporte informático. Algunas razones que justifican su uso son su capacidad para almacenar grandes volúmenes de información, la optimización de su gestión, la facilidad para realizar consultas y la exactitud, rapidez y fiabilidad en su administración (Cobo, Gómez, Pérez, & Rocha, 2011).

Existen diferentes tipos de bases de datos, entre las que se encuentran las de XML, las orientadas a objetos y las relaciones, que serán descritas a continuación.

3.5.1 Bases de datos orientadas a objetos

Con el auge de la programación orientada a objetos surgió la posibilidad de extender este concepto al de base de datos. Así, se tendría un sistema gestor de bases de datos orientadas a objetos (SGBDO), del inglés OBDMS (*Objetc DataBase Management System*). De esta forma, se puede trabajar con objetos complejos, herencias y otras características propias de los lenguajes de programación orientados a objetos (Ramakrishnan & Gehrke, 2003).

ODMG (*Object Database Management Group*, Grupo de Gestión de Bases de Datos de Objetos) ha creado un modelo de datos orientado a objetos u ODM (*Object Data Model*) y un lenguaje de consulta orientado a objetos u OQL (*Object* Query *Languague*) normalizados. Estos son el equivalente a la norma SQL en los sistemas de bases de datos relacionales.

Sin embargo, según (Silberschatz & F. Korth, 2002), el modelo de base de datos relacional, que se verá en el próximo apartado, se ha establecido hasta ahora como el principal para las aplicaciones de procesado de datos debido, principalmente, a su simplicidad frente a otros modelos.

3.5.2 Bases de datos relacionales

El modelo de datos relacional organiza y representa los datos en forma de tablas o relaciones. El término relación representa una tabla de dos dimensiones formada por filas y columnas de datos. Cada fila de esta tabla contiene un conjunto de valores relacionados entre sí. Dicha tabla tiene un nombre, así como cada una de las columnas que la forman. Estos nombres clarifican el significado de los valores contenidos en la tabla (Rivero Cornelio, Martínez Fuentes, & Alonso Martínez, 2005).

Adicionalmente, una base de datos relacional tiene los siguientes componentes:

- Estructura de datos: se trata de una colección de objetos abstractos formados por datos, dominios, tuplas, atributos y relaciones.
- Operadores: permiten manipular las estructuras de datos a partir de unas reglas bien definidas. Estos operadores, además del cambio de esquema, son la unión, diferencia, producto cartesiano, proyección y selección, es decir, los primitivos del álgebra relacional para manipulación de datos.
- Definiciones de integridad: conjunto de reglas y conceptos que posibilita expresar qué valores de datos pueden aparecer de forma válida en el esquema.

Se incluyen las claves, la posibilidad de tener valores nulos y la reglas de integridad de claves primarias y de integridad referencial.

Integridad de claves primarias:

Las claves pueden usar usadas como identificadores de las tuplas en una relación dada, ya que a cada valor de una clave le corresponde únicamente una tupla y viceversa. En el modelo relacional, solo se puede encontrar una tupla determinada si se conoce el valor de una clave.

Una relación puede disponer de varias claves, aunque se suele emplear siempre la misma como identificador. A esta clave, empleada para identificar una relación, se la denomina clave primaria. El resto de claves se llaman claves secundarias o alternativas (Rivero Cornelio, Martínez Fuentes, & Alonso Martínez, 2005).

Como se ha comentado, la clave primaria es la que identifica a una relación por lo que no debe tomar valores nulos para evitar cualquier ambigüedad. A esta condición se la llama regla de integridad de claves primarias o regla de integridad de entidad. Es decir, ningún atributo de una clave primaria podrá valores nulos.

Integridad referencial:

Unas relaciones pueden hacer referencia a otras mediante claves primarias de estas. Adicionalmente al concepto de clave primaria, existe el de clave ajena, que se da en un atributo A de una relación R cuando se requiere que todos los valores de A no nulos existan en la clave primaria de alguna relación que no tiene por qué ser necesariamente distinta de R. Es decir, si A es un conjunto de atributos (A1, A2,… An), la definición anterior es válida si A toma valor nulo cuando alguno de sus componentes sea nulo.

Al identificar como clave ajena a un atributo, es recomendable definir las acciones a realizar en caso de intentar actualizar dicho atributo con valores no válidos. Estas acciones dependerán del significado de los datos (Rivero Cornelio, Martínez Fuentes, & Alonso Martínez, 2005).

Volviendo a las bases de datos relacionales, estas presentan unas características diferenciadoras frente al resto de modelos como son atomicidad de los valores de los atributos, la no repetición de tuplas, la no ordenación de tuplas y la no ordenación de los atributos. Debido a esto, la forma en que se almacenan los datos en el modelo de datos relacional no importa, contribuyendo esto a una mayor facilidad en el uso de la base de datos por parte del usuario. Al contrario de lo que ocurría en otros modelos de bases de datos vistos anteriormente, se pueden realizar consultas para recuperar o almacenar información de forma flexible y con poder sobra la administración de la información. Adicionalmente, durante la fase de diseño de una base de datos relacional, se realiza un proceso de normalización mediante el cual cada relación queda descrita en términos de dependencia. Dicho proceso evita la redundancia de datos, lo que a su vez evita problemas al actualizar los datos y permite proteger la integridad de los mismos (Antón Rodriguez, de la Torre Díez, Gutiérrez Díez, & Díaz Pernas, 2010).

3.5.2.1 SQL

SQL (*Structured Query Language*, lenguaje estructura de consulta), fue desarrollado originalmente por IBM a partir de los proyectos SEQUEL-XRM y *System*-R (1974-1977), resultados de la búsqueda de un prototipo de SGBD relacional. Así, a finales de la década de los ochenta surgió SQL, IBM y otras empresas comenzaron a utilizar SQL en sus SGBD relacionales, adquiriendo gran popularidad hasta convertirse en el lenguaje comercial para bases de datos relacionales más utilizado actualmente.

A la hora de denominar los diferentes elementos que se encuentra en la base de datos se emplean los siguientes términos, que difieren de los introducidos anteriormente en el apartado de bases de datos relacionales: se emplea el término tabla en lugar de relación, columna en lugar de atributo y fila en lugar de tupla.

Además, SQL presenta varias características (Silberschatz & F. Korth, 2002):

- LMD, Lenguaje de Manipulación de Datos: permite a los usuarios realizar consultas a la base de datos así como insertar, eliminar y modificar filas de esta.

- LDD, Lenguaje de Definición de Datos: posibilita crear, eliminar y modificar definiciones de tablas y vistas. Adicionalmente, se pueden establecer restricciones de integridad en las tablas en el momento de crearlas o posteriormente.

- Disparadores y restricciones de integridad avanzadas: siendo los disparadores acciones ejecutadas por el SGBD cuando se realizan modificaciones en la base de datos que cumple con las condiciones establecidas en dichos disparadores.

- SQL incorporado, que permite ejecutar el código SQL en lenguajes anfitriones como C o Cobol, y SQL dinámico, que permite crear y ejecutar consultas durante el tiempo de ejecución.

- Ejecución cliente-servidor y posibilidad de acceso a la base de datos de forma remota: a partir de estas órdenes se controla la forma en que aplicaciones clientes se conectan con servidores de bases de datos de SQL o tienen acceso a los datos de estas bases de datos a través de la red.

- Gestión de transacciones: mediante distintas órdenes a partir de las cuales los usuarios controlan el modo en que se ejecutan dichas transacciones.

- Seguridad: mediante mecanismos ofrecidos por SQL para el control del acceso de los usuarios a los diferentes datos, tablas y vistas.

3.5.2.2 MySQL

MySQL es un sistema gestor de bases de datos relacionales, que además ofrece compatibilidad con *PHP*, *Perl*, C y *HTML*, y funciones avanzadas de administración y optimización de bases de

datos para facilitar las tareas habituales. Implementa funcionalidades Web, permitiendo un acceso seguro y sencillo a los datos a través de Internet. Este sistema gestor de base de datos incluye capacidades de análisis integradas, servicios de transformación y duplicación de datos y funciones de programación mejoradas.

Se puede decir que *MySQL* es un sistema cliente servidor de administración de bases de datos relacionales diseñado para el trabajo tanto en los sistemas operativos Windows como en los sistemas *UNIX*/*LINUX*. Además, determinadas sentencias de *MySQL* pueden ser embebidas en código *PHP* y *HTML* para diseñar aplicaciones Web dinámicas que incorporan la información de las tablas de *MySQL* a páginas Web (Pérez López, 2007).

Entre los competidores principales de *MySQL*, se puede citar a *PostgreSQL*, *Microsoft SQL* Server y Oracle. Sin embargo, *MySQL* dispone de una serie de ventajas que le hacen fuerte frente a sus competidores:

- Alto rendimiento. *MySQL* es muy rápido.
- Bajo coste. *MySQL* está disponible de manera gratuita.
- Facilidad de uso.
- Portabilidad. *MySQL* se puede utilizar en una gran cantidad de sistemas Unix diferentes, así como bajo Microsoft Windows.
- Código fuente accesible. Al igual que en el caso de PHP, se puede obtener y modificar el código fuente de *MySQL*.

3.6 Elección del lenguaje del lado del servidor y la base de datos

Se ha visto anteriormente, en el apartado 3.4, tres lenguajes de programación del lado del servidor diferentes, analizando sus características para estudiar la conveniencia de su uso en el presente proyecto. El primer lenguaje en ser descartado ha sido *ASP* debido, principalmente, a su dependencia de la plataforma *Microsoft*, necesitando un servidor web con sistema operativo de *Microsoft*. Adicionalmente, las aplicaciones desarrolladas en *ASP* son más lentas, más pesadas y menos estables que en alternativas como, por ejemplo, *PHP* (Cobo, Gómez, Pérez, & Rocha, 2011). De las dos opciones restantes, JSP posee una curva de aprendizaje peor que PHP, además de ser este último uno de los lenguajes más utilizados actualmente en el desarrollo de aplicaciones web.

A partir del análisis de las tres posibles opciones estudiadas de lenguajes de programación en el lado del servidor, se concluye que *PHP* es la mejor opción a utilizar en el desarrollo del proyecto.

En cuanto a la base de datos, a partir de las características presentadas anteriormente de los tipos de bases de datos vistos, se concluye que el modelo relacional es el más adecuado para el proyecto a realizar.

Por un lado, se han descartado las bases de datos *XML, p*or otro lado, las bases de datos orientadas a objetos pueden estar más indicadas para aplicaciones con datos complejos o irregulares donde se tienen patrones previsibles. Por este motivo, se opta por las bases de datos relacionales como mejor opción, añadiendo además la simplicidad de este modelo como ventaja. Adicionalmente, se tiene flexibilidad a la hora de recuperar o almacenar información y la integridad de los datos está asegurada, siendo el modelo más utilizado en sistemas del tipo del realizado en este proyecto (Antón Rodriguez, de la Torre Díez, Gutiérrez Díez, & Díaz Pernas, 2010).

Como sistema gestor de la base de datos relacional se utilizará *MySQL*, por ser uno de los *SGBDR* (Sistema gestor de bases de datos relacional) más utilizados en la actualidad, además de ser la primera opción a utilizar si se usa *PHP* como lenguaje de programación del lado del servidor, debido a su alto rendimiento.

Por lo tanto, se usara un modelo de bases de datos relacional empleando *MySQL* como SGBDR, PHP como lenguaje de programación del lado del servidor y la extensión *mysql* en PHP para la conexión desde este a *MySQL*.

3.6.1 phpMyAdmin

Tras la elección de *MySQL* como SGBDR (Sistema gestor de bases de datos relacional), resulta necesaria la mención de *phpMyAdmin*, herramienta de administración de bases de datos mediante *MySQL* desde la web escrita en *PHP*.

La administración y gestión de las bases de datos de *MySQL* mediante el propio monitor de *MySQL* resulta en ocasiones un tanto laboriosa, especialmente para aquellos usuarios acostumbrados al uso de herramientas con interfaz gráfica de usuario.

Para solventar este hecho, han ido surgiendo diversas opciones en cuanto a administración de bases de datos que pueden resultar más intuitivas y fáciles de utilizar. Muchas de estas opciones alternativas están desarrolladas en PHP y facilitan la administración de bases de datos de forma remota vía web. Quizás la herramienta más conocida, y una de las más utilizadas, sea *phpMyAdmin*, desarrollada en PHP por una comunidad de usuarios sin ánimo de lucro y disponible gratuitamente desde su propia página (http://www.phpmyadmin.net).

PhpMyAdmin no es más que un conjunto de páginas escritas en PHP que son copiadas directamente en el directorio que aloja las páginas web del servidor. Mediante diferentes páginas se pueden consultar las bases de datos disponibles, crear nuevas bases de datos, tablas, realizar consultas, insertar registros, administrar los usuarios y sus privilegios, hacer copias de seguridad de las bases de datos, etc. (Cobo, Gómez, Pérez, & Rocha, 2011).

Por la facilidad de uso que supone el empleo de esta herramienta frente a la consola SQL se decidió hacer uso de esta herramienta para el manejo de la base de datos de nuestra aplicación de una clínica pediátrica.

3.7 Servicios WEB

3.7.1 Introducción a los servicios WEB

Existen múltiples definiciones sobre lo que son los Servicios Web, por lo que resulta complicado ofrecer una versión que englobe todo lo que son e implican. Según el *W3C* (*World Wide Web Consortium*), "*un Servicio Web es un sistema software diseñado para soportar la interacción entre diferentes máquinas a través de una red. Otros sistemas interactúan con el Servicio Web de una manera predeterminada en su descripción utilizando mensajes SOAP, típicamente transportados mediante HTTP con mensajes XML junto con otros estándares relacionados con la web*" (W3C Working Group Note, 2004).

Los Servicios Web se pueden ver como una nueva etapa el desarrollo de los sistemas distribuidos. Permiten aprovechar las ventajas de trabajar en un ambiente web y utilizar una gran cantidad de tecnologías que sean adecuadas para desarrollar los componentes finales (Sosa Sosa).

Uno de los principales beneficios que proporcionan es el grado de interoperabilidad entre sistemas. O lo que es lo mismo, permiten disponer de componentes software independientes que pueden compartir sus funcionalidades con otros servicios y aplicaciones.

El esquema de funcionamiento de los Servicios Web requiere tres agentes fundamentales que pueden verse en la siguiente ilustración:

- Un proveedor del Servicio Web. Se trata de quien lo diseña, desarrolla, implementa y pone a disposición para su uso.
- Un solicitante del Servicio Web. Se trata de quien accede al componente para utilizar los servicios que presta.
- Un directorio. Sirve de enlace entre el proveedor y el solicitante, a efectos de publicación, búsqueda y localización del servicio.

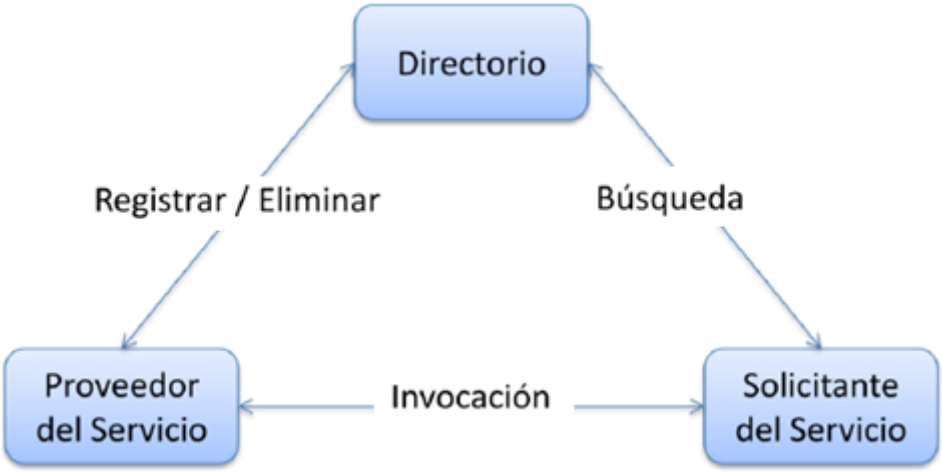


Ilustración 28. Agentes fundamentales de los servicios web

Para poner el Servicio Web a disposición para su uso se deben seguir los siguientes pasos:

- En primer lugar, es necesario definir el Servicio Web. Para que pueda interactuar con otros se debe utilizar un lenguaje común que permita estructurar los datos que componen el servicio. Para ello se suele hacer uso de XML (*eXtensible Markup Language*).
- A continuación se debe publicar el servicio, con el objetivo de que otros servicios y otras aplicaciones puedan interactuar con él. Esto implica ubicar el servicio en un determinado servidor y realizar una descripción del mismo, para que los clientes puedan saber qué funciones implementa dicho servicio y cómo se debe emplear el mismo para poder utilizar esas funciones.

El lenguaje que se utiliza para elaborar la descripción del servicio es WSDL (*Web Service Description Language*), y la publicación del mismo se realiza mediante UDDI (*Universal Description, Discovery and Integration*), sea cual sea el servidor (público o privado).

Por su parte, cuando un usuario quiera solicitar un Servicio Web debe disponer de un directorio que tenga las referencias a los servicios disponibles. Para intercambiar información entre el proveedor y el solicitante se debe hacer uso de un protocolo de comunicaciones, como SOAP (*Simple Objetc Access Protocol*), que transmitirá los datos sobre HTTP, *FTP* o *SMTP* en formato *XML*, o *REST*, que intercambiará la información sobre *HTTP* en formato *XML* o *JSON* (Mateos Diaz, 2005).

✓ XML: eXtensible Markup Language

XML es un lenguaje de marcado, como *HTML*. Sin embargo, mientras *HTML* fue diseñado para mostrar datos, centrándose en la apariencia de los mismos, *XML* fue diseñado para transportar y almacenar estos datos, centrándose en el propio contenido. Además, en *XML* las etiquetas no están predefinidas, sino que uno debe definir sus propias etiquetas. En sus orígenes se

diseñó para solventar los desafíos de la publicación electrónica a gran escala, pero actualmente está tomando un papel cada vez más importante en el intercambio de datos en la Web (W3Schools, 2015).

Si se utiliza *HTML*, será necesario un gran esfuerzo para mostrar datos dinámicos en el documento cada vez que cambien. Con *XML*, los datos pueden almacenarse en ficheros XML independientes. Así, se podrá emplear *HTML* y *CSS* para el diseño y la visualización de la información y estar seguro de que cambios en los datos no van a necesitar cambios en el código *HTML*. Únicamente con unas pocas líneas de código *JavaScript* se podrá leer los archivos *XML* externos y actualizar el contenido.

Generalmente, diferentes sistemas independientes trabajan con datos en formatos incompatibles. *XML* simplifica el intercambio de estos datos, ya que los datos *XML* se almacenan en formato de texto plano, proporcionando un software independiente de la forma de almacenamiento de los datos (W3Schools, 2015).

3.7.2 Arquitecturas comunes para servicios WEB

- **RPC: Remote Procedure Calls**

Los Servicios Web basados en *RPC* presentan una interfaz de llamada a procedimientos y funciones distribuidas. Típicamente, la unidad básica de este tipo de servicios es la operación *WSDL* que, como ya se ha explicado, es un descriptor del Servicio Web. Las primeras herramientas para Servicios Web estaban centradas en esta visión, incluso es denominado por un sector como la primera generación de Servicios *Web*. Este es el motivo por el cual RPC está muy extendido. Sin embargo, *RPC* se critica por no ser débilmente acoplado, ya que suele implementarse por medio del mapeo de servicios directamente a funciones específicas del lenguaje o llamadas a métodos.

- **SOA: Service-Oriented Architecture**

Los Servicios Web implementados en *SOA* tienen como unidad básica de comunicación el mensaje, en lugar de la operación. Esto es comúnmente referenciado como servicios orientados a mensajes. Los servicios Web basados en *SOA* son soportados por la mayor parte de desarrolladores de software y analistas. Al contrario que los Servicios Web basados en *RPC*, este estilo es débilmente acoplado, lo cual es preferible porque se centra en el contrato proporcionado por el documento *WSDL*, más que en los detalles de implementación subyacentes.

- **REST: REpresentation State Transfer**

Los Servicios Web basados en *REST* intentan emular al protocolo *HTTP* o similares mediante la restricción de establecer la interfaz a un conjunto conocido de operaciones estándar (como *GET* o *PUT*). Por tanto, este estilo se centra más en interactuar con recursos con estado que con mensajes y operaciones.

- **SOAP: Simple Object Acces Protocol**

SOAP es un protocolo de comunicación, basado en XML, que sirve para invocar Servicios Web a través de un protocolo de transporte, como HTTP. Consta de tres partes: una descripción del contenido del mensaje, unas reglas para la codificación de los tipos de datos en XML y una representación de las llamadas RPC para la invocación y respuestas generadas por el Servicio Web (W3C, 2015) .

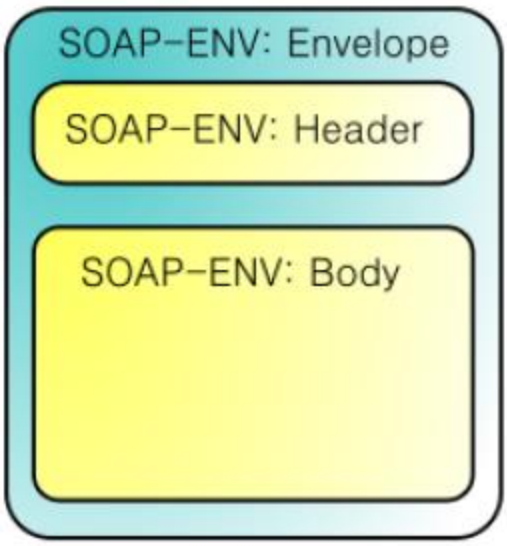


Ilustración 29. Estructura de los mensajes SOAP

El mensaje *SOAP* en un documento *XML* contiene tres elementos como se ve en la ilustración superior:

- Un elemento *envelope*, que identifica el documento XML como un mensaje SOAP.
- Un elemento *header*, que contiene la información de cabecera.
- Un elemento *body*, que contiene la información de peticiones y respuestas.

Adicionalmente puede contener un elemento *fault*, que almacena errores e información de estado.

La librería para Android es ksoap2-android, (kSoap2-Android, 2014) y nos permite crear los objetos SOAP y procesar su envío. Es importante señalar que SOAP ha tenido una gran acogida por parte de Android, lo cual será un factor determinante a la hora de decantarse por una implementación u otra.

- **RESTful**

Un servicio *RESTful* (o *RESTful* web *API*), es un Servicio Web que se ha implementado usando *HTTP* y el estilo de *REST*. *REST* proporciona un protocolo cliente/servidor sin estado, es decir, cada mensaje *HTTP* contiene toda la información necesaria para comprender la petición. Esto implica que no sea necesario recordar por parte del cliente ni del servidor recordar ningún estado de las comunicaciones entre mensajes.

Al contrario de SOAP, no hay ningún estándar oficial para RESTful, aunque existen algunas recomendaciones sobre su implementación. Además, existe una librería que implementa este servicio (Guilcher, 2015) y también tiene versión para Android.

CAPÍTULO 4

4. DESCRIPCIÓN TÉCNICA DE LA APLICACIÓN PEDIATRÍA

En este capítulo se verá una descripción técnica de la aplicación PEDIATRÍA y también se hará un análisis funcional de su funcionamiento interno.

Se comenzará explicando la estructura de la base de datos de esta aplicación PEDIATRÍA en su primera versión. Se analizará cada tabla de la base de datos y se detallará que campos contiene y de qué tipo son.

Para facilitar la comprensión de este capítulo se ha hecho uso de la herramienta "MySQL Workbench". Con esta herramienta se ha dibujado la estructura de la base de datos y las relaciones que ésta contiene.

La explicación de las diferentes funcionalidades de las que consta la aplicación se ha realizado mediante casos de uso con los que se pretende que el lector sepa que pasos se siguen internamente para llevar a cabo cada una de las funcionalidades. En algunos casos, se ha creído conveniente dibujar además los correspondientes diagramas de flujos para visualizar funcionamiento de manera más clara y precisa.

4.1 Estructura de la base de datos

En este aparatado se muestra la estructura general de la base de datos de la aplicación PEDIATRÍA en su primera versión.

La base de datos sigue una estructura racional, en ella se han relacionado tablas entre sí. A continuación de la estructura general se detallan las estructuras de cada tabla individualmente y se explica para qué sirve cada una de ellas y de qué tipo son los campos usados en cada tabla.

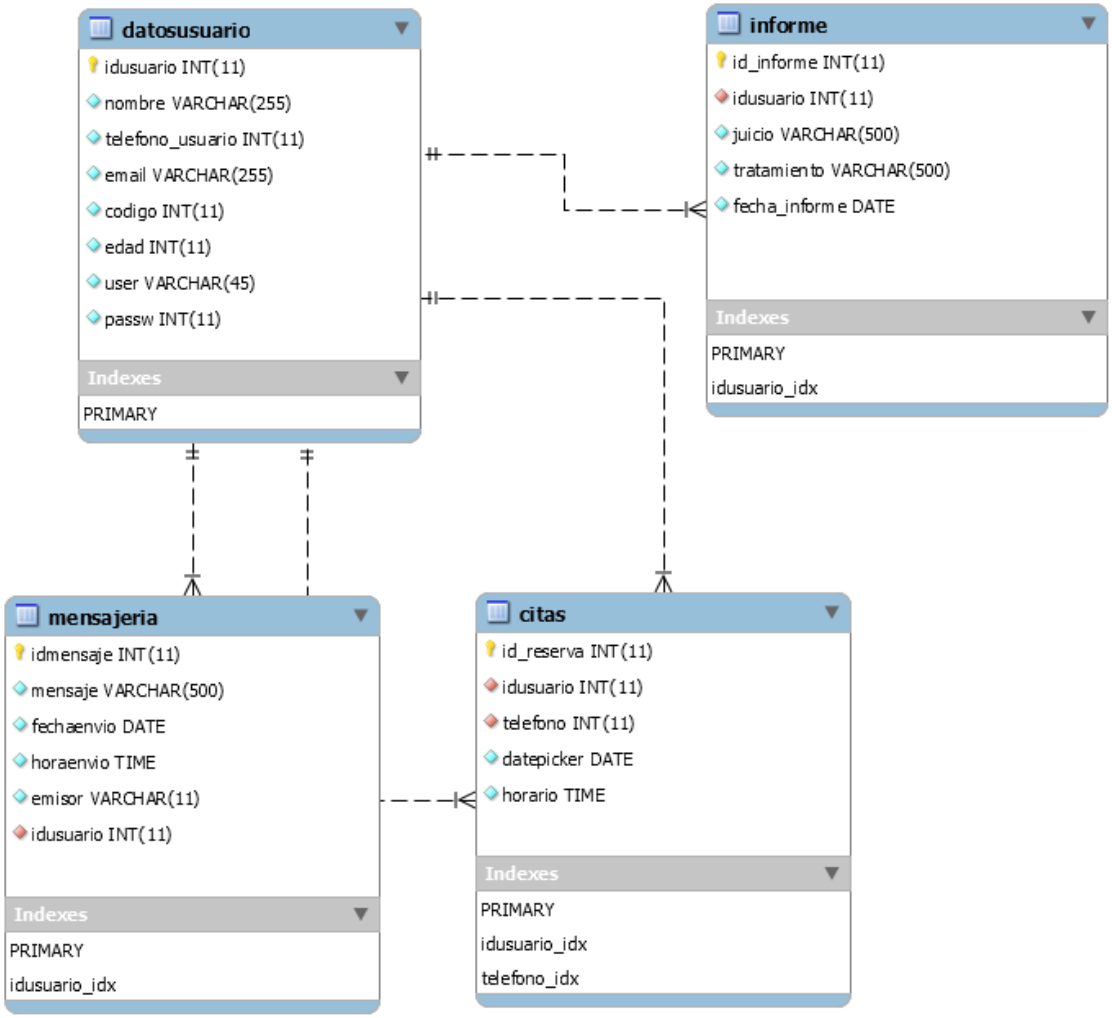


Ilustración 30. Base de datos de la aplicación PEDIATRÍA

Tabla citas:

Esta tabla contiene información de las reservas de las citas que realizan los usuarios de la aplicación PEDIATRÍA. Cada usuario, va a poder realizar las reservas de citas que desee, siempre que la hora y el día elegidos no hayan sido reservados antes por otro usuario.

Para poder hacer uso de esta funcionalidad de reserva de citas, es necesario que el usuario acceda con su nombre de usuario y su contraseña.

A continuación se describen los campos que son utilizados en esta tabla:

- ***id_reserva*** **:** Identificador único que sirve para identificar cada reserva realizada con éxito.
- ***idusuario*****:** Identificador del usuario que realiza la reserva, este identificador sirve para identificar en la tabla "datosusuario" al usuario que realiza la reserva.
- ***teléfono*****:** Teléfono de contacto del usuario que realiza la reserva.

- ***datepicker*:** Fecha que el usuario elige en la reserva de su cita.
- ***horario*:** Hora que el usuario elige para la cita.

Tabla citas				
Campo	**Tipo**	**Nulo**	**Predeterminado**	**Extra**
id_reserva	int (11)	No	Ninguna	AUTO_INCREMENT
idusuario	int (11)	No	Ninguna	
telefono	int (11)	No	Ninguna	
datepicker	date	No	Ninguna	
horario	time	No	Ninguna	

Tabla 5. Tabla de la BBDD: citas

Tabla datosusuario:

Esta tabla contiene los datos de los usuarios registrados en la base de datos y que tienen acceso a todas las funcionalidades de la aplicación.

Una vez que un cliente de su clínica pediátrica le comunica al pediatra que quiere poder hacer uso de todas las funcionalidades de dicha aplicación, el pediatra, o la persona encargada de mantener la base de datos actualizada, incluirá un nuevo registro en la tabla "datosusuario" con la información de contacto del cliente que lo solicite. El pediatra, o la persona encargada de mantener dicha tabla actualizada, asignarán al cliente un nombre de usuario (*login*) y una contraseña con la que podrá acceder a todas las funcionalidades de la aplicación PEDIATRÍA.

A continuación se describen los campos que son utilizados en esta tabla:

- ***idusuario:*** Identificador único de cada usuario registrado en el sistema.
- ***nombre:*** Nombre y apellidos del usuario.
- ***telefono_usuario:*** Teléfono de contacto del usuario.
- ***email:*** Correo electrónico del usuario.
- ***código:*** Otro identificador del usuario.
- ***edad:*** Edad del usuario.
- ***user:*** Nombre o *login* que junto con una contraseña, el usuario usará para iniciar sesión en la aplicación PEDIATRÍA.
- ***passw:*** Contraseña que junto con un nombre de usuario, el usuario usará para iniciar sesión en la aplicación PEDIATRÍA.

Tabla datosusuario					
Campo	**Tipo**	**Cotejamiento**	**Nulo**	**Predeterminado**	**Extra**
idusuario	int (11)		No	Ninguna	AUTO_INCREMENT
nombre	varchar (255)	utf8_spanish_ci	No	Ninguna	
telefono_usuario	int (11)		No	Ninguna	
email	varchar (255)	utf8_spanish_ci	No	Ninguna	
codigo	int (11)		No	Ninguna	
edad	int (11)		No	Ninguna	
user	varchar (11)	utf8_spanish_ci	No	Ninguna	
passw	int (11)		No	Ninguna	

Tabla 6. Tabla de la BBDD: datosusuario

Tabla informe:

Esta tabla contiene información de las citas que han tenido los pacientes con su doctor. En ella, el pediatra, o persona que mantenga actualizada la base de datos, incluirá la información de la consulta que tuvo el paciente. Estos datos son, por ejemplo, los síntomas que el paciente explicó al doctor, tratamiento que le fue recomendado por él, o la fecha en la que se tuvo la cita. Los usuarios registrados en la aplicación PEDIATRÍA podrán hacer uso de esta información accediendo al apartado de consultar su informe, con su nombre de usuario y su contraseña.

A continuación se describen los campos que son utilizados en esta tabla:

- ***id_informe:*** Identificador único de cada consulta que un paciente ha tenido con su doctor.
- ***idusuario:*** Identificador del usuario que realiza la reserva, este identificador sirve para identificar en la tabla "datosusuario" al usuario que realiza la reserva.
- ***juicio:*** Síntomas que el paciente ha comunicado al doctor en su cita (juicio crítico).
- ***tratamiento:*** Tratamiento a seguir que el doctor recomendó en la cita al usuario.
- ***fecha_informe:*** Fecha en la cual se realizó la consulta al paciente.

Tabla informe					
Campo	**Tipo**	**Cotejamiento**	**Nulo**	**Predeterminado**	**Extra**
id_informe	int (11)		No	Ninguna	AUTO_INCREMENT
isuduario	int (11)		No	Ninguna	
juicio	varchar (500)	utf8_spanish_ci	No	Ninguna	
tratamiento	varchar (500)	utf8_spanish_ci	No	Ninguna	
fecha_informe	date		No	Ninguna	

Tabla 7. Tabla de la BBDD: informe

4.2 Relaciones entre las tablas de las bases de datos

En este segundo apartado se muestran las relaciones que existen entre las tablas de datos detalladas anteriormente. Estas relaciones se han dibujado haciendo uso de la herramienta ya mencionada, “MySQL Workbench” puesto que facilita la visualización con colores de las relaciones existentes entre las tablas.

En la primera figura, se observa que la tabla “datosusuario” está relacionada con las otras dos tablas, informe y citas. En la tabla “datosusuario” la clave primario es el campo llamado: idusuario.

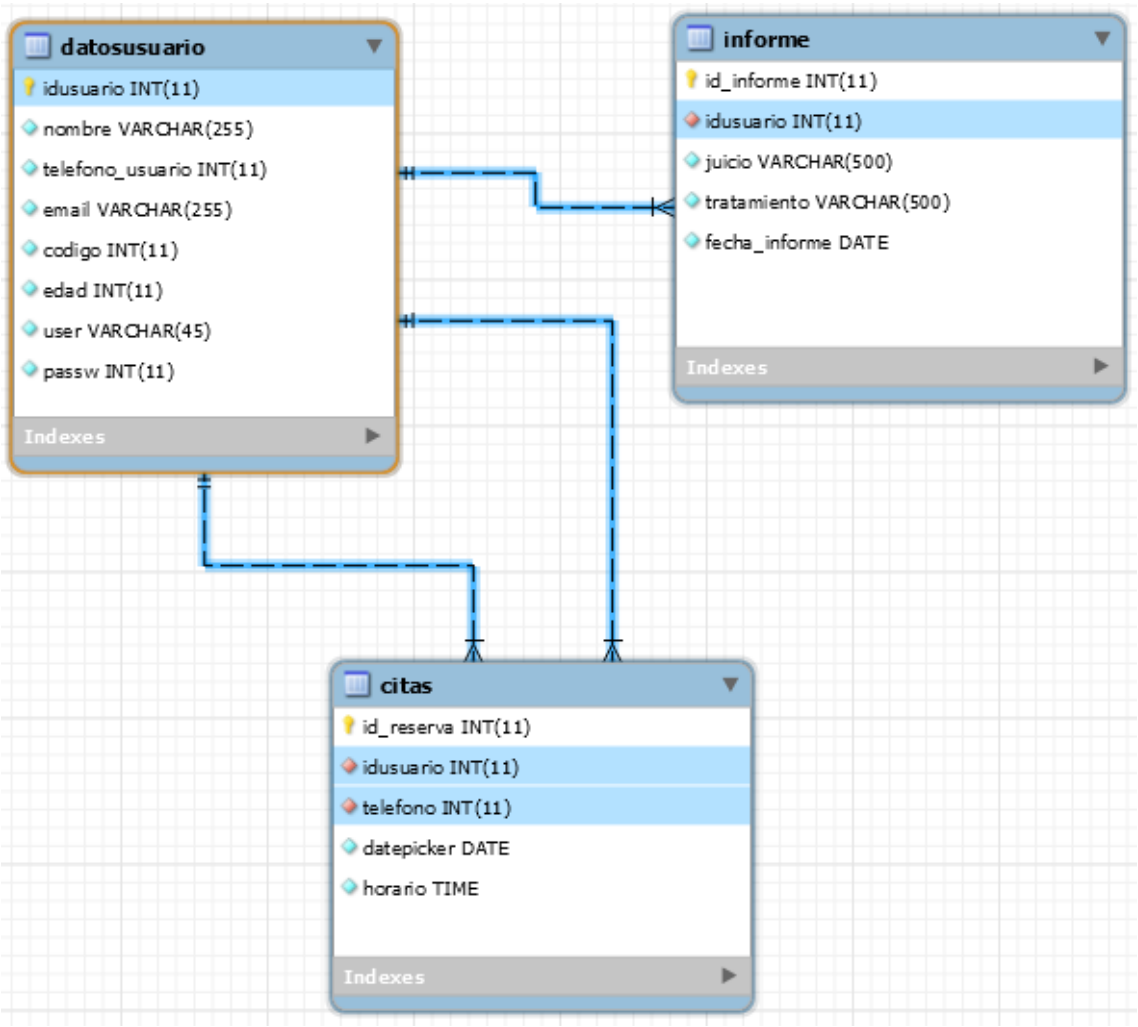


Ilustración 31. Relación BBDD I

La tabla “informe” tiene una relación con la tabla “datosusuario” a través del campo llamado idusuario. La clave principal de la tabla “informe” es el campo id_informe.

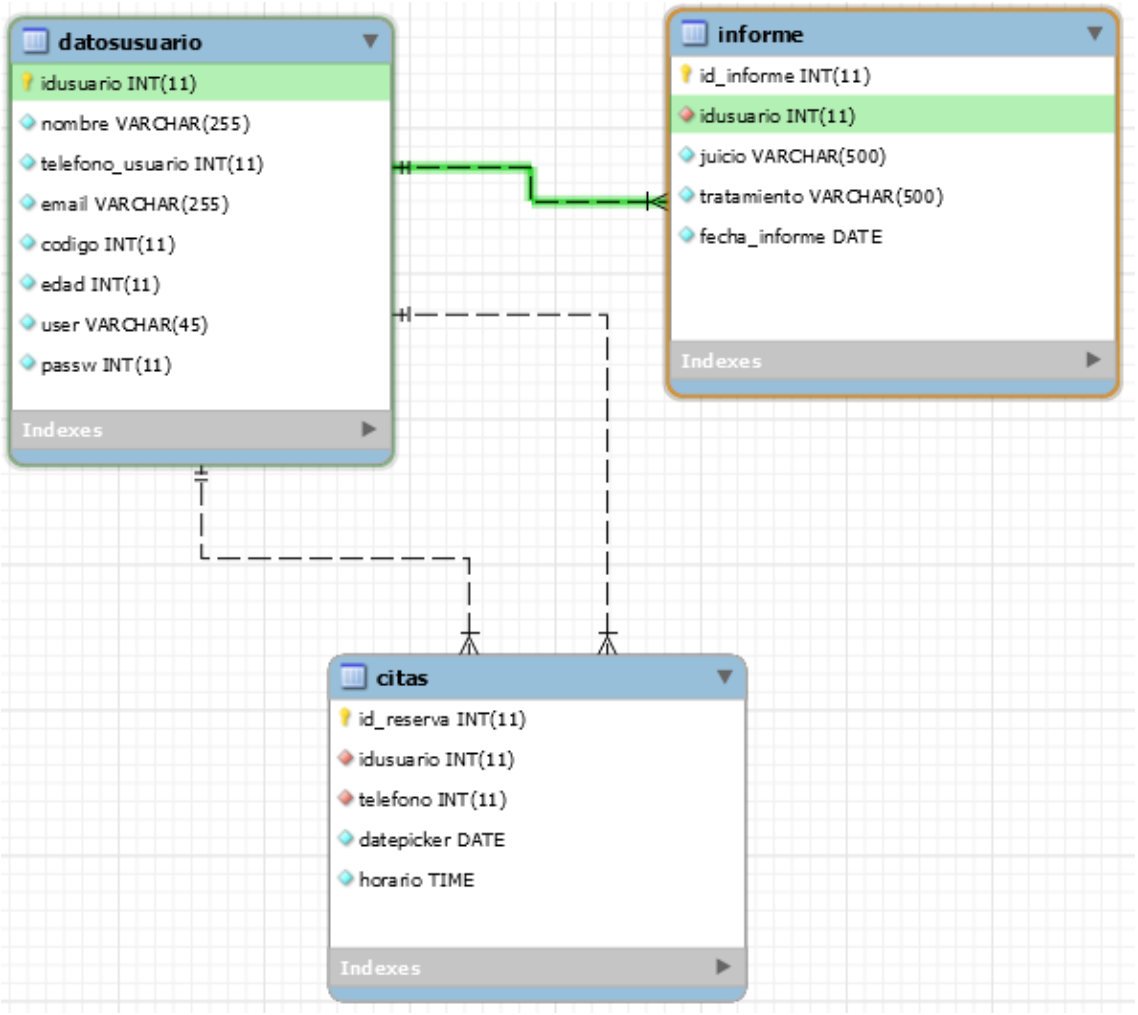


Ilustración 32. Relación BBDD II

De la misma manera, la tabla "citas" se relaciona con la tabla "datosusuario" a través del campo idusuario. La clave principal de esta tabla "citas" es el campo id_reserva.

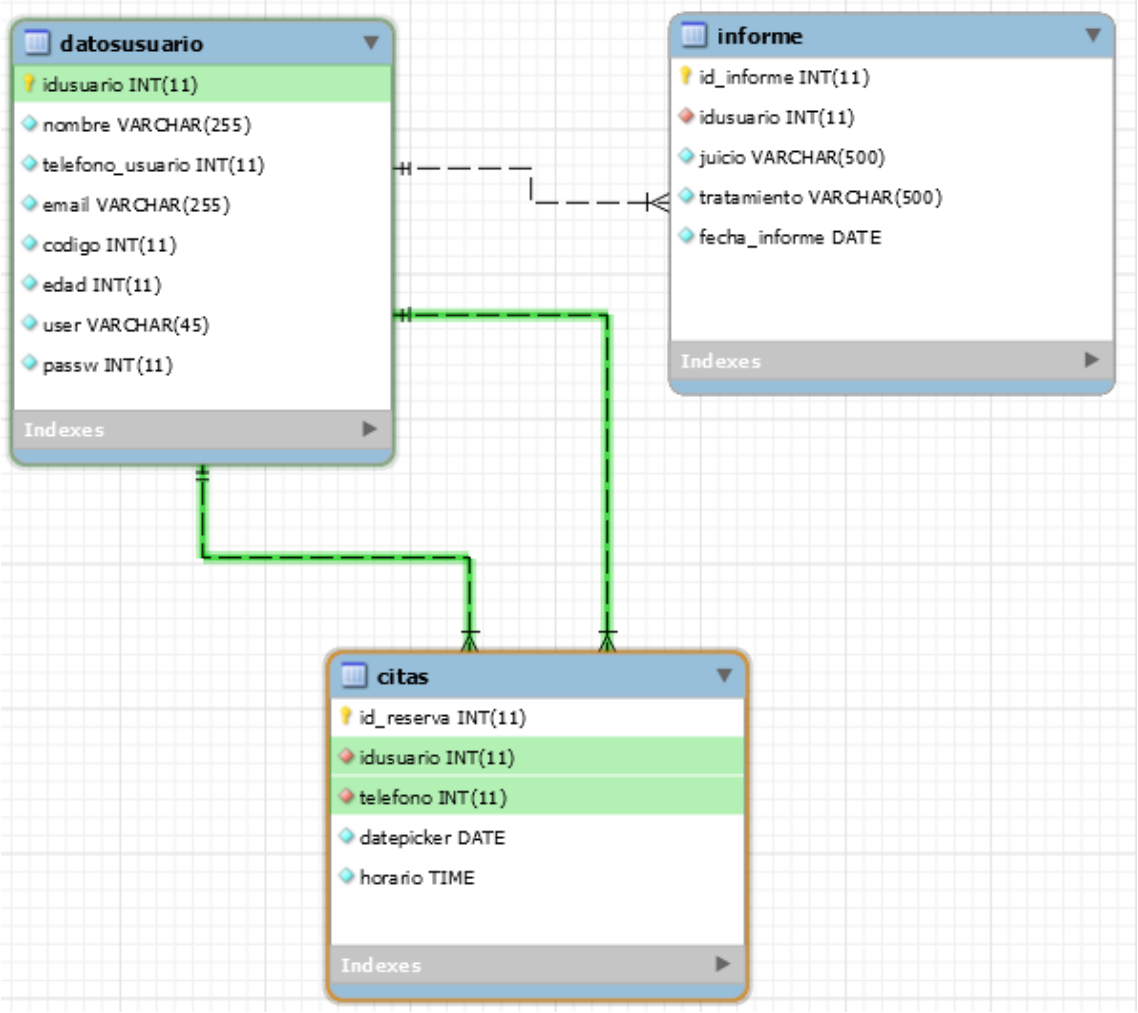


Ilustración 33. Relación BBDD III

4.3 Casos de uso y diagramas de actividad de las distintas funcionalidades de la aplicación.

En esta sección se analizan mediante tablas los casos de uso correspondientes a cada funcionalidad de la aplicación PEDIATRÍA.

Para comenzar, se muestra una ilustración de un diagrama UML en el que se pueden diferenciar todas las funcionalidades de las que consta dicha aplicación.

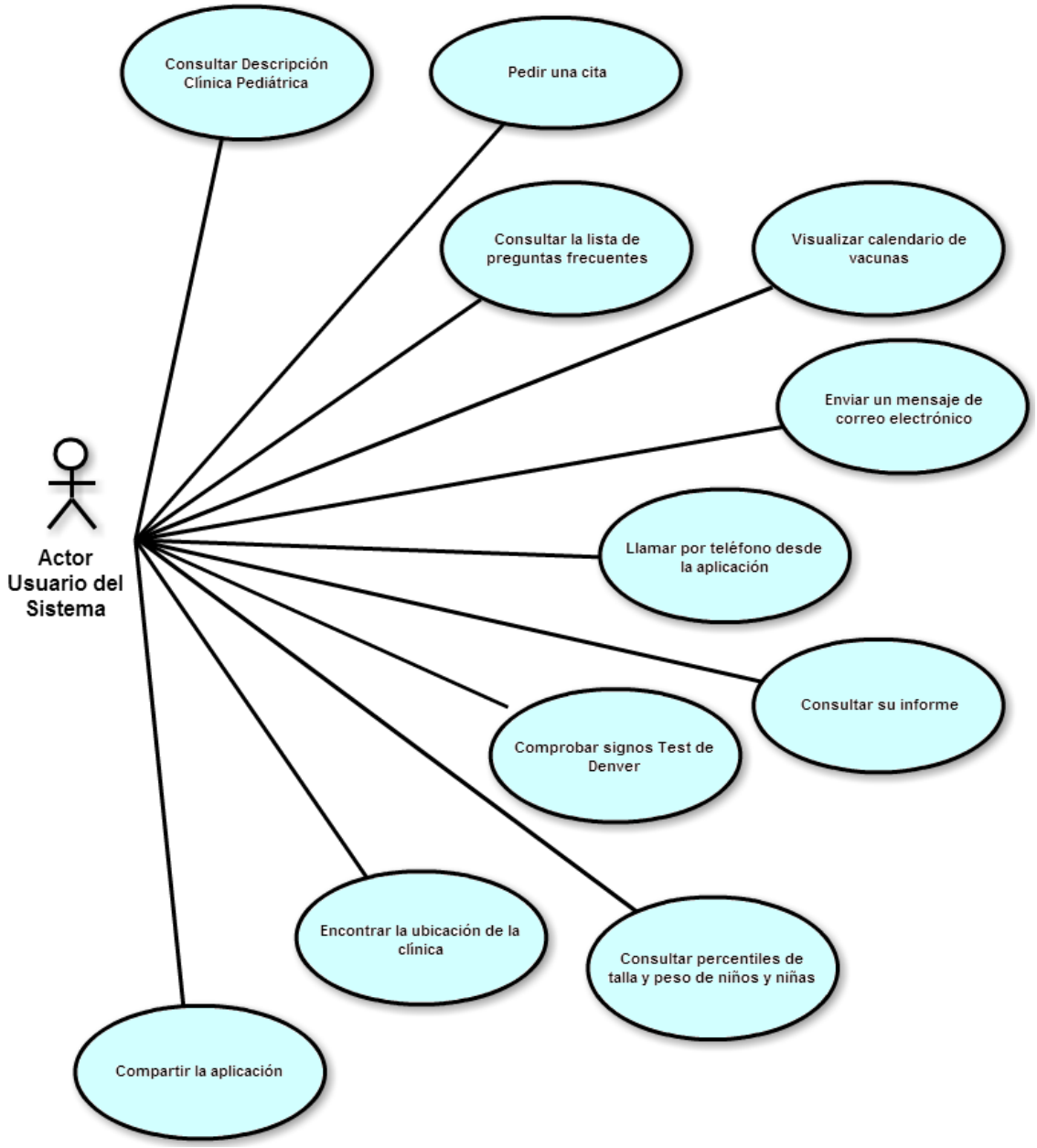


Ilustración 34. Diagrama UML Funcionalidades de la aplicación

A continuación se van describiendo todos los casos de uso que se observan en el diagrama UML anterior a través de tablas con los que se pretende se facilite la comprensión de éstos.

Para las funcionalidades más complejas se ha decidido mostrar además de la tabla del caso de uso, el diagrama de actividad correspondiente. Con estos diagramas de actividad se podrá visualizar de manera más clara y rápida la secuencia de pasos llevada a cabo en dichas funcionalidades.

- ## ¿Quién somos?

ID	¿Quién somos?	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario visualiza una descripción de la clínica pediátrica	
Personal Involucrado	Usuario del sistema	
Precondición	No hay	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa sobre "¿Quién somos?"
	2	La aplicación muestra una pantalla en la que se describen las características de dicha clínica. El caso de uso finaliza.
Postcondición	Se ha accedido correctamente a la funcionalidad de ubicación. Se visualiza un mapa, el horario y los teléfonos.	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para visualizar una descripción de la clínica pediátrica.	

Tabla 8. Caso de uso ¿Quién somos?

- ## Inicio sesión:

ID	Inicia sesión	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario necesitará identificarse como un usuario registrado en el sistema para acceder a determinadas funcionalidades.	
Personal Involucrado	Usuario del sistema	
Precondición	El usuario debe haber recibido por parte del encargado de mantener actualizadas las bases de datos un nombre de usuario y una contraseña con la que poder iniciar sesión.	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario introduce su nombre de usuario
	2	El usuario introduce su contraseña
	3	El usuario pulsa sobre "Login"
Postcondición	Se ha accedido correctamente a la funcionalidad deseada.	
Alternativas	Si el inicio de sesión no se ha ejecutado correctamente, debido por ejemplo a una contraseña incorrecta, el caso de uso finaliza sin postcondición.	
Requisitos especiales	No hay	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para poder acceder a las funcionalidades que lo	

	requieren.

Tabla 9. Caso de uso Inicia sesión

- **Pide tu cita**

ID	Pide tu cita	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario podrá reservar una cita con el pediatra a través de la aplicación.	
Personal Involucrado	Usuario del sistema	
Precondición	1.1	El usuario que desee acceder a esta funcionalidad debe ser un usuario registrado en el sistema. El responsable de esta aplicación debe de haber facilitado al usuario que desea hacer uso de esta funcionalidad, un nombre de usuario y una contraseña con la que poder iniciar sesión.
	1.2	La conexión con la base de datos debe realizarse con éxito.
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa el botón "Pide tu cita".
	2	El usuario accede a una pantalla de inicio de sesión.
	3	El usuario introduce su nombre de usuario y su contraseña. Se realiza el inicio de sesión correctamente.
	4	El usuario rellena en el formulario los datos requeridos para la reserva de la cita. A continuación pulsa sobre "Enviar".
	5	La aplicación se conecta a la base de datos y consulta si para esa fecha y esa hora existe alguna reserva previamente realizada por otro usuario.
	6	En caso de que esté libre, se insertará la nueva reserva en la base de datos.
	7	Se mostrará por pantalla un mensaje confirmando que la reserva se ha realizado correctamente.
	8	El caso de uso finaliza.
Alternativas	Pasos	Acción
	1	El usuario pulsa el botón "Pide tu cita".
	2	El usuario accede a una pantalla de inicio de sesión.
	3	El usuario introduce su nombre de usuario y su contraseña. Se realiza el inicio de sesión correctamente.
	4	El usuario rellena en el formulario los datos requeridos para la reserva de la cita. A continuación pulsa sobre "Enviar".
	5	La aplicación se conecta a la base de datos y consulta si para esa fecha y esa hora existe alguna reserva previamente realizada por otro usuario.
	6	En caso de que esté ocupada, se buscará la siguiente hora más cercana que esté libre, y se hará la reserva con dicha hora.
	6.1	Si el día que el usuario eligió para la cita no dispone de más huecos libres, no se realizará ninguna reserva y se pedirá al usuario que escoja un nuevo día para su reserva. Empezaría un nuevo caso de uso de este tipo.
	7	Se mostrará por pantalla un mensaje confirmando que la reserva se ha realizado correctamente.

	8	El caso de uso finaliza.
Postcondición	La reserva de la cita se ha realizado correctamente.	
Requisitos especiales	El usuario debe haber iniciado sesión satisfactoriamente con su nombre de usuario y su contraseña.	
Comentarios	En los casos en los que el usuario no complete todos los datos solicitados por la aplicación o el usuario salga de la funcionalidad sin pulsar sobre "Enviar", el caso de uso finalizaría sin éxito.	
Importancia	Este caso de uso es muy útil para reservar una cita de manera fácil y rápida.	

Tabla 10. Caso de uso Pide tu cita

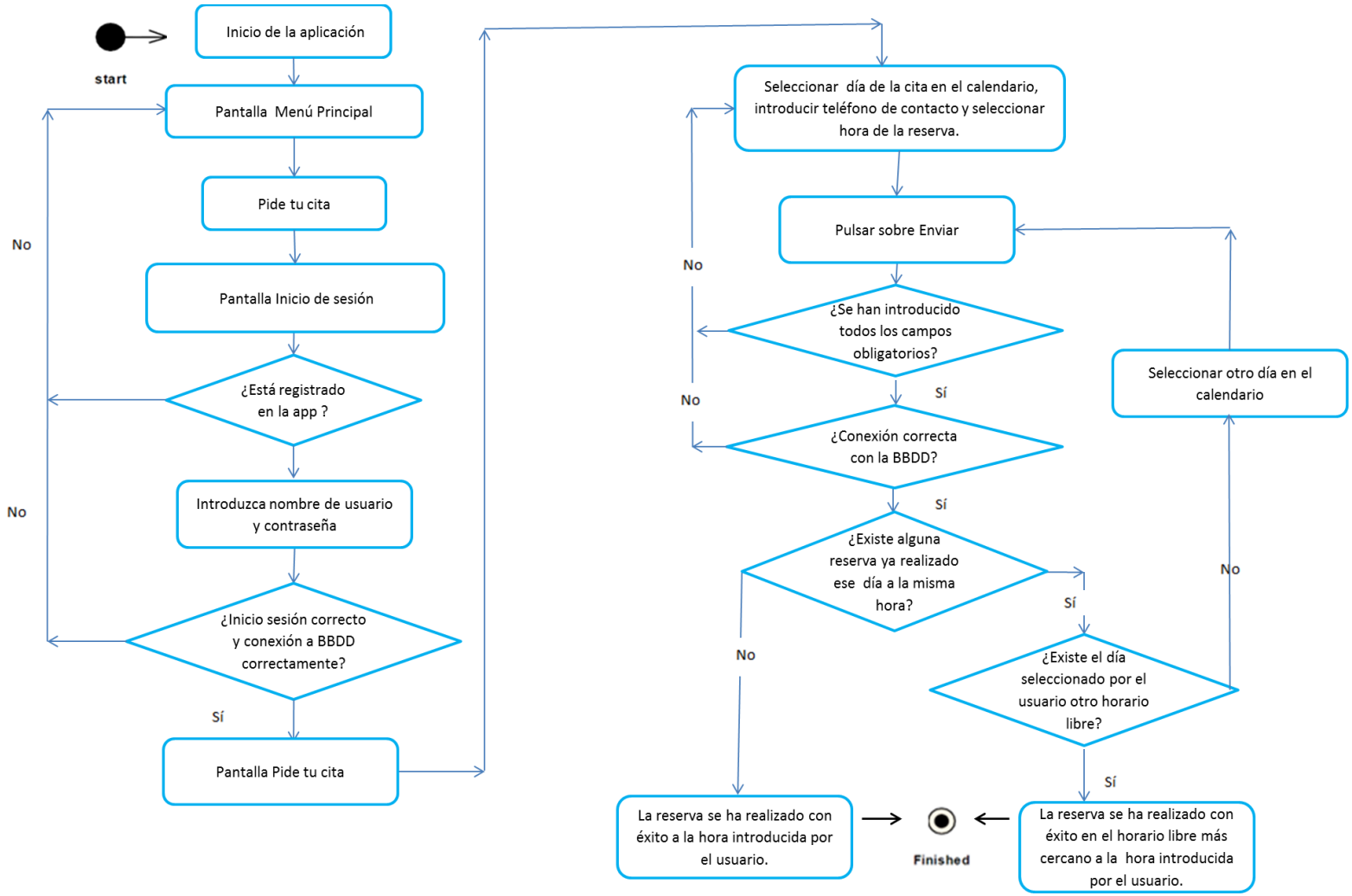


Ilustración 35. Diagrama de actividad Pide tu cita

- ## Lactancia: Preguntas frecuentes FAQ

ID	Lactancia: Preguntas frecuentes FAQ	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario visualizará una serie de preguntas frecuentes sobre Lactancia	
Personal Involucrado	Usuario del sistema	
Precondición	No hay	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa sobre "¿Lactancia FAQ?"
	2	La aplicación muestra una pantalla en la que se describen algunas de las preguntas frecuentes sobre lactancia que más se consultan en la página web de la Asociación española de Pediatría. El caso de uso finaliza.
Postcondición	Se ha accedido correctamente a la funcionalidad de preguntas frecuentes sobre lactancia.	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para visualizar las consultas más frecuentes sobre lactancia.	

Tabla 11. Caso de uso Lactancia: Preguntas frecuentes FAQ

- ## Calendario de vacunas (Castilla y León)

ID	Calendario de vacunas	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario visualiza las vacunas recomendadas en Castilla y león para los niños y niñas entre 0 y 14 años	
Personal Involucrado	Usuario del sistema	
Precondición	No hay	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa sobre "Calendario de vacunas"
	2	La aplicación muestra una pantalla en la que se muestran las vacunas recomendadas en castilla y león para los niños y niñas de entre 0 y 14 años. El caso de uso finaliza.
Postcondición	Se ha accedido correctamente a la funcionalidad del calendario de vacunas en castilla y león.	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para visualizar las vacunas recomendadas en Castilla y león	

Tabla 12. Caso de uso Calendario de vacunas (Castilla y León)

- **Consulta tu informe**

ID	Consulta tu informe	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario visualizará la información de los informes de las consultas que ha tenido con el doctor en la clínica pediátrica. Visualizará las fechas de las consultas, los síntomas que se describieron al doctor, y el tratamiento sugerido por él.	
Personal Involucrado	Usuario del sistema	
Precondición	1.1	El pediatra o responsable de mantener actualiza la base de datos, debe incluir en la base de datos la información de las consultas que tenga con los diferentes pacientes.
	1.2	El usuario que desee acceder a esta funcionalidad debe ser un usuario registrado en el sistema. El responsable de esta aplicación debe de haber facilitado al usuario que desea hacer uso de esta funcionalidad, un nombre de usuario y una contraseña con la que poder iniciar sesión.
	1.3	La conexión con la base de datos debe realizarse con éxito
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa el botón "Consulta tu informe".
	2	El usuario accede a una pantalla de inicio de sesión.
	3	El usuario introduce su nombre de usuario y su contraseña. Se realiza el inicio de sesión correctamente.
	4	La aplicación se conecta a la base de datos y extrae la información de las consultas del paciente que está accediendo a la aplicación.
	5	La aplicación muestra por pantalla la información de las consultas que el paciente ha tenido con el doctor en dicha clínica pediátrica. El caso de uso finaliza.
Postcondición	Se visualiza la información de las consultas que el paciente ha tenido con el doctor.	
Requisitos especiales	El usuario debe haber iniciado sesión satisfactoriamente con su nombre de usuario y su contraseña.	
Comentarios	No hay.	
Importancia	Este caso de uso es vital para consultar los informes que el doctor ha tenido con el usuario del sistema.	

Tabla 13. Caso de uso Consulta tu informe

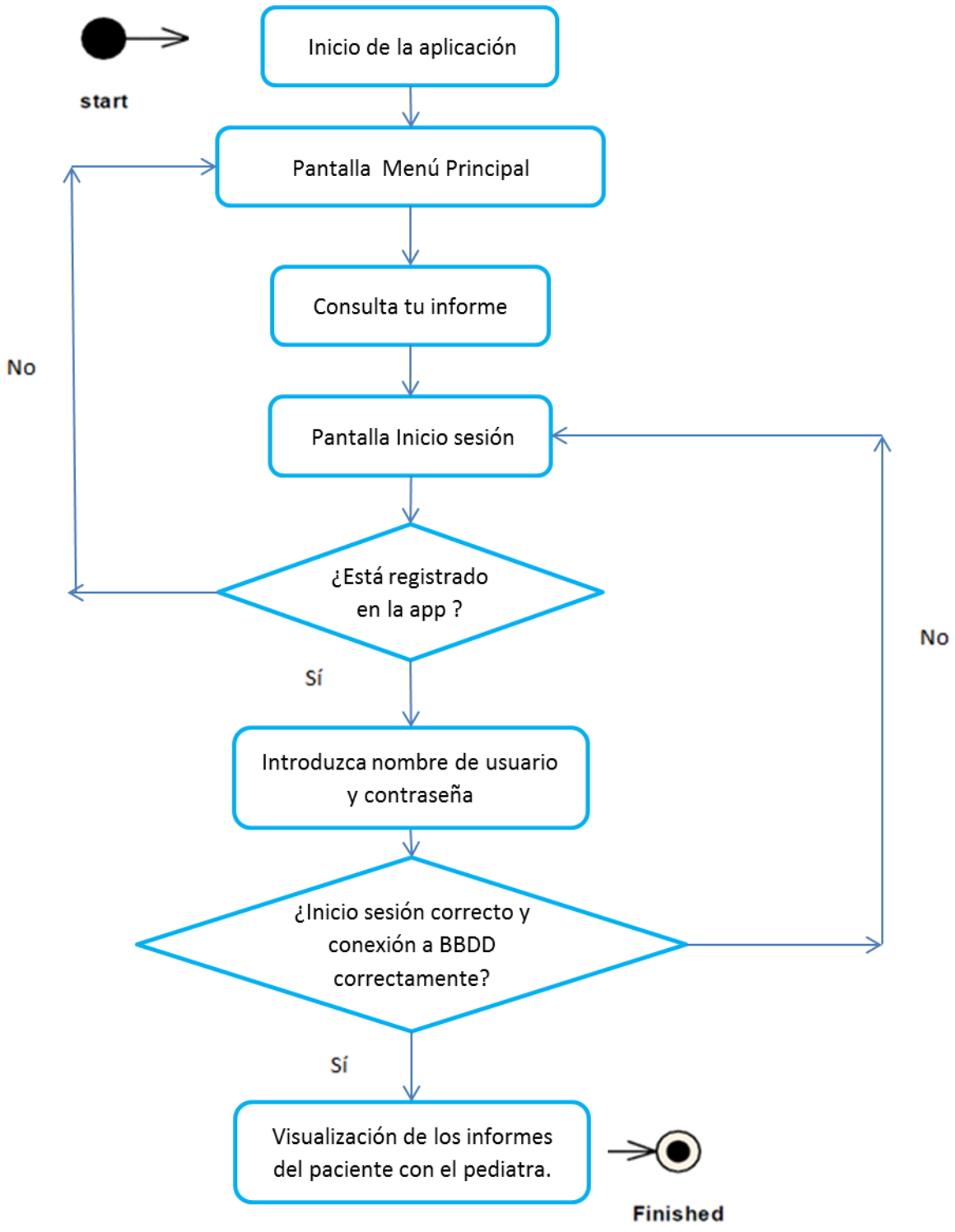


Ilustración 36. Diagrama de actividad Consulta tu informe

- **Test de Denver**

ID	Test de Denver

Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario visualiza los signos característicos del Test de Denver	
Personal Involucrado	Usuario del sistema	
Precondición	No hay	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa sobre "Test de Denver"
	2	La aplicación muestra una pantalla en la que se muestran algunos signos que niños y niñas no deberían, a priori, tener a determinadas edades. El caso de uso finaliza.
Postcondición	Se ha accedido correctamente a la funcionalidad de los signos del Test de Denver, esta funcionalidad es meramente informativa y siempre habrá que consultar con un especialista	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para visualizar las características del Test de Denver.	

Tabla 14. Caso de uso Test de Denver

- ## Percentiles

ID	Percentiles	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario accede a los diferentes percentiles de talla y peso para niños y niñas.	
Personal Involucrado	Usuario del sistema	
Precondición	No hay	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa sobre "Percentiles"
	2	La aplicación muestra una nueva pantalla con un nuevo menú para la elección del percentil que se desee visualizar. Se elige entre talla o peso, para niño o para niña.
	3	El usuario pulsa sobre el percentil que desea visualizar
	4	El caso de uso finaliza.
Postcondición	Se ha accedido correctamente a la funcionalidad de ubicación. Se visualiza un mapa, el horario y los teléfonos.	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para acceder a la visualización de los diferentes percentiles existentes en la aplicación.	

Tabla 15. Caso de uso Percentiles

- ## Ubicación (¿Dónde estamos?)

ID	Ubicación	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario visualiza en un mapa la localización de la clínica pediátrica. Se visualizan también teléfonos de urgencias y el horario.	
Personal Involucrado	Usuario del sistema	
Precondición	No hay	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa sobre "Ubicación"
	2	La aplicación muestra la pantalla correspondiente a esta funcionalidad. El caso de uso finaliza.
Postcondición	Se ha accedido correctamente a la funcionalidad de ubicación. Se visualiza un mapa, el horario y los teléfonos.	
Comentarios	No hay	
Importancia	Este caso de uso es imprescindible para acceder a la ubicación y contacto.	

Tabla 16. Caso de uso Ubicación

- ## Enviar mensaje

ID	Enviar mensaje	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario podrá enviar una consulta al pediatra enviando un mensaje de correo electrónico a través de la aplicación.	
Personal Involucrado	Usuario del sistema	
Precondición	1.1	El usuario debe rellenar todos los campos solicitados por la aplicación.
	1.2	La conexión con el servidor debe realizarse de manera satisfactoria para que el caso de uso finalice con éxito.
Escenario Principal de éxito	Pasos	Acción
	1	El usuario pulsa el botón "Mensaje"
	2	El usuario rellena todos los campos requeridos
	3	El usuario pulsa el botón "Mensaje"
	4	Aparece una alerta por pantalla indicando que el mensaje ha sido enviado correctamente. El caso de uso finaliza.
Postcondición	Se ha enviado un mensaje al correo electrónico del pediatra	
Alternativas	Si el mensaje no ha sido enviado con éxito aparecerá por pantalla un mensaje indicando que el mensaje no ha sido enviado. El caso de uso finaliza sin ninguna postcondición.	

Requisitos especiales	No hay
Comentarios	El caso de uso se da cuando el usuario desea enviar un correo electrónico de consulta al pediatra.
Importancia	Este caso de uso es muy importante puesto que permite a cualquier usuario de la aplicación tener una comunicación directa con el pediatra.

Tabla 17. Caso de uso Enviar Mensaje

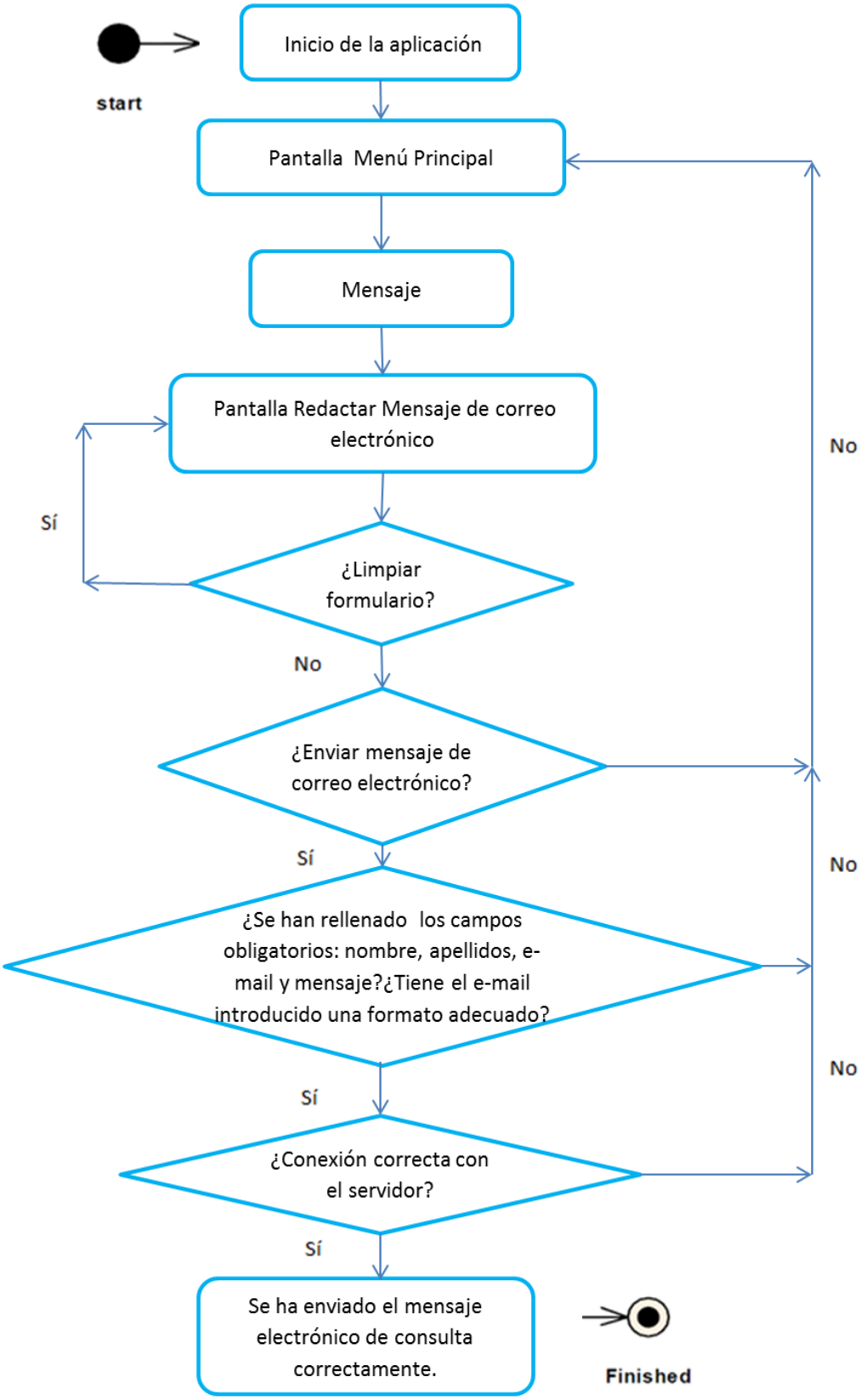


Ilustración 37. Diagrama de actividad Enviar mensaje

- **Limpiar formulario**

ID	Limpiar formulario mensaje	
Versión	1.0	
Fecha	10 de Enero	
Autor	Julia Santo Domingo Gómez	
Descripción	El usuario elimina todos los componentes que forman la plantilla, dejándola vacía de forma rápida y sencilla.	
Personal Involucrado	Usuario del sistema	
Precondición	El usuario debe haber rellenado alguno de los campos requeridos para que el caso de uso tenga sentido.	
Escenario Principal de éxito	Pasos	Acción
	1	El usuario rellena alguno de los campos del formulario para el envío de un mensaje electrónico
	2	El usuario pulsa el botón "Limpiar"
	3	El sistema elimina todos los componentes de la plantilla y la muestra vacía. El Caso de Uso finaliza.
Postcondición	Se ha eliminado la información que el usuario había introducido en los diferentes campos solicitados en el formulario.	
Requisitos especiales	No hay	
Comentarios	Este caso de uso se da cuando el usuario está rellenado los datos del formulario para el envío de un correo electrónico de consulta al pediatra.	
Importancia	Este caso de uso es útil para borrar todos los campos de una forma fácil y rápida.	

Tabla 18. Caso de uso Limpiar formulario

4.4 Estructura de ficheros

Estructura de la organización de los ficheros que están subidos en el servidor:

Las imágenes y los ficheros con extensión .php que dan respuesta a las peticiones que los usuarios hacen desde la aplicación PEDIATRÍA deben alojarse en un servidor. En este caso, estos ficheros se encuentran en /domains/public_html/Pediatria.

Los ficheros con extensión .php que se encuentran alojados en el servidor dentro de la carpeta denominada “Pediatría” son los siguientes:

consultainforme.php: Fichero con extensión .php que se usa cuando el usuario desea consultar la información de sus informes con el doctor. Consulta la base de datos y extrae la información solicitada.

enviar_mensaje.php: Fichero con extensión .php en el que se realiza el envío de un mensaje electrónico desde la aplicación a la dirección de correo contenida en este fichero (actualmente

las consultas que se envíen mediante correo electrónico desde la aplicación PEDIATRÍA llegarán al buzón de entrada de mi correo).

identificación.php: Fichero con extensión .php con el que se realiza el inicio de sesión del usuario en la aplicación. Con este fichero se harán las comprobaciones necesarias en la base de datos del nombre de usuario y su contraseña. Consulta información en la base de datos.

pidetucita.php: Fichero con extensión .php que se usa cuando el usuario de la aplicación desea realizar la reserva de una cita. Consulta e inserta información en la base de datos.

Estructura del fichero usado en *PhoneGapBuild*:

El fichero comprimido, con extensión .zip, que se sube en la página web de *PhoneGapBuild* y que genera el fichero que se instala en los diferentes móviles para el uso de la aplicación contiene una estructura de carpetas como el que se muestra a continuación.

Fichero comprimido que se sube a *PhoneGapBuild* contiene:

- Carpeta /www:
  - Carpeta: css → Contiene las hojas de estilo que se usan para dar forma y color a la aplicación PEDIATRÍA.
    - Carpeta: themes → Contiene más hojas de estilo usadas y la imagen del fondo de la aplicación.
      - ❖ Carpeta: images → Contiene el icono de la aplicación.
        - Carpeta: icons-png →Contiene diferentes iconos que son predefinidos de Cordova.
  - Carpeta: img →Contiene todas las imágenes que se usan en la aplicación PEDIATRÍA.
  - Carpeta: js → Contiene los ficheros con extensión .js.
  - Carpeta: plugins → Contiene la información de los plugins que se usan en esta aplicación.
  - Ficheros .html →Todos los ficheros con extensión .html que se usan en la aplicación y que corresponden a las diferentes pantallas que ésta contiene.
  - Config.xml → Fichero de configuración en el que se le indica a la herramienta *PhoneGapBuild* como debe ser el funcionamiento de la aplicación. Por ejemplo, en ella se incluyen los tamaños de las distintas resoluciones de las pantallas de los teléfonos móviles, el icono de la aplicación, la imagen que se usa cuando se inicia la aplicación,…

Esta estructura de carpetas y ficheros que luego se comprimirá para subirlo a *PhoneGapBuild* debe seguir un patrón definido por dicha herramienta para que *PhoneGap* pueda generar correctamente el archivo que instalará la aplicación en los distintos dispositivos.

4.5 Conexión de la aplicación móvil con el servidor

La implementación de los servicios web, es decir, las conexiones necesarias entre la aplicación móvil y el servidor web, se realiza mediante un servicio *RESTful*. A través de conexiones HTTP se podrán realizar comunicaciones en las que cada mensaje va a contener toda la información necesaria para comprender una petición o una respuesta.

El protocolo HTTP: es un protocolo de la capa de aplicación que se emplea para la transferencia de información entre sistemas, de manera clara y rápida. Es el protocolo que utiliza el *World-Wide Web* desde 1990.

En las peticiones se debe incluir el método que se aplica al recurso que se solicita. Los principales métodos que existen son:

Método GET

Este método requiere la devolución de información al cliente identificada por la URI. Si la URI se refiere a un proceso que produce información se devuelve la información, y no la fuente del proceso.

Método HEAD

Se trata de un método similar a GET, con la diferencia de que el servidor no tiene que devolver el contenido sino solo las cabeceras. Estas cabeceras deberían ser las mismas que las que se devolverían empleando el método GET.

Método POST

El método POST se utiliza para realizar peticiones en las que el servidor acepta el contenido de la petición como nuevos parámetros del recurso pedido. Este método se creó para cubrir funciones como enviar mensajes a grupos de usuarios, dar un bloque de datos como resultado de un formulario a un proceso o añadir nuevos datos a una base de datos.

CAPÍTULO 5

5. MANUAL DE USUARIO

El objetivo de este capítulo es dar a conocer la aplicación PEDIATRÍA implementada en este TFG. Se describirá la aplicación y todas sus funcionalidades.

Mediante capturas de pantalla realizadas en un móvil: LG 9 Optimus con versión de Android: 4.4.2, se seguirán una serie de pasos descriptivos para que cualquier usuario que quiera hacer uso de la aplicación pueda realizarlo siguiendo este manual de usuario.

A lo largo del manual de usuario, se incluirán también algunas fotografías de la aplicación instalas en diferentes dispositivos. Así podrá observarse como se ve la aplicación instalada en un móvil realmente.

Esta aplicación fue implementada mientras cursaba la asignatura: Prácticas del grado de Tecnologías de las telecomunicaciones, en la empresa de acogida SYLTEC.

Como introducción, la aplicación PEDIATRÍA es una aplicación que ha sido diseñada para una clínica pediátrica ideal, mediante la cual los padres o tutores de los niños, pacientes de esta clínica, podrán beneficiarse de todas sus funcionalidades.

Toda la información médica/pediátrica que se ha usado para la implementación de esta aplicación ha sido obtenida de la página web de la Asociación Española de Pediatría (http://www.aeped.es/).

Esta aplicación, dispone en su mayoría de funcionalidades abiertas que cualquier usuario que descargara la aplicación podría usar sin necesidad de ser cliente específico de esa clínica pediátrica. Sin embargo, hay alguna funcionalidad para la que si se requiere ser cliente de esa clínica y se necesita haber sido dado de alta previamente por el pediatra en la base de datos de la aplicación. Una vez que el usuario es dado de alta por el pediatra en la base de datos, se concederá al usuario un nombre de usuario y una clave para que pueda acceder a las funcionalidades específicas de la aplicación.

Estas funcionalidades a las cuales sólo se puede acceder con un usuario y una contraseña otorgada por el pediatra son: la solicitud de una cita para consulta y el acceso al historial del paciente.

Para salir de cualquiera de las funcionalidades y volver al menú principal bastará con pulsar sobre el botón Menú orientado en la franja azul inferior.

5.1 Escritorio: Icono de la aplicación

Una vez descargada e instalada la aplicación desde el *market place* o mediante el código QR que se muestra cuando se sube el fichero .zip a la página web de PhoneGapBuild, aparecerá en el escritorio del móvil el icono de la aplicación PEDIATRÍA que ha sido implementada en este TFG:

Ilustración 38. Icono de la aplicación Pediatría

Si pulsamos sobre el icono de la aplicación, aparecerá durante unos segundos una pantalla de inicio como la siguiente. Esta opción de *Splash Screen* ha sido posible gracias al plugin *"org.apache.cordova.splashscreen"* ofrecido por el *framework* usado: PhoneGapBuild.

Ilustración 39. Splash Screen: Imagen que se muestra al abrir la aplicación Pediatría

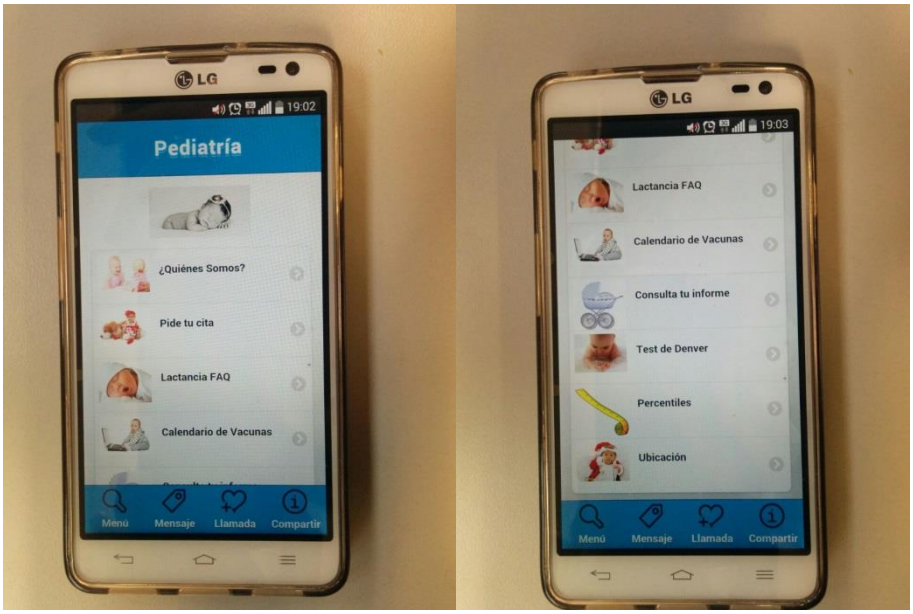
5.2 Pantalla inicio de la aplicación: Menú principal

Tras la imagen de carga anterior, se mostrará el menú principal de la aplicación desde el que se podrá acceder a todas sus funcionalidades. Como se verá en todas las pantallas de la aplicación siempre habrá:

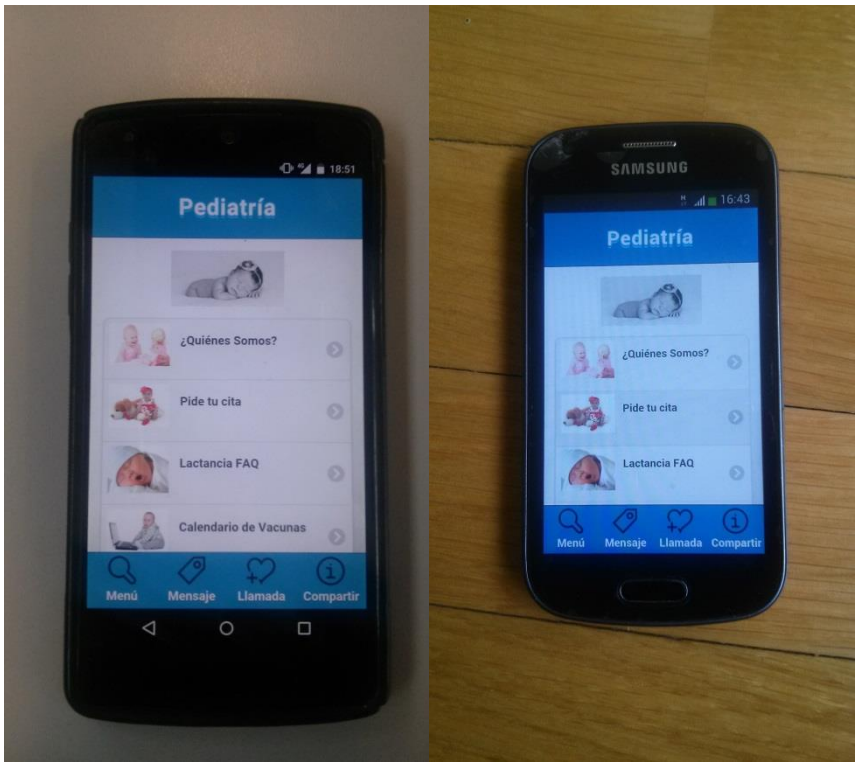
- Una cabecera azul en la parte superior, en la cual se mostrará la funcionalidad en la que se encuentra el usuario en cada momento
- Un menú fijo en la parte inferior de cada pantalla en el que siempre habrá cuatro opciones disponibles:
  - Menú: para volver a la pantalla del menú principal
  - Mensaje: para enviar un mail al pediatra
  - Llamada: es una acceso directo para llamar al teléfono de urgencias
  - Compartir: para compartir la aplicación en redes sociales

El menú principal se muestra de la siguiente forma:

Ilustración 40. Menú principal de la aplicación

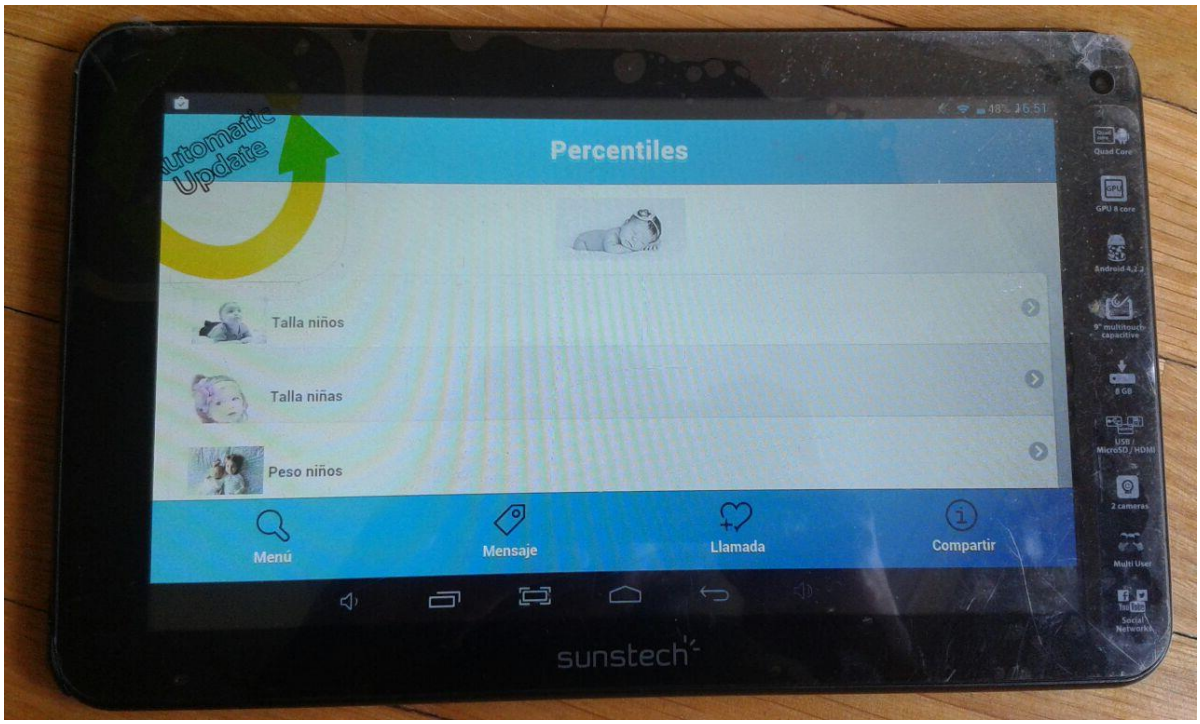


Fotografía 1. Pantalla Menú en LG 9 Optimus



Fotografía 2. Pantalla Menú en Nexus 5 (izquierda) y Samsung GT S7580 (derecha)

Fotografía 3. Menú vertical en Tablet Sunstech



Fotografía 4. Menú horizontal en Tablet Sunstech

Se observa que la aplicación tiene las siguientes funcionalidades:

- ¿Quién somos?
- Pide tu cita
- Lactancia: Preguntas frecuentes FAQ
- Calendario de vacunas (Castilla y león)
- Consulta tu informe (Consulta tus últimas visitas al pediatra y recomendaciones)
- Test de Denver
- Percentiles
- Ubicación (¿Dónde estamos?)

A continuación se describe el funcionamiento de cada una de las funcionalidades disponibles:

5.3 Pantalla ¿Quién somos?

Al pulsar sobre este icono se obtendrá una nueva pantalla en la que se muestra una descripción ideal de la que sería la clínica pediátrica:

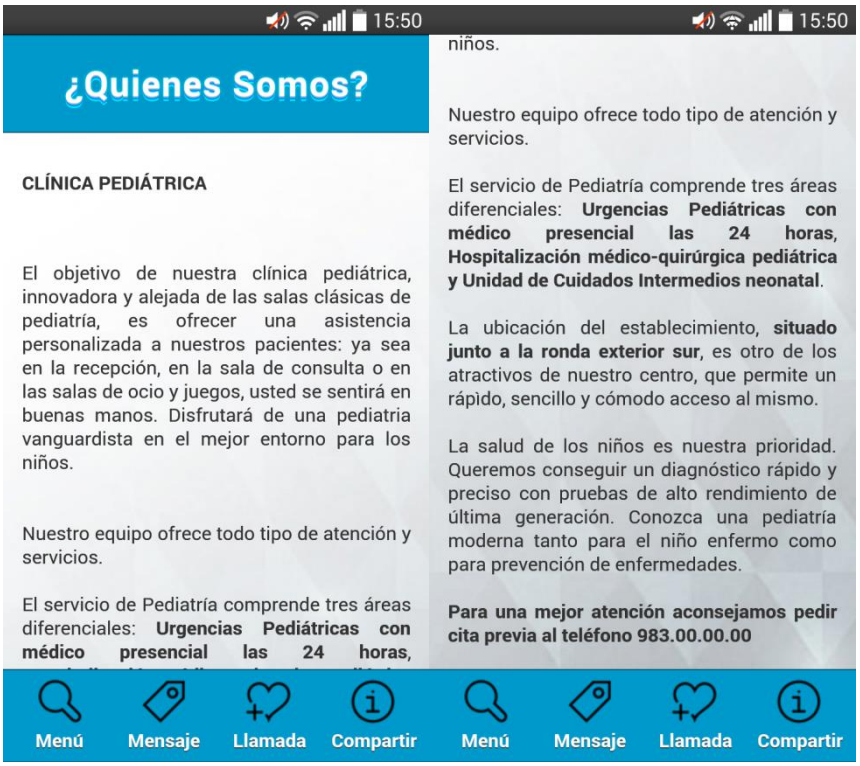


Ilustración 41. Pantalla ¿Quién somos?

5.4 Pantalla Pide tu cita:

En caso de que el usuario desee hacer uso de esta funcionalidad, el cliente debe haber sido dado de alta previamente por su pediatra en el sistema. Una vez contacte con el pediatra, y éste le registre en la base de datos de usuarios, le será entregado al usuario un nombre de usuario y una contraseña con la cual podrá acceder a dicha funcionalidad.

Esta funcionalidad sirve para pedir una cita directamente a su pediatra.

Por lo tanto, la primera pantalla a la que se accederá si pulsamos sobre Pide tu cita, será la pantalla de inicio de sesión:

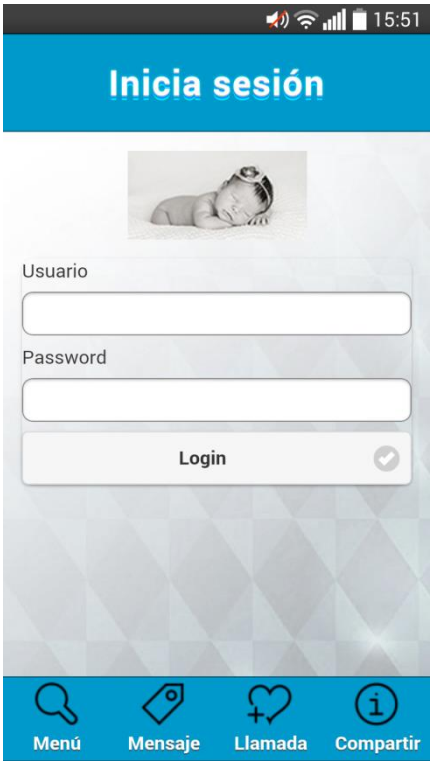


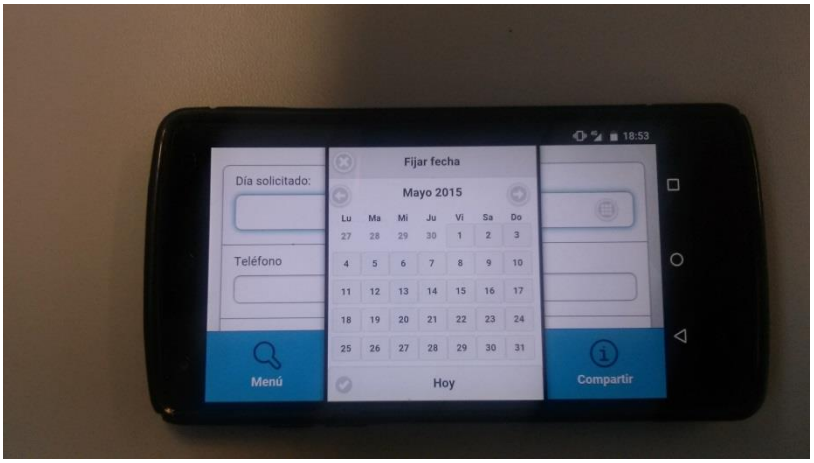
Ilustración 42. Pantalla Inicio sesión

Una vez el usuario haya accedido correctamente a la funcionalidad con sus credenciales, la aplicación se conectará a la base de datos mediante *MySql* y a continuación se mostrará un formulario que el usuario completará para solicitar una cita con el pediatra.

El formulario tendrá un aspecto como el siguiente.

Ilustración 43. Pantalla Pide tu cita

Todos los datos requeridos en los tres campos que aparecen en la pantalla, el día solicitado, el teléfono de contacto y la hora de la cita, son obligatorios para poder pedir una cita. En caso de que alguno de ellos no sea completado, la aplicación mostrará un aviso en el que indica al usuario que se deben introducir todos los datos requeridos.



Fotografía 5. Selección fecha cita en el calendario en Nexus 5 horizontal

Fotografía 6. . Selección fecha cita en el calendario en LG 9 Optimus

El día solicitado para la cita se introducirá marcando el día deseado sobre el calendario emergente que aparece al pulsar sobre el icono de la derecha del campo fecha.

Al igual, la hora deseada para la consulta del pediatra se elige de entre una lista de horas predefinidas por la aplicación.

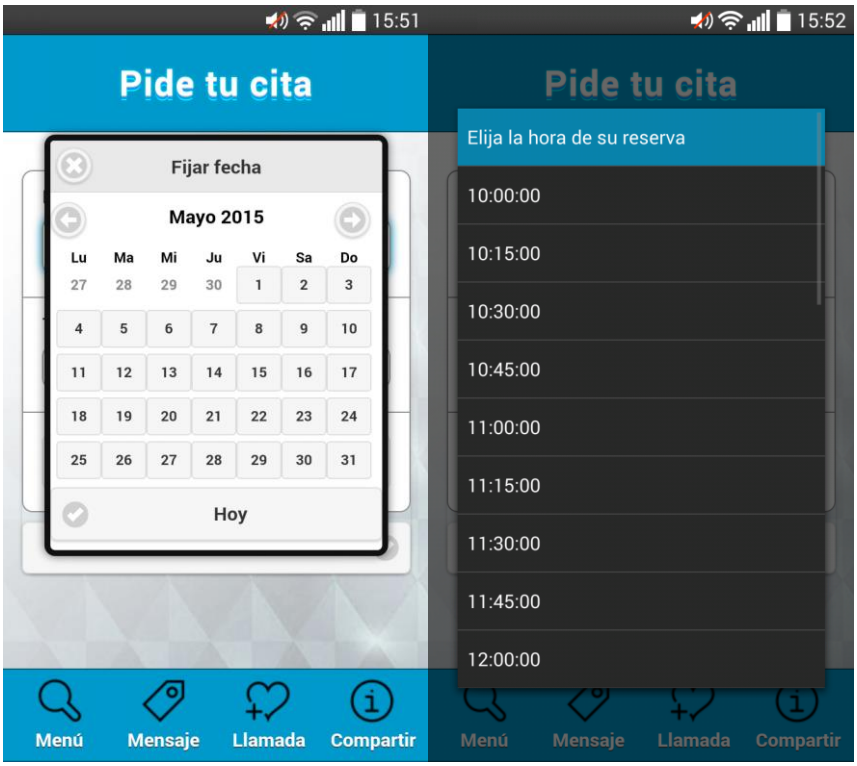


Ilustración 44. Pantalla selección fecha y hora en el formulario para la reserva de cita

Una vez introducidos los datos solicitados, se debe pulsar sobre "Enviar" para confirmar la cita. Si el día y la hora elegida por el usuario está libre en el momento del envío del formulario y la conexión a la base datos en la que se introduce la información de la consulta se ha realizado con éxito, se mostrará un mensaje de confirmación de la cita indicando de nuevo la hora de la consulta.

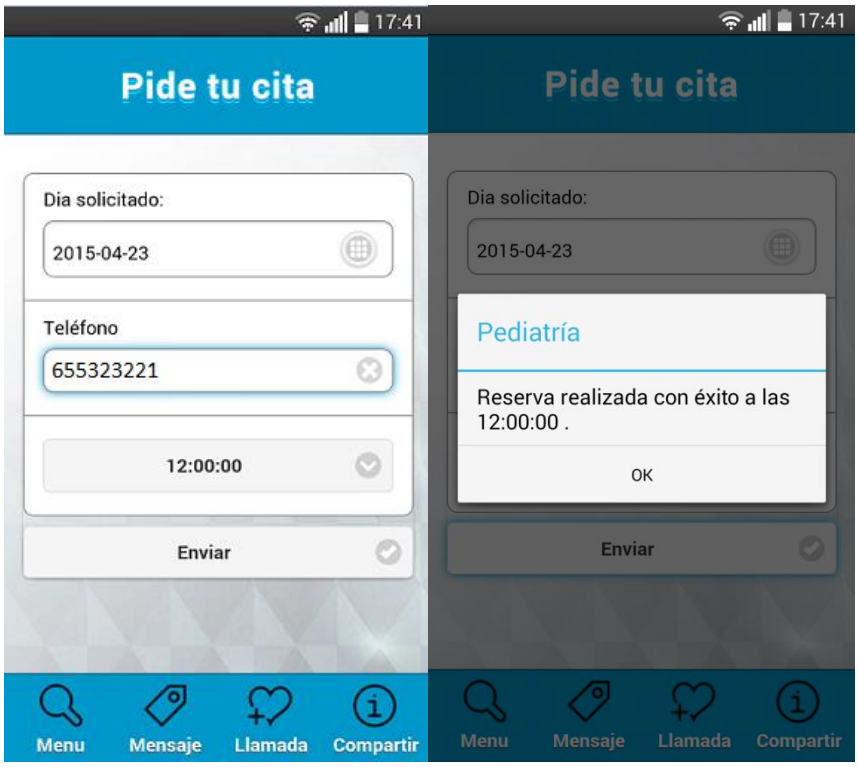


Ilustración 45. Pantalla de confirmación de cita

El pediatra puede consultar las citas solicitadas por los usuarios a través de la aplicación accediendo a la base de datos. En ella encontrará toda la información sobre el paciente que ha solicitado la cita y las horas ocupadas.

Si otro usuario intentase pedir una cita para un día y una hora que ya estuviera ocupada, se le asignará a este usuario la hora de cita más cercana a la que en principio hubiese solicitado. Es decir, si este segundo usuario intentase solicitar una cita para el día 23 de abril a las 12:00h (día y hora solicitadas en el ejemplo anterior por otro usuario), se asignará a este segundo usuario una cita el mismo día 23 de abril pero en la siguiente hora libre más cercana a las 12:00h, siendo en este caso a las 12:15h (hora libre más cercana a la hora solicitada inicialmente por el usuario).

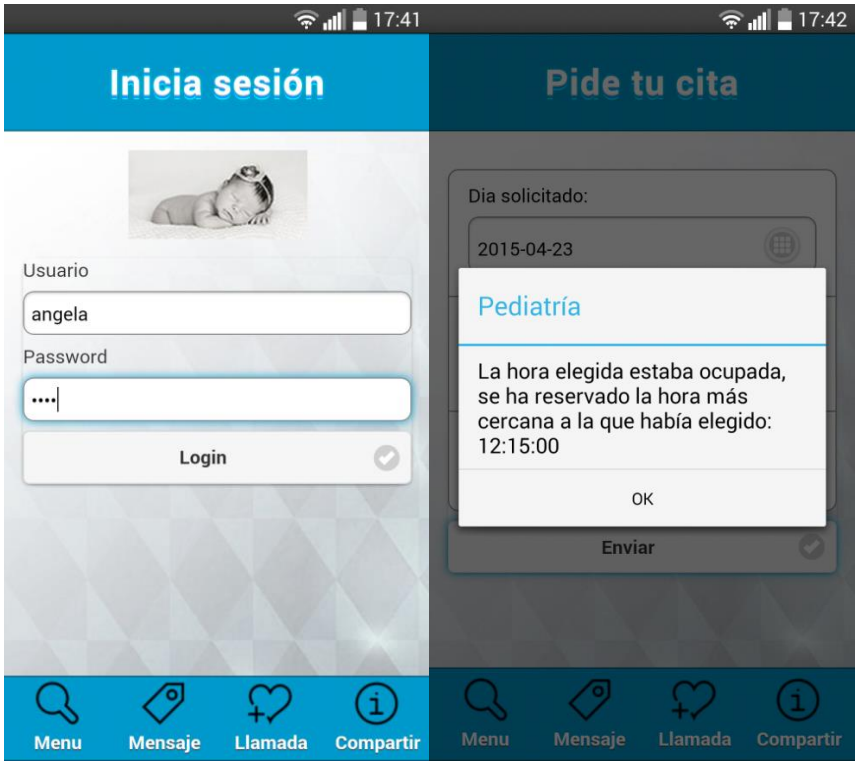


Ilustración 46. Solicitud de cita coincidente con otra ya existente

5.5 Pantalla Lactancia: Preguntas frecuentes FAQ

En esta sección se encuentran las dudas más frecuentes que los padres/madres con hijos en periodo de lactancia suelen tener. Se muestra a continuación un extracto de algunas de las preguntas y respuestas que pueden encontrarse en la aplicación.

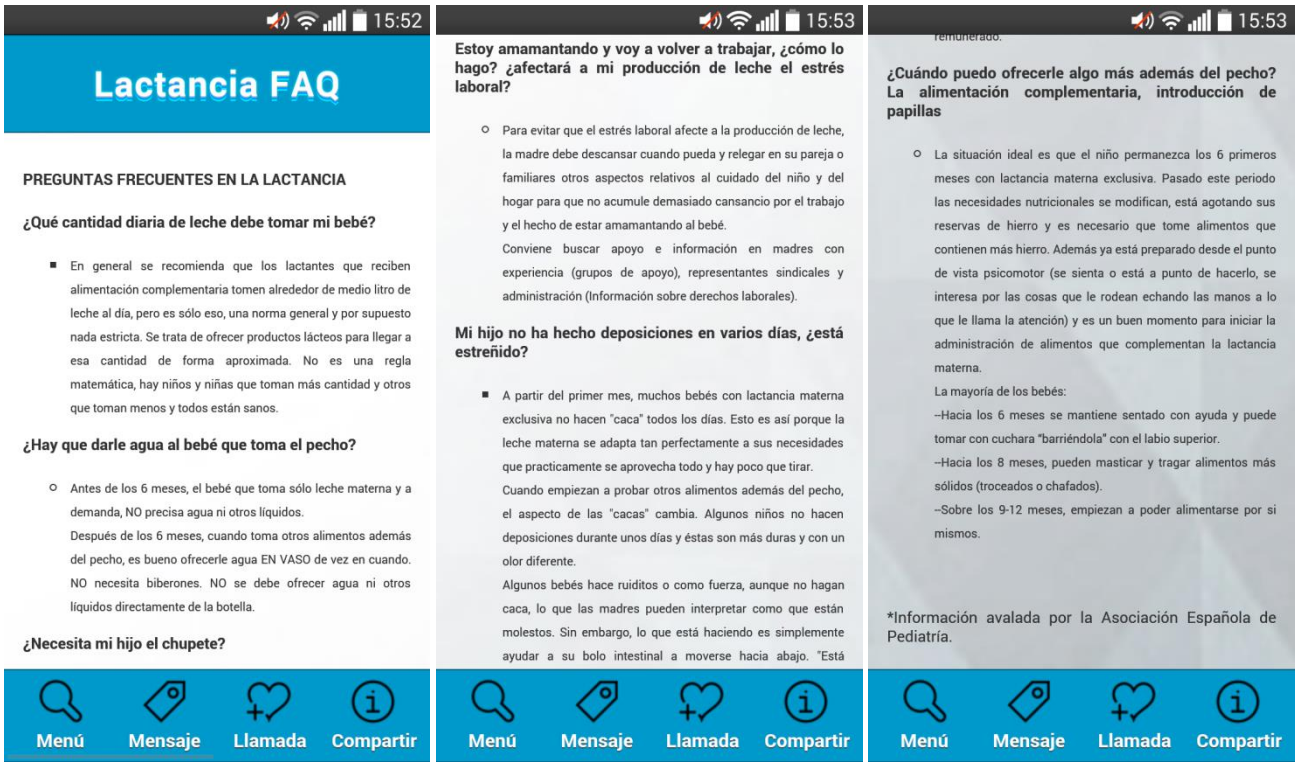
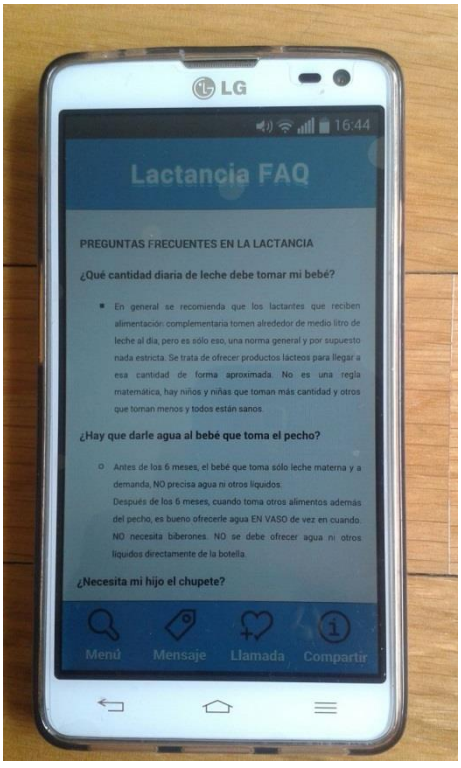


Ilustración 47. Pantalla Lactancia preguntas frecuentes (FAQ)



Fotografía 7. Pantalla Lactancia en LG 9 Optimus

5.6 Pantalla Calendario de vacunas (Castilla y León)

En este apartado se encuentra el calendario de vacunas seguido en la comunidad autónoma de Castilla y León. Pueden apreciarse las diferentes vacunas que deben ser suministradas a los niños desde el nacimiento (0 meses) hasta los 14 años (14a).

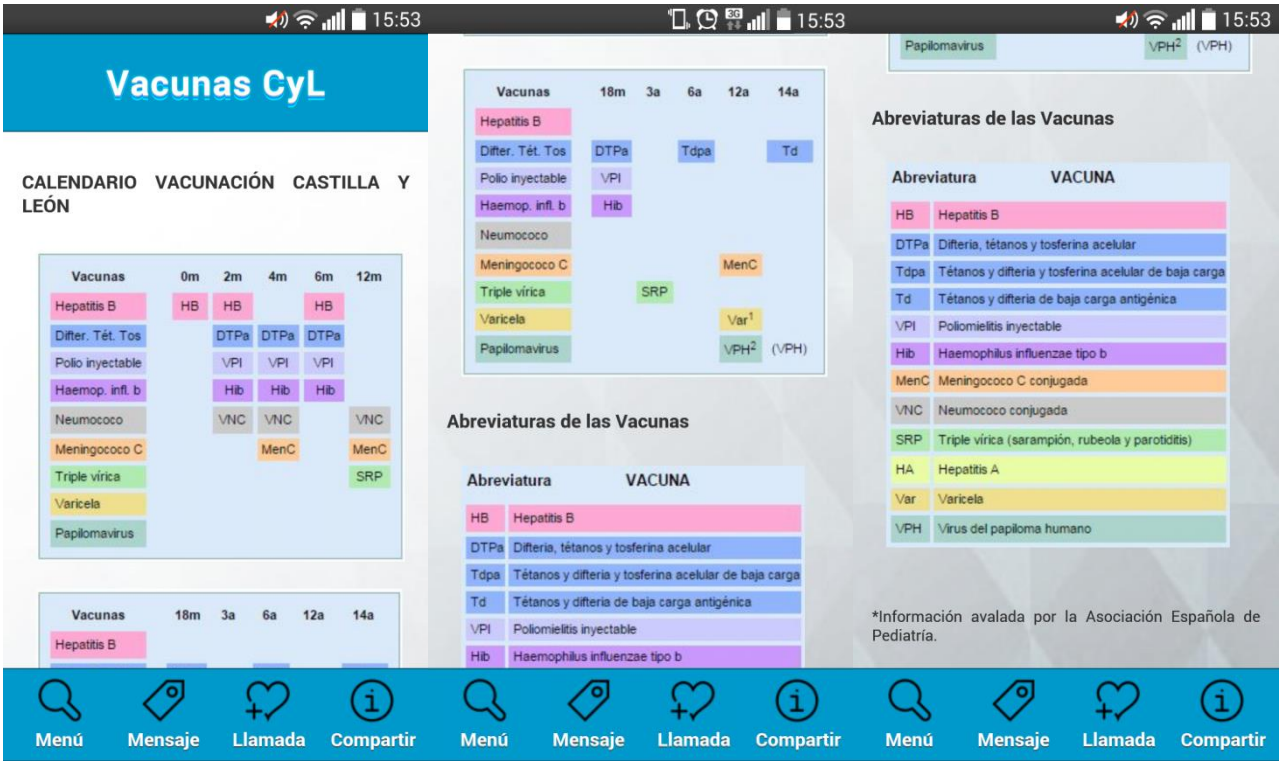
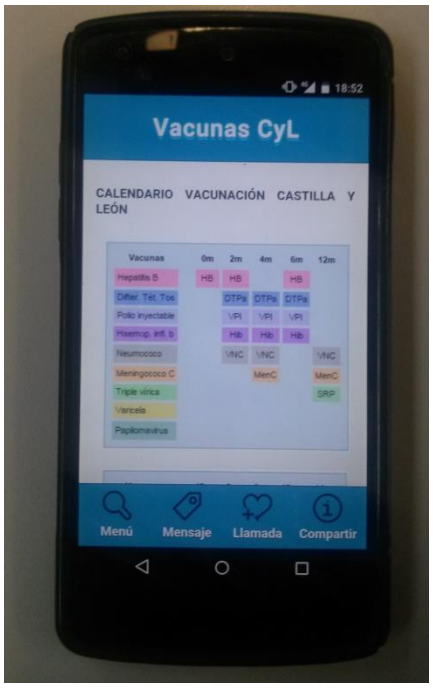


Ilustración 48. Pantalla calendario de vacunas castilla y león



Fotografía 8. Pantalla Calendario de vacunas en Nexus 5

5.7 Pantalla Consulta tu informe (Consulta tus últimas visitas al pediatra)

En caso de que el usuario desee hacer uso de esta funcionalidad, el cliente debe haber sido dado de alta previamente por su pediatra en el sistema. Una vez contacte con el pediatra, y éste le registre en la base de datos de usuarios, le será entregado al usuario un nombre de usuario y una contraseña con la cual podrá acceder a dicha funcionalidad.

Esta funcionalidad sirve para consultar las fechas en las que el usuario ha ido al pediatra, los síntomas que se tenían y el tratamiento que fue aconsejado por su médico.

Por lo tanto, la primera pantalla a la que se accederá si pulsamos sobre Consulta tu informe, será la pantalla de inicio de sesión:

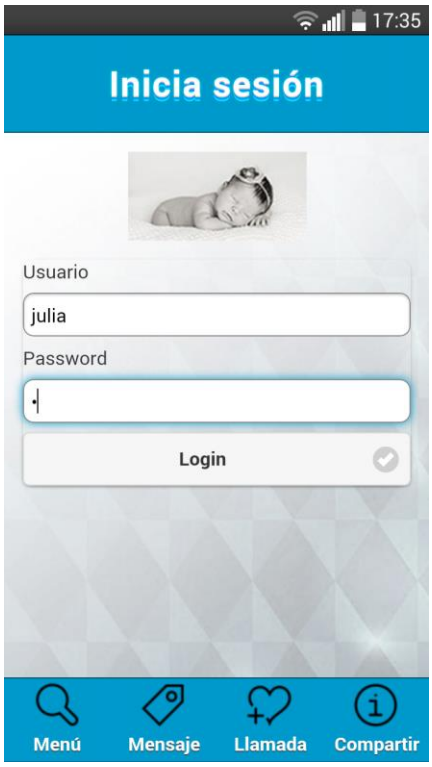


Ilustración 49. Pantalla inicio de sesión

Una vez el usuario haya accedido a la funcionalidad con sus credenciales correctamente, la aplicación se conectará a la base de datos mediante *MySql* y se mostrarán los datos de síntomas y tratamiento de cada consulta que el pediatra haya introducido en la base de datos. Tendrá un aspecto así:

Ilustración 50. Pantalla Consulta tu informe

Las dos consultas de la captura anterior son del usuario: julia (usuario con el que se ha iniciado la sesión), que según la base de datos tiene un idusuario= 2:

idusuario	nombre	telefono_usuario	email	codigo	edad	user
2	julia	655323221	jsanto.@syltec.es	789456	4	julia

Tabla 19. Base de datos de los usuarios. *User* = julia, Idusuario = 2

Por lo que las consultas que se observan en las capturas anteriores (en el apartado CONSULTA TU INFORME en el cual el usuario puede consultar la información de sus visitas al doctor) serán aquellas que el pediatra haya registrado en la base de datos, cuyo idusuario=2 (idusuario correspondiente al usuario que ha iniciado la sesión en este ejemplo). En las captura del móvil, se observan las dos primeras visitas.

id_informe	idusuario	juicio	tratamiento	fecha_informe
3	2	Fiebre	Paracetamol y reposo	2014-12-09
6	2	Catarro	Ibuprofeno	2014-12-30
8	2	Dolor muscular	Reposo	2014-12-23
9	1	Fiebre	Tratamiento contra la fiebre	2014-12-23

Tabla 20. Base de datos : Informes del usuario con Idusuario = 2

5.8 Pantalla Test de Denver

En esta sección los padres y madres podrán tener una pequeña guía sobre algunos de los síntomas que se usan para la realización del Test de Denver a los niños. Hay que saber que estos datos son orientativos, y siempre se debe consultar a un especialista antes cualquier duda.

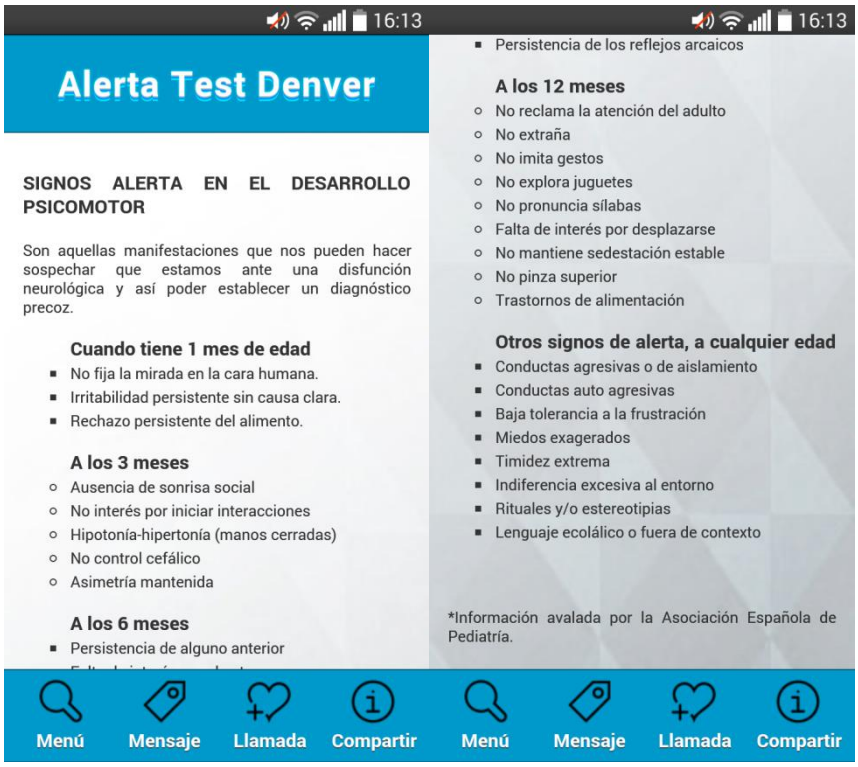
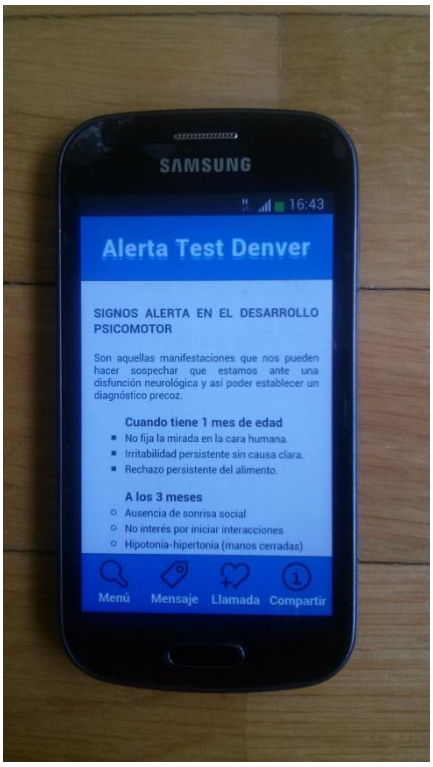


Ilustración 51. Pantalla Test de Denver



Fotografía 9. Pantalla Test de Denver en Samsung GT S7580

5.9 Pantalla Percentiles (Talla y peso)

Al pulsar sobre el icono de “Percentiles”, se abrirá una nueva pantalla como la que se muestra a continuación en la que se podrá elegir que percentil de talla o longitud se desea consultar según sea niño o niña.

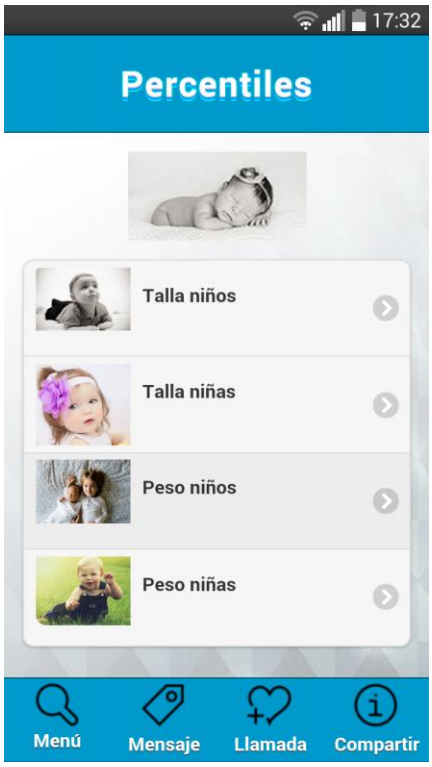
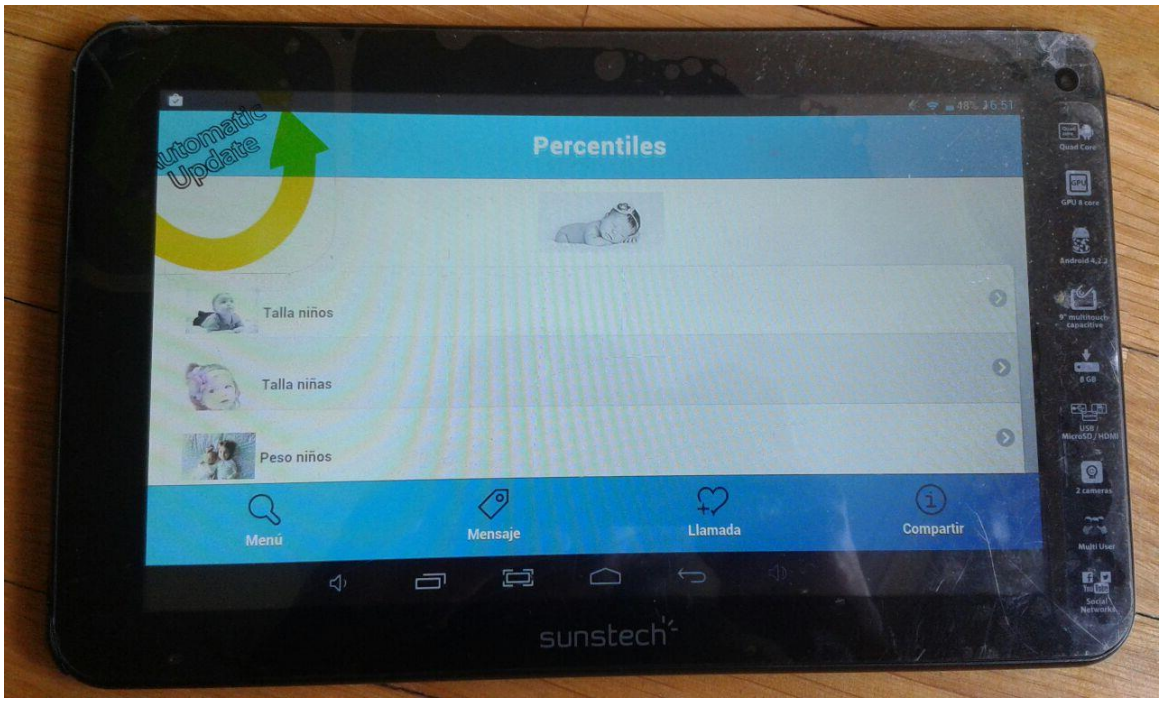


Ilustración 52. Pantalla Menú percentiles

Actualmente en la aplicación se encuentran disponibles 3 percentiles (P3, P50 y el P97) para la talla y peso. También está disponible la columna DS que hace referencia al Desvío Estándar.

Para comprender que significan estos percentiles, se incluye un ejemplo: Un niño o niña que en una edad determinada esté por ejemplo en el percentil 50 de peso, quiere decir que en una muestra de 100 niños, habrá 50 que pesen menos y 50 que pesen más.



Fotografía 10. Pantalla Percentiles en Tablet Sunstech

Se podrán visualizar u ocultar las columnas con los percentiles que no se deseen pulsando sobre el botón “Columns”. La columna EDAD y la columna del percentil P50, serán fijas siempre.

Para volver al menú general de percentiles, mostrado en la ilustración anterior, y consultar otro percentil, se pulsará sobre el botón “Back”. Para volver al menú general de la aplicación, se pulsará el botón “Menú”, disponible en el menú azul inferior.

5.9.1 Pantalla Percentiles: Talla niños

Se visualizan los percentiles P3 y P50 de la talla en cm de los niños con edad comprendida entre el nacimiento y los 18 años.

Ilustración 53. . Pantalla Percentiles longitud (cm) niños

5.9.2 Pantalla Percentiles: Talla niñas

Se visualizan los percentiles P3 y P50 de la talla en cm de las niñas con edad comprendida entre el nacimiento y los 18 años.

Ilustración 54. Pantalla Percentiles longitud (cm) niñas

5.9.3 Pantalla Percentiles: Peso niños

En la siguiente ilustración se observa como si se pulsa sobre el botón "Columns", puede seleccionarse las columnas que se desean ver u ocultar. En este caso se ha seleccionado que se visualicen los tres percentiles disponibles para el estudio longitudinal del peso de los niños.

Ilustración 55. Pantalla Percentiles peso (kg) niños

5.9.4 Pantalla Percentiles: Peso niñas

En la siguiente ilustración se observa como si se pulsa sobre el botón ”Columns”, puede seleccionarse las columnas que se desean ver u ocultar. En este caso se ha seleccionado que se visualicen dos percentiles más, además del percentil P50 que es fijo, para el estudio longitudinal del peso de las niñas.

Ilustración 56. Pantalla Percentiles peso (kg) niñas

5.10 Pantalla Ubicación (¿Dónde estamos?)

Al pulsar sobre el botón ubicación del menú general, se accederá a una pantalla como la siguiente, en la cual, haciendo uso de la geolocalización de Google Maps se observará un mapa con la ubicación de la clínica pediátrica.

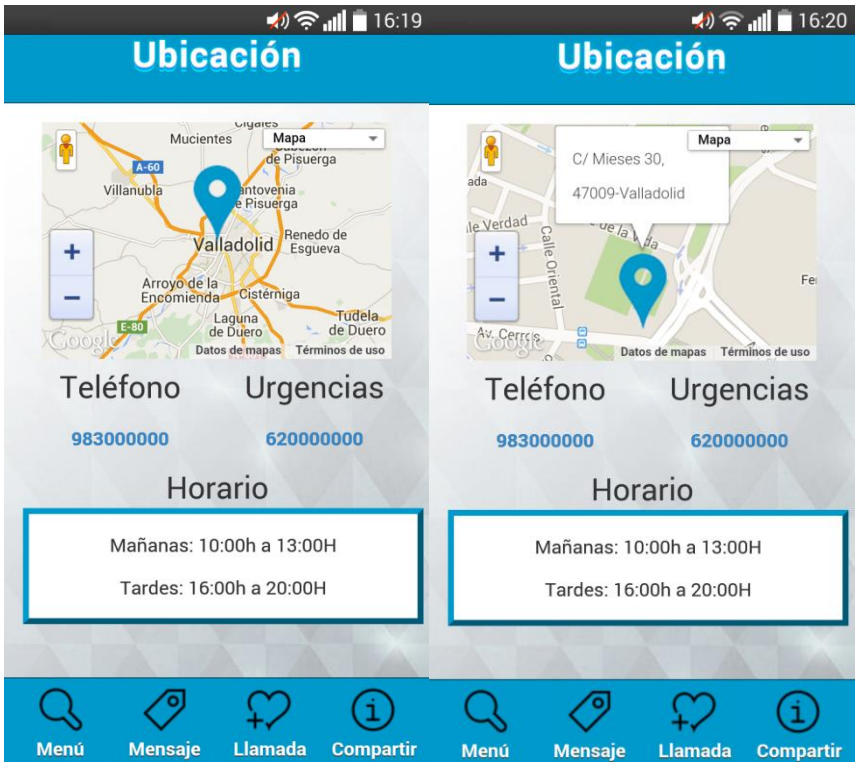


Ilustración 57. Pantalla Ubicación

En este mapa, tocando sobre el icono azul, aparecerá la dirección exacta de la clínica. Además los dos teléfonos indicados, son dos accesos directos. Al tocar sobre alguno de ellos, se abrirá automáticamente y para su comodidad, la pantalla de llamada del móvil del usuario dónde aparecerá tecleado ya el número del pediatra. (Lo mismo ocurre si pulsamos sobre el botón de "Llamada" que aparece en la franja azul inferior de todas las pantallas).

5.11 Pantalla Envío Mensaje (Email)

Al pulsar sobre la tecla "Mensaje" de la franja inferior azul, se abrirá una nueva pantalla mediante la cual cualquier usuario podrá enviar una consulta al pediatra. Sólo se requiere que el usuario de la aplicación rellene los campos solicitados. Idealmente, el pediatra contestaría a la duda o consulta del usuario al correo que el usuario introdujo cuando realizó la consulta.

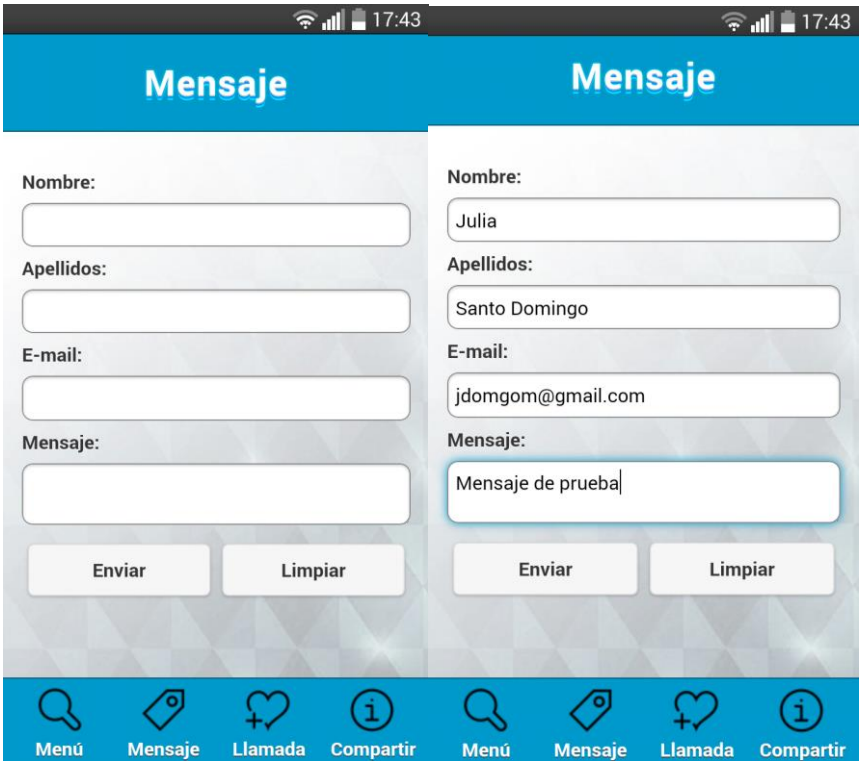
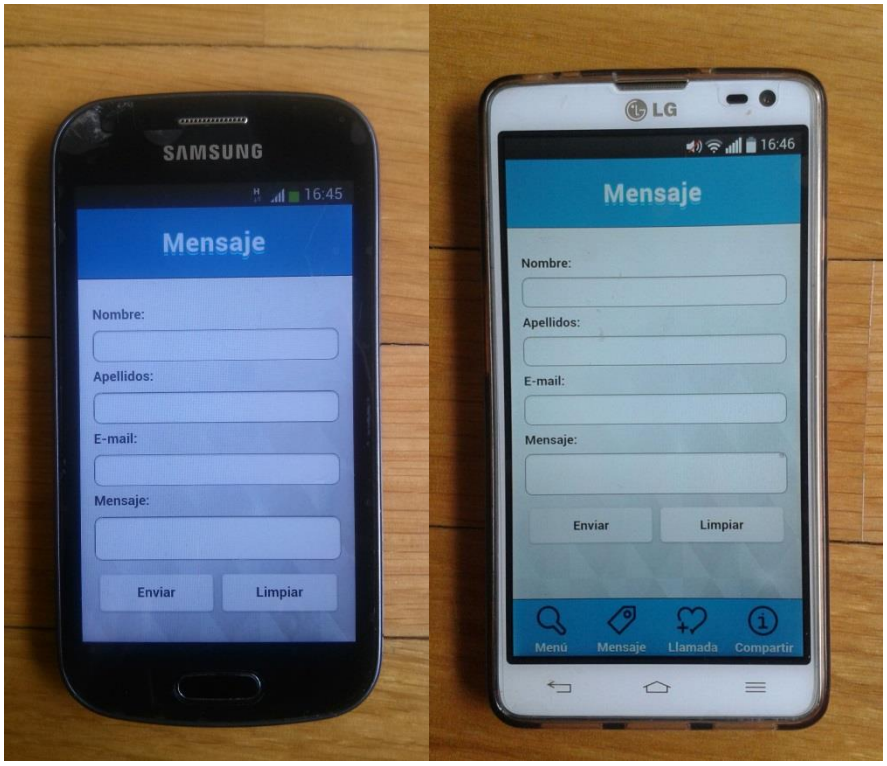


Ilustración 58. Pantalla Envío mensaje electrónico



Fotografía 11. Pantalla Envío Mensaje en Samsung GT S7580 (izquierda) y en LG 9 Optimus (derecha)

Si todo ha ido correctamente, y el mensaje se ha enviado al pediatra correctamente, se mostrará un mensaje de aviso al usuario. Y se comprueba que efectivamente el "pediatra" ha recibido en su buzón de entrada el correo de consulta del usuario.

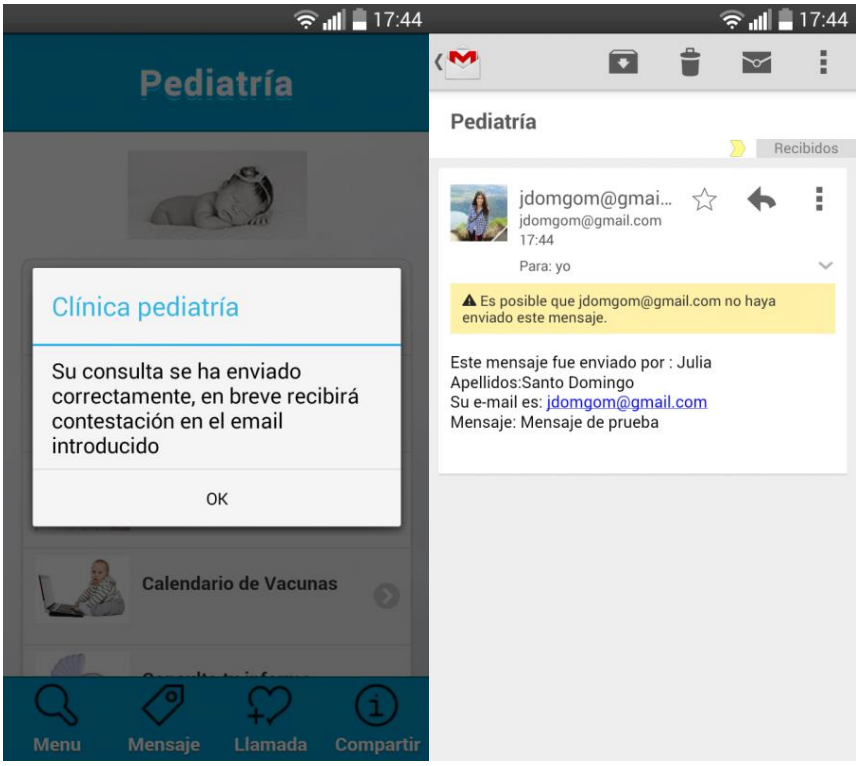


Ilustración 59. Confirmación de envío de mensaje electrónico para el usuario. Recepción correcta del mensaje en el buzón de entrada del pediatra

5.12 Opción Compartir:

Al pulsar sobre la tecla Compartir de la franja inferior azul, aparecerán los medios dónde compartir la aplicación. Esta opción de *Social Sharing* ha sido posible gracias al plugin "*nl.x-services.plugins.socialsharing*" ofrecido por el *framework* usado: PhoneGapBuild.

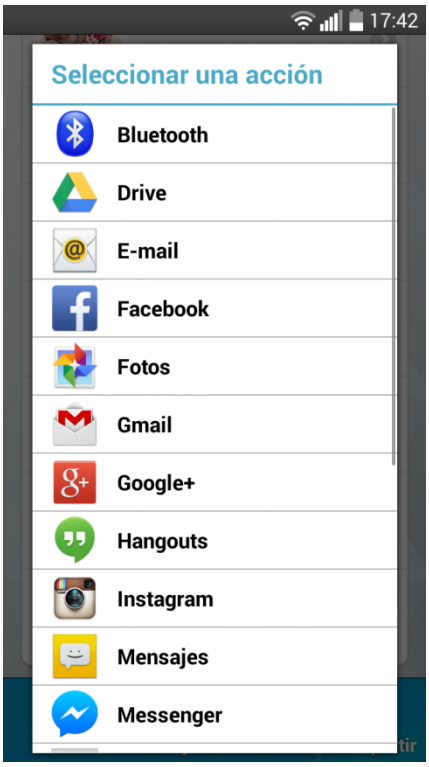


Ilustración 60. Opción Compartir la aplicación

CAPÍTULO 6

6. PRESUPUESTO ECONÓMICO

En este capítulo se estimará un presupuesto económico del proyecto realizado. Siguiendo las diferentes fases seguidas en la realización de este proyecto, se irán incluyendo los diferentes gastos tanto de software como de hardware que han surgido durante este desarrollo.

Antes de comenzar con dicha estimación, se comentará que existen numerosos criterios para establecer un precio a una aplicación. Algunas empresas, priman la mano de obra, otras el tiempo dedicado a su desarrollo, y otras la calidad del software usado.

Por ejemplo, según la funcionalidad de la aplicación, los precios son menores para las aplicaciones con funcionalidades básicas, y van aumentando el precio según se van introduciendo bases de datos con funcionalidades personalizadas, juegos, mejoras o modificaciones en el hardware del dispositivo, hasta llegar a las aplicaciones totalmente dinámicas, como pueden ser *Pages* o *Adobe Ideas*, cuyo su precio alcanza cifras muy altas.

Además existen otros factores que incrementan el precio de las aplicaciones móviles. Incluir en una aplicación básica la funcionalidad de comprar, puede aumentar el precio en más de 3000€. Los servicios web, para llevar el contenido a un punto de acceso remoto para que pueda actualizar su aplicación con archivos XML, y los *game center*, también hacen que su precio se incremente entre 1500 y 2500€ a mayores.

El presupuesto estimado para esta aplicación de PEDIATRIA suponiendo que esta aplicación la hubiese realizado como una trabajadora autónoma, ha sido:

Estimación presupuesto económico		
Coste mano de obra programador junior	13€/h * 160h	2.080 €
Cuota autónomo	264.44€/mes = 8.59€/día → En 20 días: 170€	170 €
Licencia DreamWeaber	24,19 €/mes	15 €
Licencia PhotoShop	12€/mes	8 €
Costes publicitarios	En un periódico y en una revista médica	100 €
Costes oficina y gastos corrientes	Alquiler + Luz + Conexión Internet + Teléfono + Gas (20 días)	196 €
Software Libre	phpMyAdmin como gestor de base de datos relacionales	0 €
	NotePad++	0 €
	PHP5 para conexión MySql	0 €
	PhoneGap y PhoneGapBuild	0 €
Tecnologías web estándar	HTML, CSS, JavaScript y jQueryMobile.	0 €
Coste ordenador portátil con Windows 7	Coste total: 600€. Amortización fiscal 25% al año (según Hacienda): 150€. Al mes: 12,5€.	13 €
TOTAL		**2.582 €**

Tabla 21. Estimación presupuesto económico

El ordenador portátil ha sido necesario durante todo el proceso de desarrollo de aplicación. Primero se usó para la búsqueda de documentación inicial y para el aprendizaje de *jQueryMobile*, *framework* que se ha usado para el desarrollo de la aplicación, y que ha sido aprendido gracias a documentación y a ejemplos gráficos encontrados en Internet. Una vez finalizada la fase previa de documentación y aprendizaje, se comenzó a plantear lo que sería la aplicación en sí. Para ello se tuvieron reuniones con dos pediatras que han vitales para la realización de una aplicación funcional de fácil manejo y que efectivamente pudiera servir de apoyo para los padres y madres de los niños que acuden a un pediatra.

A continuación se empezó el desarrollo de la aplicación PEDIATRÍA. A la hora de realizar el presupuesto económico, se han tenido en cuenta aproximadamente unas 160 horas, en las que no se tiene en cuenta la fase inicial de aprendizaje, pero si se han tenido cuenta las horas en las que se han llevado a cabo reuniones con ambos pediatras para la solución de distintas dudas que han surgido durante el desarrollo y programación de esta aplicación.

Es por esto que teniendo en cuenta el salario de un programador *junior,* aproximadamente de 13€/hora, si ha necesitado unas 160 horas para la realización de esta aplicación, se estima un gasto de mano de obra aproximado de 2080€. Aproximadamente 160 € son las horas equivalentes a 20 días de trabajo con una jornada laboral de 8 horas al día. Es por este motivo por el que todos los cálculos de los gastos se estiman aproximadamente para 20 días.

Aclaraciones a las diferentes partidas de este presupuesto:

- Salario del programador, que es la única partida de ingresos y que representa el mayor coste del presupuesto.
- Cuota mensual de la seguridad social en régimen de autónomos (266.44€/mes). En este presupuesto este gasto se ha prorrateado para 20 días.
- Licencias de dos programas usados para la realización de la aplicación:
    - Adobe Dreamweaver: Se trata de un software de diseño web, desarrollado para la creación y la edición de sitios web HTML y aplicaciones para dispositivos móviles. Es el programa de desarrollo y diseño web más utilizado, debido a las funcionalidades que ofrece y a la posibilidad de integrar otras herramientas como *Adobe Flash*. Además permite la conexión a un servidor o a una base de datos.
    - Adobe Photoshop: Consiste en un programa para la creación, edición y retoque de imágenes, desarrollado por la compañía *Adobe Systems*. En un principio fue creado para la plataforma Apple pero posteriormente se desarrolló para Windows. Sin duda se trata del programa de edición de fotografía más popular, ello se debe a las numerosas posibilidades de retoque y modificación de fotografías que ofrece.
- Costes publicitarios: Para dar a conocer la aplicación se hizo publicidad sobre ella en un periódico local y en una revista médica. Los costes prorrateados de la inserción de estos anuncios son los que se reflejan en el presupuesto anterior.
- Gastos generales correspondientes al lugar de trabajo. Estos son, el coste del alquiler, la luz, el gas, la conexión a Internet y el teléfono. La estimación de estos gastos fijos se ha prorrateado igualmente para 20 días.
- Coste del ordenador portátil. El ordenador tiene una depreciación y según el Ministerio de Hacienda, se permite que anualmente se incluya como gasto de depreciación en la declaración de impuestos hasta un 25% del valor total.
  Esto implica que en 4 años el ordenador queda desgravado, por lo que al año se puede poner como gasto de depreciación del ordenador un 25% de su valor (150€). De este importe, se incluirá en este presupuesto la parte proporcional correspondiente a un mes (12.50€), y no a 20 días como en los casos anteriores, porque suponemos que cuando adquiramos un nuevo equipo éste tendrá un coste superior al actual (aplicando el criterio contable de coste de renovación).

Finalmente, se ha estimado que el presupuesto para el desarrollo de la aplicación PEDIATRÍA elaborada en este TFG, es de aproximadamente 2600€.

De este importe, se estima un beneficio aproximado para el programador de 1900€.

CAPÍTULO 7

7. CONCLUSIONES Y LÍNEAS FUTURAS

7.1 Conclusiones

En este capítulo, me gustaría reflexionar sobre lo que ha supuesto para mí la realización de este Trabajo de Fin de Grado tanto a nivel profesional y académico como a nivel personal.

Este Trabajo Fin de Grado lo comencé a la vez que realizaba las prácticas curriculares en empresa de 300 horas de duración, en la empresa SYLTEC. Con la realización de este TFG, he aprendido mucho de manera autodidacta sobre nuevas tecnologías, tanto web cómo móvil, ampliando así las ya estudiadas en los últimos años de la carrera. La realización de estas prácticas y también de este Trabajo Fin de Grado me ha enseñado a superar dificultades que iban surgiendo a lo largo del desarrollo del mismo. Desde aquí, por tanto, quiero agradecer a Rocío, tutora de las prácticas en SYLTEC, todo lo que me ha enseñado en este tiempo. También agradecer a mi tutora Míriam las facilidades prestadas, apoyándome en el desarrollo de este TFG, lo que ha supuesto una gran ayuda.

A continuación se describirá cómo han sido cumplidos satisfactoriamente los objetivos que se planificaron en el capítulo uno.

Primeramente se contextualizó la necesidad de desarrollar una aplicación destinada al apoyo de una Clínica Pediátrica para dispositivos móviles, describiendo la panorámica actual de las aplicaciones móviles en el campo de la salud móvil.

Después se estudiaron distintas tecnologías WEB que fueron utilizadas durante la realización de mis prácticas en empresa y algunas fueron usadas también para el desarrollo de la aplicación PEDIATRÍA, gracias a esto se ha perfeccionado mucho de manera práctica el uso de estas tecnologías.

Por otro lado, para el desarrollo de la aplicación móvil como herramienta de apoyo para los usuarios de una Clínica Pediátrica, se analizaron las distintas tecnologías móviles existentes,

haciendo una comparativa entre ambas y justificando por qué se eligió realizar la aplicación PEDIATRÍA con una tecnología híbrida mediante el uso del framework de jQuery Mobile.

Una vez puestos en contexto se desarrolló el análisis funcional de la aplicación PEDIATRÍA, objetivo final de esta Trabajo de Fin de Grado. Gracias a este análisis, podemos comprobar que se cumplen los objetivos de la aplicación sobre escalabilidad, facilidad de uso y dinamismo de su contenido.

Posteriormente, se ha elaborado un manual de uso, con el fin de que pueda servir de guía a los posibles usuarios de la aplicación, donde se recogen las posibles situaciones o escenarios que puedan presentarse, habiéndose verificado el correcto funcionamiento de las aplicaciones.

Por lo tanto, los objetivos establecidos en el primer capítulo se han cumplido de forma satisfactoria. He logrado por una parte, aprender a programar mediante el uso de un framework, ampliamente utilizado en el desarrollo de aplicaciones móviles como es jQuery Mobile, y por otra parte, implementar una aplicación móvil m-health en el campo de la pediatría que espero, algún día pueda ayudar a los usuarios de una Clínica Pediátrica.

Finalmente, me gustaría decir que tras estos años en la Universidad de Valladolid, hoy me siento muy feliz y orgullosa de poder decir que con este Trabajo Fin de Grado pongo punto y seguido a una etapa de mi vida que he aprovechado al máximo y en la que he adquirido infinidad de conocimientos.

Nadie me dijo que comenzar aquí fuera a ser fácil, pero paso a paso, viendo cómo iba logrando poco a poco mis objetivos, aprendiendo de cada experiencia que he vivido en estos años, he ido sacando la fuerza necesaria que me hace estar aquí hoy, con la misma ilusión con la que entré el primer día en la Universidad y con la que ahora me pregunto qué me deparará el futuro.

7.2 Líneas futuras

A lo largo de todo el proceso de desarrollo de este Trabajo Fin de Grado han ido surgiendo nuevas ideas y mejoras que pueden ser incluidas, en un futuro, en una versión de la aplicación para mejorarla y aumentar sus funcionalidades.

De la misma manera que surgían nuevas ideas, también aparecieron inconvenientes y problemas, que hubo que aprender a resolver, para sacar adelante esta aplicación.

En un principio se planteó un modelo de aplicación caracterizado por la reserva de citas y la consulta del historial de los pacientes, pero según se fue trabajando en el desarrollo de la aplicación fue necesario incluir las funcionalidades de percentiles, test de Denver y preguntas frecuentes sobre lactancia, puesto que consultando con la pediatra, María Pilar García López, me explicó que eran tres de los aspectos por las que los padres y madres de los niños más acuden a su consulta.

En cuanto a nuevas funcionalidades que se pueden implementar para la siguiente versión de la aplicación de manera práctica, se proponen las siguientes:

- Implementar un chat interno dentro de la aplicación en el que el paciente pudiera consultar al pediatra en tiempo real. Su funcionamiento sería parecido al de la famosa aplicación WhatsApp, pero sin la necesidad de números de teléfono. Los mensajes enviados a través de la aplicación podrían almacenarse en una tabla creada para este fin en la base de datos.

  Esta característica supondría una gran diferencia con respecto a otras aplicaciones que pudieran parecerse y haría que fuese mucho más útil y provechosa.

- Realizar una nueva vista de los historiales de los pacientes para el pediatra. Actualmente, al usuario sólo se le muestra por pantalla cierta información sobre sus informes de las visitas al pediatra. Sería una buena idea, por lo tanto, implementar otra vista de historiales completos para el pediatra, facilitando toda la información al pediatra que se traduciría en un ahorro de tiempo.

- Diabetes. Constituye una de las enfermedades crónicas más frecuentes en la infancia con una prevalencia de 1,7 individuos afectados por 1.000 habitantes menores de 20 años de edad. Diseñar un nuevo apartado orientado a describir información sobre la enfermedad y sus tratamientos.

- La enfermedad celíaca (EC) consiste en una intolerancia a las proteínas del gluten. Hoy en día, numerosos niños padecen esta enfermedad, por lo que podría ser recomendable añadir un apartado más en el que ofrecer información valiosa sobre alimentos seguros para consumir y sobre el día a día en casa, en la escuela, viajes, etc.

Si la aplicación creciera y se incluyeran estas funcionalidades u otras que se creyeran convenientes, también se podría crear un sistema gestor del contenido de la aplicación con distintos roles de usuario. De esta manera, pediatras o usuarios podrían acceder al contenido relevante para cada uno de ellos de forma más cómoda.

Existe todo un abanico de funcionalidades dentro del campo de la medicina que podrían ser incluidas en versiones posteriores de la aplicación PEDIATRÍA. Esto haría que la aplicación creciera y pudiera convertirse en una aplicación de referencia tanto para los pacientes como para los pediatras que podrían encontrar en ésta una solución más rápida a para la inserción de datos en los historiales de los pacientes.

Creo fielmente que con estas nuevas mejoras, la aplicación se abriría camino en un mercado tan grande como el las aplicaciones m-health; en el que con un poco de tiempo y publicidad, la aplicación PEDIATRÍA llegaría a un público mucho muy amplio, que deseará aprovecharse de las ventajas y facilidades que esta aplicación les ofrece.

BIBLIOGRAFÍA

Actualidad iPhone. (30 de Marzo de 2015). *Actualidad iPhone: El 78% de los dispositivos ya usan iOS 8*. Recuperado el 14 de Abril de 2015, de http://www.actualidadiphone.com/el-78-de-los-dispositivos-ya-usan-ios-8/

Adobe Systems Inc. (2015). *PhoneGap: Supported Features*. Recuperado el 22 de Marzo de 2015, de http://phonegap.com/about/feature/

Álvarez, M. A. (2012). *Qué es HTML 5*. Recuperado el 13 de Febrero de 2015, de http://www.desarrolloweb.com/articulos/que-es-html5.html

Android Developers. (2014). *Introducción a Android*. Recuperado el 15 de Marzo de 2015, de http://developer.android.com/intl/es/guide/index.html

Antón Rodríguez, M., & Pérez Juárez, M. Á. (2014). *'CSS3' documentación de la asignatura de Laboratorio de Desarrollo de Sistemas Telemáticos.* Valladolid: Universidad de Valladolid.

Antón Rodríguez, M., & Pérez Juárez, M. Á. (2014). *'JavaScript' documentación de la asignatura de Laboratorio de Desarrollo de Sistemas Telemáticos.* Valladolid: Universidad de Valladolid.

Antón Rodríguez, M., & Pérez Juárez, M. Á. (2014). *'PHP', documentación de la asignatura de Laboratorio de Desarrollo de Sistemas Telemáticos.* Valladolid: Universidad de Valladolid.

Antón Rodriguez, M., de la Torre Díez, I., Gutiérrez Díez, P., & Díaz Pernas, F. (2010). *"Sistema de acceso inalámbrico para la gestión de historiales clínicos electrónicos de pacientes con discapacidad cognitiva," en Conferencia IADIS Ibero Americana WWW/INTERNET, CIAWI* .

Cabo Diez, L. (18 de Febrero de 2015). *mHealth: desarrollo, crecimiento y futuro*. Recuperado el 3 de Febrero de 2015, de http://blogthinkbig.com/mhealth-desarrollo-crecimiento-y-futuro/

Charte Ojeda, F. (2004). *PHP 5.* Anaya Multimedia.

Cobo, Á., Gómez, P., Pérez, D., & Rocha, R. (2011). *PHP y MySQL: Tecnología para el desarrollo de aplicaciones web (Spanish Edition).* Díaz de Santos.

Com Salud. (2014). *La eSalud, el portal de la eSalud en español*. Recuperado el 3 de 4 de 2015, de http://laesalud.com/que-es-esalud/

CreativeCommons. (2014). *Lungo.JS un framework para desarrollo de aplicaciones móviles en HTML5*. Recuperado el 25 de Marzo de 2015, de http://html5facil.com/tips/lungo-js-un-framework-para-desarrollo-de-aplicaciones-moviles-en-html5/

De la Serna, J. L., & Mugarza, F. (2014). *The App Intelligence. Las 50 mejores apps de salud en español.* Observatorio Zeltia. Obtenido de http://www.theappdate.es/blog/wp-content/uploads/2014/03/Informe-TAD-50-Mejores-Apps-de-Salud.pdf

Eguiluz, J. (2015). *Introducción a AJAX*. Recuperado el 3 de Abril de 2015, de http://librosweb.es/libro/ajax/

Europa press. (19 de Marzo de 2014). *Impulsamos*. Recuperado el 7 de Abril de 2015, de http://www.europapress.es/impulsamos/vida-saludable/noticia-aplicaciones-moviles-salud-generaran-2015-volumen-negocio-4000-millones-euros-espana-20140319162446.html

Firtman, M. (2008). *Ajax: Web 2.0 para profesionales.* S.A. Marcombo.

Franklin, J. (2013). *Beginning jQuery (Beginning Apress).* Apress.

García Echegaray, B. (18 de Octubre de 2014). *Introducción a Phonegap: webview, plugins y herramientas*. Recuperado el 25 de Marzo de 2015, de http://blog.garciaechegaray.com/2014/10/18/phonegap-workshop.html

Gauchat, J. D. (2013). *El Gran libro de HTML5, CSS3 y Javascript .* Barcelona: Marcombo.

Glera Aransay, C. (2013). *Desarrollo de una guía para dispositivos móviles de establecimientos para celiacos en Logroño.* Logroño: Proyecto fin de carrera.

Google Play. (2015). *Developer Console Help*. Recuperado el 10 de Febrero de 2015, de https://support.google.com/googleplay/android-developer/table/3539140?hl=en&rd=1

Gruman, G. (18 de Marzo de 2015). *Mobile security: iOS vs. Android vs. BlackBerry vs. Windows Phone*. Recuperado el 22 de Marzo de 2015, de http://www.infoworld.com/article/2898102/mobile-security/mobile-security-ios-vs-android-vs-blackberry-vs-windows-phone.html

Guapu Technologies, S.L. (2013). *Aprendiz de Objective-C*. Recuperado el 15 de Marzo de 2015, de http://www.cursoios.es/cursos-de-programacion/aprendiz-de-objective-c/

Guilcher, P. (2015). *Restlet*. Recuperado el 10 de Abril de 2015, de http://restlet.org/.

Gutierrez Gallardo, J. D. (2009). *Desarrollo web con PHP 5 y MYSQL.* Madrid: Anaya.

Hernandez Vargas, H. M. (1 de Junio de 2014). *El BlackBerry OS es un sistema operativo móvil desarrollado*. Recuperado el 20 de Marzo de 2015, de

https://prezi.com/klui__8_3bvz/el-blackberry-os-es-un-sistema-operativo-movil-desarrollado/

Jiménez Díaz, J. (04 de 02 de 2014). *La eSalud que queremos. Ensayos sobre la eSalud: una definición en 1000 palabras*. Recuperado el 3 de 4 de 2015, de http://laesaludquequeremos.blogspot.com.es/2014/02/ensayos-sobre-la-esalud-una-definicion.html

jQuery Mobile. (2015). *ThemeRoller for jQuery Mobile*. Recuperado el 2 de Enero de 2015, de http://themeroller.jquerymobile.com/

kSoap2-Android. (2014). *Efficient SOAP library for the Android platform*. Recuperado el 10 de Abril de 2015, de https://code.google.com/p/ksoap2-android/

Lance Talent. (20 de Febrero de 2015). Recuperado el 1 de Marzo de 2015, de Los 3 tipos de aplicaciones móviles: ventajas e inconvenientes: http://www.lancetalent.com/blog/tipos-de-aplicaciones-moviles-ventajas-inconvenientes/

Macprogramadores. (2013). Recuperado el 14 de Marzo de 2015, de Comunidad de programadores Mac OS X y iOS: http://www.macprogramadores.org/?q=content/ios

Mateos Diaz, C. M. (2005). *SWAM: Un lenguaje para la programaci´on.* Tandil: Universidad del centro de la provincia de Buenos Aires.

Meléndez, C. (Octubre de 2012). *Maestros del WEB: Aplicaciones móviles con Sencha Touch*. Recuperado el 27 de Marzo de 2015, de http://www.maestrosdelweb.com/aplicaciones-moviles-sencha-touch/

Microsoft. (20 de Marzo de 2015). *Windows: Centro de desarrollo*. Obtenido de http://dev.windows.com/es-es

Navajas Ojeda, A. (2012). *Guía completa de CSS3.* Autoedición.

Open Handset Alliance. (2012). *Open Handset Alliance: Members*. Recuperado el 10 de Marzo de 2015, de http://www.openhandsetalliance.com/oha_members.html

Oracle and/or its affiliates. (2014). *Java™ Platform, Standard Edition 7*. Recuperado el 15 de Marzo de 2015, de http://docs.oracle.com/javase/7/docs/api/

Oros Cabello, J. C. (2002). *Diseño de páginas web interactivas con JavaScript y CSS. (3ª edición).* Madrid: RA-MA.

Ortiz de Zárate, I. (Noviembre de 2011). *Una docena de frameworks para el desarrollo de aplicaciones en dispositivos móviles*. Recuperado el 27 de Marzo de 2015, de

http://unadocenade.com/una-docena-de-frameworks-para-el-desarrollo-de-aplicaciones-en-dispositivos-moviles/

Ortiz, A. (10 de Enero de 2013). *Juego de tronos entre fabricantes y plataformas móviles.* Obtenido de http://www.xataka.com/moviles/juego-de-tronos-entre-fabricantes-y-plataformas-moviles-tecnologia-2013

Pastor, J. (12 de Marzo de 2014). *Desarrollo de aplicaciones móviles (I): así está el mercado*. Obtenido de http://www.xatakamovil.com/mercado/desarrollo-de-aplicaciones-moviles-i-asi-esta-el-mercado

Pérez López, C. (2007). *MySQL para Windows y Linux.* Paracuellos del Jarama (Madrid): Ra-MA.

Petrazzini, G. O. (2012). *Sistemas Operativos en Dispositivos.* Recuperado el 15 de Marzo de 2015, de http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/Sistemas_Operativos_en_Dispositivos_Moviles.pdf

Pimienta, P. (5 de Mayo de 2014). *Tipos de aplicaciones móviles y sus características.* Recuperado el 27 de Marzo de 2015, de http://deideaaapp.org/tipos-de-aplicaciones-moviles-y-sus-caracteristicas/

Prieto Cortera, J. (25 de Febero de 2015). *¿Cuántos smartphones usan iOS o Android?* Obtenido de http://es.blastingnews.com/tecnologia/2015/02/cuantos-smartphones-usan-ios-o-android-00283881.html

PWC. (Junio de 2013). *Socio-economic impact of mHealth An assessment report for the European Union.* Recuperado el 7 de Abril de 2015, de http://www.gsma.com/connectedliving/wp-content/uploads/2013/06/Socio-economic_impact-of-mHealth_EU_14062013V2.pdf

Ramakrishnan, R., & Gehrke, J. (2003). *Database Management System. (Tercera edición).* McGRAW-HILL.

Rivero Cornelio, E., Martínez Fuentes, L., & Alonso Martínez, I. (2005). *Bases de datos relacionales: fundamentos y diseño lógico.* Madrid: Universidad Pontificia de Comillas.

Rodríguez Lorenzana, L. (2012). *Desarrollo y puesta a punto de una aplicación en Android para la rehabilitación de pacientes con déficits y/o deterioros cognitivos.* Proyecto Fin de Carrera.

Silberschatz, A., & F. Korth, H. (2002). *Fundamentos de bases de bases de datos (Cuarta edición).* McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S. A. U.

Sociedad española de informática de la salud. (2015). *Informática y salud. Especial mHealth: salud móvil.* Madrid. Obtenido de http://www.seis.es/documentos/revistas/revistacompleta/Revista_SEIS_IS_110.pdf

Sosa Sosa, V. J. (s.f.). *MIDDLEWARE: Arquitectura.* Cinvestav-Tamaulipas.

The jQuery Foundation. (2015). *jQuery Mobile: A Touch-Optimized Web Framework*. Recuperado el 25 de Marzo de 2015, de https://jquerymobile.com/

The jQuery Foundation. (2015). *jQuery: write less, do more*. Recuperado el 10 de Abril de 2015, de http://jquery.com/

Universidad Carlos III. (Marzo de 2014). Software de comunicaciones. Programación en dispositivos móviles portables. Madrid, España.

W3C. (2015). *Guía Breve de Servicios Web*. Recuperado el 10 de Abril de 2015, de http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb

W3C Working Group Note. (11 de Febrero de 2004). *Web Services Architecture*. Recuperado el 5 de Abril de 2015, de http://www.w3.org/TR/ws-arch/

W3Schools. (2015). *W3Schools : Introduction to XML*. Recuperado el 10 de Abril de 2015, de http://www.w3schools.com/xml/xml_whatis.asp

W3techs. (17 de Febrero de 2015). *W3Techs-World WideWeb Technology Surveys: Web especialista en encuestas sobre las tecnologías web.* Recuperado el 10 de Abril de 2015, de http://w3techs.com/

Zamora, J. Á. (2014 de Mayo de 2015). *Xamarin, la API para crear aplicaciones multiplataforma en C#/.NET*. Recuperado el 27 de Marzo de 2015, de http://www.elandroidelibre.com/2014/05/xamarin-la-api-para-crear-aplicaciones-multiplataforma-en-c-net.html

Zend Technologies . (2015). *Zend*. Recuperado el 10 de Marzo de 2015, de (http://www.zend.com)

ANEXOS TÉCNICOS

ANEXO 1: PhoneGap

En el apartado correspondiente al estudio de los *frameworks* ya se ha explicado el servicio de compilador en la nube que ofrece *PhoneGap*. Este compilador lo que hace es convertir la aplicaciones basadas tecnologías web (HTML, CSS y JavaScript) en aplicaciones híbridas. Con el desarrollo anterior se ha conseguido una aplicación web y ahora mediante el compilador *PhoneGapBuild* se obtendrá la aplicación híbrida.

Este compilador empaqueta las aplicaciones web dentro de una aplicación nativa de cualquiera de las plataformas soportadas (Android, iOs, Blackberry, Windows Phone, Web Os, Symbian).

Este servicio es gratuito aunque solo es posible realizar una aplicación en caso de querer realizar más aplicaciones será necesario paga una cuota mensual. En este caso con la cuenta gratuita será suficiente. Además permite registrarnos mediante una cuenta de Adobe.

Una vez registrados solo hay que subir el código al compilador. Para subir el código es necesario cumplir con unas especificaciones. Se debe comprimir la carpeta que contiene todos los archivos CSS, HTML y JavaScript. Además dentro de esa carpeta deben aparecer dos ficheros de gran importancia:

- "index.html", fichero con extensión html que contiene la página de inicio de la aplicación
- "config.xml", fichero con extensión xml que será el que contenga las propiedades necesarias para que se compile correctamente la aplicación desde *PhoneGapBuild*. Algunos de los datos que contiene son por ejemplo: el autor y la descripción de la aplicación, tamaños de las pantallas de diferentes dispositivos, icono personalizado de la aplicación y la versión utilizada de *PhoneGap* .

Una vez subido el archivo comprimido a *PhoneGkap* el fichero comprimido de nuestra aplicación, el compilador realiza su trabajo y generará para cada plataforma los ficheros correspondientes. En el caso de que surja algún error *PhoneGapBuild* indica que ha habido un error y especifica en qué consiste dicho error facilitando el trabajo de resolución de errores.

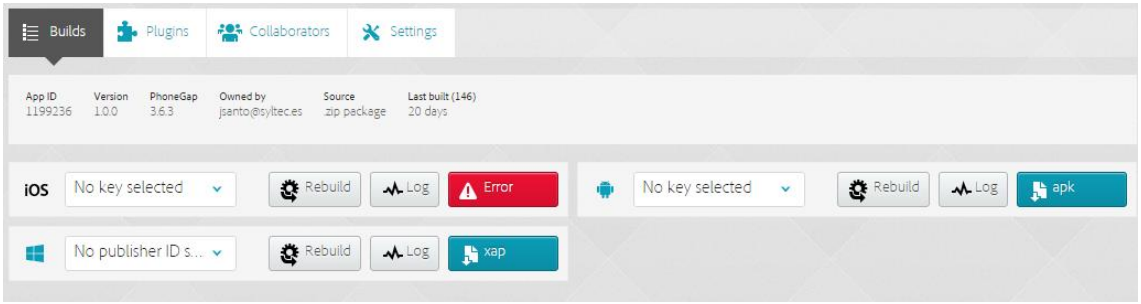


Ilustración 61. Subida de un fichero .zip a PhoneGapBuild sin errores

En la ilustración siguiente se observa como indica error en la plataforma iOS y especifica que falta la clave. Este error es debido a que en *PhoneGapBuild*, para desarrollar en la plataforma iOS es necesario disponer de un código de desarrollador que otorga la empresa Apple.

Una vez se genera el fichero correspondiente con la aplicación para la plataforma deseada, en este caso Android, será posible descargar la aplicación resultante pulsando los botones azules que hay a la izquierda del nombre de cada plataforma. Aunque la forma más habitual de instalar en un dispositivo móvil la aplicación de *PhoneGapBuild* es mediante un código QR que facilita un enlace directo de descarga de la aplicación en el dispositivo móvil.

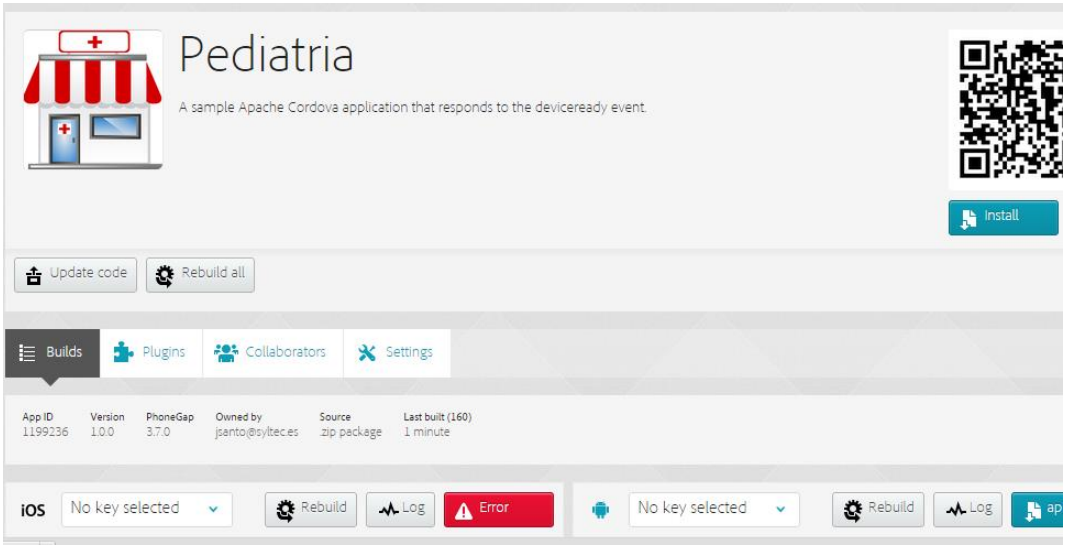


Ilustración 62. Visualización de la página web *PhoneGapBuild* una vez concluida la subida del fichero .zip con la aplicación comprimida.

ANEXO 2: Publicar aplicaciones

A lo largo de este anexo, se explicarán las fases que se deben seguir para la publicación de una aplicación, como la desarrollada, u otra desarrollada en otro lenguaje de programación, en *Google Play* (hasta hace poco llamado *Android Market*). Para publicar una aplicación, configurar su distribución o precio se hará a través de la propia herramienta utilizada para desarrollar la aplicación, desde cualquier navegador Web (Android Developers, 2014).

1. Preparando la aplicación

Se puede configurar la publicación en *Google Play* en tan solo unos minutos. He aquí cómo hacerlo:

- Primero hay que registrarse para obtener una cuenta de editor en *Google Play*.
- Si se desea vender aplicaciones, hay que configurar una cuenta de *Google Checkout* como comerciante.
- Explorar la herramienta de desarrollo para *Google Play* y aprender sobre los mecanismos para la publicación.

2. Registro para obtener cuenta editor

El primer paso es visitar la herramienta de desarrollo para *Google Play* y registrarse para obtener una cuenta de editor (Google Play, 2015).

Esto es lo que hay que hacer durante el registro:

- Visitar la herramienta de desarrollo para *Google Play* de *Android* en https://play.google.com/apps/publish/.
- Introducir la información básica acerca de su identidad de desarrollador - Nombre del desarrollador, dirección de correo electrónico, y así sucesivamente. Esta información puede ser modificada posteriormente.
- Leer y aceptar el Acuerdo de distribución para desarrolladores que se aplica a su país o región. Hay que tener en cuenta que las aplicaciones y los listados que aparecen en *Google Play* deben cumplir con las Políticas del programa de desarrollo y la ley exportación para aplicaciones de los EE.UU.
- Pagar una cuota de 25$ (USD) como registro para el uso de *Google Checkout*. Si carece de una cuenta de *Google Checkout*, se puede configurar rápidamente una durante el proceso.

Cuando el registro es verificado, se le notificará a la dirección de correo electrónico que especificó al registrarse.

3. Configurar una cuenta como comerciante *de Google Checkout*

Si se desea vender productos en *Google Play,* será necesario configurar su cuenta de comerciante, de *Google Checkout*. Puede hacerlo en cualquier momento, pero hay que asegurarse de revisar primero la lista de los países comerciantes (Google Play, 2015).

Para configurar una cuenta de hay que seguir los siguientes pasos:

1. Explorar posibilidades de la herramienta de desarrollo

Una vez que el registro se haya verificado, se puede acceder a la herramienta de desarrolladores de *Android*, la cual será la sede para sus operaciones de publicación de aplicaciones en *Google Play*.