



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

MONITOREO DE LA ACTIVIDAD FÍSICA DE LAS PERSONAS UTILIZANDO TELÉFONOS INTELIGENTES

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN
COMPUTACIÓN

PABLO SEBASTIÁN ROMO NAVARRETE

PROFESOR GUÍA:
SERGIO FABIÁN OCHOA DE LORENZI

MIEMBROS DE LA COMISIÓN:
JUAN ÁLVAREZ RUBIO
RODRIGO ARENAS ANDRADE

SANTIAGO DE CHILE
2014

Resumen

Durante los últimos años, la sociedad ha tomado conciencia de la importancia de realizar actividad física. El índice de obesidad a nivel mundial ha alcanzado niveles críticos, levantando alarmas no sólo en organizaciones de salud que operan a nivel global, sino también en los gobiernos de un gran número de países. Por otra parte, el ritmo de vida de la sociedad actual hace muy difícil la realización de actividad física, y llevar un control de éstas para tratar de mantener este aspecto de nuestras vidas bajo control. Sin embargo, no todos los cambios introducidos por la vida moderna son perjudiciales, ya que en muchos casos, el avance tecnológico ha ayudado a resolver problemas que históricamente han sido difíciles de abordar, como es el caso de la salud.

Buscando contribuir a mantener o mejorar nuestra salud, este trabajo de memoria tiene por finalidad desarrollar una aplicación móvil para el monitoreo de la actividad física de las personas, utilizando teléfonos inteligentes. A través del uso de esta aplicación, se busca que las personas tomen conciencia de cuánta actividad física realizan diariamente, y así ayudar a combatir el sedentarismo. Esta aplicación está pensada para ser utilizada bajo dos enfoques: (1) el auto-monitoreo de actividades físicas, es decir, para uso personal y (2) para el monitoreo de las actividades físicas de terceros, por ejemplo adultos mayores bajo la supervisión de un médico o de los familiares. Este último enfoque de uso, busca particularmente ayudar a los usuarios a que puedan monitorear a otras personas, sin necesidad de tener que presenciar constantemente la actividad física del otro. Es por esto, que se desarrollaron dos entornos de aplicación: (1) una aplicación móvil para uso personal y monitoreo de terceros, y (2) una aplicación Web para consultar los datos del monitoreo recolectados por la aplicación anterior.

Respecto a la primera aplicación, el servicio de auto-monitoreo, permite llevar un control de la actividad física realizada día a día por el usuario, así como también, un control de las calorías quemadas.

Además, la aplicación permite ver el tipo de actividades realizadas por el usuario en un período de tiempo. Por otra parte, el servicio de monitoreo de terceros, es bastante similar a la anterior en términos de funcionalidad; sin embargo, la persona monitoreada no siempre es consciente de que su teléfono está monitoreando su actividad física. Dicha aplicación, funciona como un servicio que sólo puede ser detenido por la persona que hace el seguimiento (monitoreo remoto).

La aplicación Web está diseñada para acceder tanto a los datos propios, como a los de los dispositivos monitoreados. Esta aplicación, muestra la actividad física de la gente a través de diversos gráficos y estadísticas. La representación gráfica de esta información busca facilitar su comprensión por parte del usuario final.

Finalmente, mencionar que la aplicación fue desarrollada para móviles con sistema operativo Android y en el caso de la aplicación Web para navegadores de computadoras o Laptops.

Agradecimientos

Primero que todo, debo agradecer inmensamente a mi familia. En un país donde, el lugar en el cual naces parece determinar tu vida, tener una familia donde mi madre y mis abuelos se han esforzado tanto por mí, es realmente una bendición. No habría forma de devolverles la mano con respecto a esto, pero sé que al menos estarán orgullosos de mí por haberme podido llevar a obtener un título profesional. También mi hermano ha sido un apoyo constante para mí, y nunca olvido que puedo contar con él para cualquier cosa.

En el aspecto académico, creo que es importante destacar a las personas que despertaron en mí, el interés por esta determinada área del saber. Agradezco a mi profesor de matemáticas desde 6to básico Luis Reyes, que me inspiró para seguir tempranamente el camino de las matemáticas, y también al profesor Rafael Labarca, que me dio una visión de la educación y las oportunidades, que hoy me han hecho llegar hasta acá. Así también, debo agradecer al profesor Sergio Ochoa que con su constante apoyo y preocupación me ayudó a llevar a cabo este trabajo, aportando siempre con soluciones que ante mí pasaban desapercibidas.

Otro aspecto muy relevante es el ámbito emocional. Me parece que todas las personas pasan momentos complejos en la vida, a veces, la falta de motivación pasa la cuenta y las habilidades quedan en un segundo plano. En este aspecto debo agradecer a mi polola Victoria Venegas, que siempre me instó a aprovechar mis habilidades y me apoyo demasiado para terminar este difícil camino. Además que ha hecho mi vida feliz, lo que se ha traducido en mucho optimismo y un mayor cariño a mí mismo, ayudándome a alcanzar mis metas.

Los amigos, me parece que también son cruciales en el camino de la educación, y acá hay dos grupos importantes que me acompañaron durante todo este trayecto. Por una parte, tengo a mis amigos del colegio que estuvieron conmigo forjando el camino hacia la universidad acompañándome hasta hoy: Alfredo, Raúl, Gonzalo, Claudio, Maxi, Guga, Romero. Por otro lado, también estuvieron conmigo los amigos que conocí en la universidad, con los que pasé muchos momentos juntos: Perro, Javier, Pato, Carlos, Monje, Jorge, Espinoza, Abogado, Oriel, Nico Mechón. Así también agradecer a amigos que conocí en otras instancias como Marraqueta, Marroquin y otro que nadie explica de donde salió, pero también se ha convertido en uno de mis mejores amigos: Joako. Quiero agradecer en general, los buenos momentos vividos junto a ustedes, su apoyo y amistad. Si se me fue alguien, no le digan.

Este trabajo de memoria ha sido parcialmente financiado por Fondecyt (Chile), Proyecto Nro: 1120207.

Tabla de Contenido

1. INTRODUCCIÓN	1
1.1. PROBLEMA A RESOLVER	1
1.2. JUSTIFICACIÓN DE LA PROPUESTA	2
1.3. OBJETIVOS DE LA MEMORIA	3
2. ANTECEDENTES	4
2.1. TRABAJOS RELACIONADOS.....	4
2.2. TECNOLOGÍAS RELEVANTES PARA EL PROYECTO	4
2.2.1. ANDROID	5
2.2.2. ALGORITMO DE DETECCIÓN DE PASOS Y APLICACIONES RELACIONADAS.....	5
3. DISEÑO	8
3.1. PRINCIPALES REQUISITOS	8
3.1.1. REQUISITOS PARA LA APLICACIÓN MÓVIL.....	8
3.1.2. REQUISITOS PARA LA APLICACIÓN WEB	9
3.2. DISEÑO ARQUITECTÓNICO DE LA SOLUCIÓN	9
3.3. DISEÑO LÓGICO	11
3.4. MODELO DE DATOS	12
4. IMPLEMENTACIÓN	15
4.1. HERRAMIENTAS DE DESARROLLO.....	15
4.2. ENTORNO TELÉFONO INTELIGENTE ANDROID	15
4.2.1. ASPECTOS ESENCIALES DEL DESARROLLO DE LA APLICACIÓN.....	15
4.2.2. MAPA DE NAVEGACIÓN.....	19
4.2.3. DETALLE DE INTERFACES Y FUNCIONALIDADES.....	21
4.3. ENTORNO DE LA APLICACIÓN WEB.....	28
4.3.1. ASPECTOS RELEVANTES DEL DESARROLLO.....	29
4.3.2. DETALLE DE INTERFACES Y FUNCIONALIDADES.....	30
4.4. ENTORNO DE SERVIDOR WEB.....	35
5. PRUEBAS DEL PROTOTIPO	39
5.1. PRECISIÓN.....	39
5.1.1 EXPERIMENTO	39
5.1.2. RESULTADOS	39

5.1.2. DISCUSIÓN	41
5.2. CONSUMO DE BATERÍA	42
5.2.1. EXPERIMENTO	42
5.2.2. RESULTADOS	43
5.2.3. DISCUSIÓN	43
6. CONCLUSIONES Y TRABAJO FUTURO	45
7. BIBLIOGRAFÍA Y REFERENCIAS	47
ANEXO A. ESPECIFICACIÓN DE REQUISITOS	49
A.1. REQUISITOS DETALLADOS DE LA SOLUCIÓN.....	49
A.1.1. REQUISITOS DE USUARIO.....	49
A.1.2. REQUISITOS DE SOFTWARE.....	51
ANEXO B. RESULTADOS EXTENDIDOS DE LOS EXPERIMENTOS	55

Índice de Tablas

Tabla 1. Aplicaciones de registro de actividad física	6
Tabla 2. Letras asignadas para despliegue de resultados en prueba de precisión	40
Tabla 3. Pasos registrados por la aplicación desarrollada para prueba de precisión	41
Tabla 4. Consumo de batería por componentes de aplicación	43
Tabla 5. Impacto del consumo de batería de la aplicación en relación al consumo total	43
Tabla 6. Pasos registrados por las aplicaciones seleccionadas para 3 velocidades	55
Tabla 7. Pasos registrados por las aplicaciones seleccionadas para 2 velocidades	55

Índice de Gráficos

Gráfico 1. Errores totales en conteo de pasos para el experimento 1	40
Gráfico 2. Porcentaje de error para el conteo de pasos en Experimento 1	41

Índice de Figuras

Figura 1. Diseño arquitectónico de la solución	10
Figura 2. Diseño lógico de la solución.....	12
Figura 3. Modelo de datos.....	14
Figura 4. Mapa de navegación.....	20
Figura 5. Pantalla de inicio/login entorno teléfono	21
Figura 6. Selección de modo de uso entorno teléfono	21
Figura 7. Registro de usuario entorno teléfono	23
Figura 8. Registro usuario monitoreado.....	23
Figura 9. Interfaz de calibración.....	24
Figura 10. Interfaz de final de la calibración.....	24
Figura 11. Interfaz principal del entorno teléfono	25
Figura 12. Historial de actividades	26
Figura 13. Detalle de actividades.....	26
Figura 14. Interfaz de bienvenida	27
Figura 15. Interfaz de bienvenida alternativa	27
Figura 16. Lista de dispositivos asociados	28
Figura 17. Resumen del dispositivo.....	28
Figura 18. Ingreso aplicación web de la solución	30
Figura 19. Pantalla bienvenida aplicación web	31
Figura 20. Lista de dispositivos en aplicación web de la solución.....	32
Figura 21. Actividades registradas en visualización por día.....	32
Figura 22. Actividades registradas en visualización mensual.....	33
Figura 23. Actividades registradas en visualización anual	33
Figura 24. Edición de datos personales en interfaz web	34
Figura 25. Cambio de contraseña interfaz web	34
Figura 26. Interfaz de pruebas de API REST del servidor web	37
Figura 27. Consumo de bacteria utilizando internet en diversos teléfonos.....	43

1. Introducción

Los índices de sedentarismo en Chile han aumentado a cifras alarmantes según los resultados de las encuestas CASEN y ECVS. Estos índices muestran que un 70% de la población no realiza actividad física alguna y un 90% es considerada sedentaria (Alcalá., 2010). Más alarmante aún, es el hecho de que los índices de sedentarismo se elevan con la edad. Por lo tanto, dado que la población a nivel mundial está envejeciendo, en algunos años más los problemas de salud por sedentarismo se acentuarán considerablemente.

Gobiernos de distintos países han concluido que los gastos en salud provocados por los altos niveles de obesidad y sedentarismo pueden controlarse con campañas que fomenten la actividad física y la buena alimentación. Chile no se encuentra exento de ello, y desde el año 2011 ha comenzado a desarrollar la campaña “Elige vivir Sano”, que fomenta actividades deportivas y cambios en los hábitos alimenticios de la población chilena. El impacto de esta campaña tendrá resultados en el futuro, sin embargo, con este tipo de campañas al menos se ha visto un cambio un cambio de actitud de los gobiernos para enfrentar el problema.

Más allá de las campañas de los gobiernos, está claro que la batalla contra el sedentarismo la debe librar cada individuo, probablemente con la ayuda de su entorno familiar. Por lo tanto tomar conciencia acerca de cuánta actividad física uno hace, y cuánta debería hacer, es el primer paso para abordar este problema. Por lo tanto, este trabajo de memoria busca desarrollar una herramienta computacional móvil, que permita llevar un autocontrol de la actividad física realizada por el usuario. Además, debe permitir realizar el monitoreo de la actividad física de terceros. Por ejemplo, en el caso de adultos mayores, es usual que los familiares directos supervisen la realización periódica de sus ejercicios. Esta herramienta, permitiría efectuar este monitoreo de forma más efectiva y flexible.

Debido a la masiva inclusión de los teléfonos inteligentes en nuestra sociedad, y a sus diversas capacidades de monitoreo y cómputo, resulta bastante evidente el uso de estos dispositivos como portador de un sistema de monitoreo de actividad física.

1.1. Problema a Resolver

En esta memoria se busca incrementar la cantidad de actividad física desarrollada por los usuarios para ayudar a resolver el problema del sedentarismo. El sedentarismo generalmente se acompaña de una mala alimentación, lo cual con el paso del tiempo, provoca distintas enfermedades como: hipertensión arterial, diabetes, artrosis, entre otras. Por otra parte, otro problema que viene acompañado del sedentarismo, es la falta de tiempo para la integración de la actividad física, por lo tanto, que se requiere una integración de estas actividades como parte de la vida cotidiana de las personas y mostrar que con pequeños esfuerzos se pueden lograr cambios. Además esta falta de tiempo trae a la vez una tercera problemática en nuestro caso de estudio: se

hace imposible para las personas llevar un control de sus seres queridos en caso de que estén con problemas de sedentarismo.

1.2. Justificación de la Propuesta

Para generar un cambio de hábitos en lo que respecta al sedentarismo de la población, se requiere que la solución impacte las actividades rutinarias de las personas. Para ello, la propuesta de solución a desarrollar en este trabajo de memoria, pretende abordar el problema desde dos enfoques distintos:

- *Enfoque de Autocontrol:* La idea es generar un desafío en el usuario y mostrarle cuán sedentario se ha comportado día a día, una forma de hacer esto es presentarle la cantidad de ejercicio desarrollado a lo largo del día, versus la cantidad mínima recomendada por especialistas según su rango etario. Se espera que el usuario al ver que día a día se encuentra bajo de la cantidad de ejercicio recomendada, genere alguna acción para combatir su sedentarismo.
- *Enfoque de Monitoreo:* La idea es poder realizar un seguimiento a la actividad física de un tercero (por ejemplo, un niño o un abuelo), de manera de apoyar y alentar dicha actividad. Mucha gente que debe realizar ejercicios diariamente (por ejemplo, por prescripción médica) y que es un tanto reacio a hacerlo, necesita ayuda externa para poder cumplir con esa meta. Si el usuario que necesita monitorear a otro puede acceder a la información remotamente, el proceso de monitoreo y posterior motivación se vuelve más sencillo. Para esto el usuario monitor contará con una aplicación compatible con la del usuario monitoreado. De esta manera, el dispositivo del usuario monitoreado recopila los datos de la actividad física, y el monitor los revisa desde algún portal Web, usando un PC o un celular. Probablemente la aplicación de monitoreo deberá funcionar como un espía, para evitar que el usuario monitoreado deje de utilizar su celular si se siente observado.

Para abordar ambos enfoques se pretende desarrollar tres aplicaciones que constituyen la solución planteada, las cuales se explican brevemente a continuación:

1. Una aplicación móvil para teléfonos inteligentes que tenga como función principal contabilizar la actividad física del usuario, en función del movimiento que se percibe a través de sus sensores inerciales (acelerómetro, cuenta pasos, giroscopio, etc.) y el GPS. En base a dicha información, se estima la cantidad de calorías que la persona ha gastado durante el día y luego se muestra al usuario para ayudarlo a tomar consciencia de la cantidad de actividad física realizada, versus la recomendada. La aplicación deberá eventualmente enviar esta información a un servidor (si es que el usuario es monitoreado por alguien más). Este servidor es el que se encarga de llevar registro de las actividades físicas de las personas monitoreadas.
2. Una aplicación Web (para PC) que permita revisar y analizar la actividad física de una persona, la cual puede ser el mismo usuario o un tercero.

3. Una aplicación servidora que reciba y administre la información del sistema. Esta aplicación debe interactuar con los otros dos componentes antes definidos.

El software a desarrollar involucra diversos desafíos técnicos y de diseño, que hacen interesante a la solución propuesta. A continuación se detallan los más relevantes:

- En el caso de la aplicación cliente para celular, la solución no sólo debe interactuar con los sensores inerciales del teléfono, sino que además debe hacerlo en forma transparente para no interferir con el uso que otras aplicaciones haga de estos sensores. La estrategia de uso de los dispositivos inerciales deberá considerar un consumo mínimo de la batería del celular.
- Las aplicaciones cliente deberán además ser multiplataforma, debido a la gran diversidad de sistemas operativos para PC que están actualmente disponibles en el mercado.
- La información recolectada desde los celulares deberá ser precisa, y mantener niveles importantes de seguridad y privacidad. Además, el canal de comunicación entre el servidor central y los dispositivos debe ser transparente y seguro.
- La usabilidad de las aplicaciones y la pertinencia de la información que ésta entregue, será un aspecto vital para que la gente la encuentre útil y efectivamente la use.
- El desarrollo del agente espía (spyware), también representa un interesante desafío.

1.3. Objetivos de la Memoria

Este trabajo de memoria tiene como objetivo general, desarrollar una solución de software que mida la actividad física que una persona realiza diariamente. En base a esto, se espera que la aplicación desarrollada estimule en los usuarios un hábito de vida que incluya la actividad física. Los objetivos específicos derivados del propósito general son los siguientes:

1. **Desarrollar una aplicación para teléfonos inteligentes** que recopile la información de actividad física de su usuario y permita visualizar diversos indicadores asociados a ésta; por ejemplo, cantidad de calorías gastadas, períodos de actividad, actividad física mínima requerida, etc. Esta aplicación eventualmente enviará dicha información a un servidor central (a través de Internet), el cual permitirá realizar análisis más complejos. Además, esta misma aplicación permitirá revisar y analizar la actividad física de los usuarios monitoreados.
2. **Desarrollar una aplicación Web (para PC)** que permita revisar y analizar la actividad física de una persona, la cual puede ser el mismo usuario o un tercero.
3. **Desarrollar una aplicación servidora** que reciba y guarde la información del sistema. Esta aplicación debe tener la capacidad de interactuar con los otros dos componentes antes definidos.

2. Antecedentes

A continuación, se presentan los conceptos y tópicos relevantes para el entendimiento y focalización del presente trabajo.

2.1. Trabajos Relacionados

Hay diversos trabajos relacionados a esta propuesta. Por ejemplo Guo (Guo 2012) desarrolló una aplicación para dispositivos Android, para llevar el control de los pasos realizados por el usuario y la cantidad de calorías consumidas al día, además de generar recordatorios de voz que inciten a retomar la actividad física. La aplicación se enfoca en usuarios con trastornos psicológicos, y podría ser muy útil para el reconocimiento de las actividades desarrolladas por los usuarios. Esto serviría como una primera versión, para llevar un registro de pasos y habilitar recordatorios de voz que inciten a la actividad física. Sin embargo, la propuesta de Guo no aborda el problema de llevar el registro de las actividades a través de la aplicación de tipo spyware.

Otro trabajo relacionado es el presentado por Oner y otros (Oner 2012). Dicho trabajo busca detectar tempranamente una posible tropiezo o caída por parte de la persona monitoreada. Como un primer paso para lograr este objetivo, los autores desarrollaron una aplicación que usando los sensores del teléfono inteligente (sin modificaciones), permite detectar movimientos como caminar o correr. Esta detección de actividades y el algoritmo mismo, podrían ser bastante útiles.

Un trabajo también relacionado con el de Oner (Oner 2012), es el realizado por Shin otros (Shin 2010). Estos investigadores desarrollaron un algoritmo que permite detectar actividades físicas. Además, realizaron una propuesta para minimizar el consumo de energía en la detección de movimientos, a través de un dispositivo que regula la potencia eléctrica consumida. Resulta interesante la minimización de la energía consumida, pero ésta depende de un dispositivo externo, lo cual quedaría fuera del marco del trabajo de memoria acá presentado. Por otro lado, el algoritmo de detección de actividades permite discernir entre: caminar, caminar en colinas, trotar, subir y bajar escaleras con un alto nivel de precisión, por lo que sería muy útil a la hora de generar los filtros para las distintas actividades a registrar en este trabajo. En particular, si se pudiera generar un doble filtro usando los algoritmos propuestos por Oner (Oner 2012) y Shin (Shin 2010), sería bastante interesante desde el punto de vista de la exactitud de las actividades registradas.

2.2. Tecnologías Relevantes para el Proyecto

En seguida, se presenta y discuten brevemente las tecnologías involucradas en este trabajo de memoria.

2.2.1. Android

Android es un sistema operativo basado en Linux, diseñado principalmente para dispositivos móviles con pantalla táctil como teléfonos inteligentes o tabletas. En la actualidad, Android se posiciona como el sistema operativo más utilizado para teléfonos inteligentes y tabletas (Liria 2013), además es utilizado por teléfonos de todo rango de precios.

Uno de los grandes beneficios del desarrollo de aplicaciones para Android, es la constante evolución del sistema operativo, su SDK y API, lo que permite al desarrollador generar aplicaciones de mejor calidad y donde la curva de aprendizaje es bastante aterrizada. Otro aspecto importante, es la gran cantidad de documentación presente en la web y la gratuidad de sus herramientas de desarrollo.

El desarrollo de aplicaciones para este sistema operativo, está principalmente enfocado en el uso de Java como lenguaje de programación, aunque también hay acceso a otras opciones como el desarrollo de aplicaciones web móviles con HTML5. En esta memoria en particular, es importante tener un contacto directo con los sensores inerciales, por lo que el enfoque es utilizar el SDK oficial y el lenguaje de programación JAVA.

Como último punto, es importante mencionar que se utilizará la API número 14 de Android, lo que hace la aplicación óptima para la versión 4.0 de Android, lo cual no determina que no se pueda utilizar en versiones anteriores, pero podría funcionar erráticamente en ellas. En la actualidad, la versión 4.0 o superior de Android ya abarca alrededor de un 70% del mercado (Google A 2013).

2.2.2. Algoritmo de Detección de Pasos y Aplicaciones Relacionadas

Para comenzar a trabajar con la detección de actividad física en dispositivos móviles, es interesante investigar cuáles son las alternativas en el mercado para determinar, cuál es la mejor al momento de implementar la aplicación. Primero que todo, se prevén dos diferentes enfoques:

1. *Aplicación de terceros*: Acceder a la información proporcionada por una aplicación externa con respecto a la actividad física. En este caso, se podrían usar APIs u otras formas para acceder a la información.
2. *Implementar algoritmos externos*: Encontrar en la literatura u otras fuentes de internet, un algoritmo de detección de actividad física que se basen en el acelerómetro en el teléfono inteligente.

A primera vista, parece una buena idea encontrar alguna aplicación de un tercero que funcione de manera correcta y usar la información obtenida por ella, para nutrir la aplicación desarrollada.

Considerando esto, se debe estudiar de qué forma guardan la información las distintas aplicaciones en el ecosistema Android, y si es posible acceder a esta información.

En toda aplicación Android, la información es guardada por defecto en el almacenamiento interno del teléfono y toda esta información es privada para otras aplicaciones (Google B 2013). Otra opción es escribir la información en la tarjeta SD del dispositivo, pero esto es público para todo el dispositivo, por lo que sería extraño que una aplicación guarde su información en esa locación, a menos que deseen que esta información sea pública, por ejemplo, al tomar una foto con alguna aplicación es preferible que esa foto quede en la tarjeta SD y sea pública para que el usuario pueda acceder a ella desde otras aplicaciones. Conociendo esto, veamos cuáles son las aplicaciones que se estudiaron y dónde guardan los datos obtenidos (Tabla 1).

Tabla 1. Aplicaciones de registro de actividad física

Aplicaciones de Podómetro
RuntasticPedometer
Podómetro–Accupedo
Smart Pedometer
NoomWalk: podómetro
Me paseopodómetro
All-in Podómetro
Podómetro Plus
Beppi'sPodómetrofácil
WalkLoggerpodómetro
New Pedometer Free
Endomondo

Estas aplicaciones fueron descargadas de la tienda de aplicaciones de Android, y son las que poseen mayor cantidad de descargas y mejores calificaciones a la vez. De todas las aplicaciones estudiadas, ninguna de ellas guarda su información en la tarjeta SD, por lo que la única posibilidad es acceder a la información privada de la aplicación. ¿Cómo podemos acceder a esta información privada? Hay dos maneras de acceder:

1. *A través de Content Providers.* Los Content Provider es la forma oficial que tiene una aplicación para compartir información privada. Para poder acceder a esta información, el desarrollador de la misma debe indicar cuáles son las URI por las cuáles se puede acceder a la información y que información proporcionan (Google C 2013).

2. *A través del modo de administrador.* Una forma no oficial, es extraer la información gracias a los permisos de administrador desbloqueables en un teléfono Android. Esta forma no es recomendada, porque el proceso de desbloquear el modo de administrar no es sencillo y sólo está destinado a usuarios avanzados, por lo que no se puede pedir como requisito que una aplicación sencilla como la desarrollada, necesite de este modo.

Por lo tanto, se buscó alguna API o un Content Provider administrado por alguna de las aplicaciones listadas, no encontrando uno público. Esto no es ninguna sorpresa, porque gran parte de las aplicaciones listadas son aplicaciones de gran prestigio, y sería raro que hicieran pública la información que obtienen para que alguien más le saque provecho a su trabajo. Es por esto, que se realizó una investigación en el dispositivo (usando las herramientas que provee la API de Android) donde fue posible encontrar que sólo una de ellas provee un Content Provider, la aplicación Runtastic. Sin embargo, el problema es que no hay ninguna documentación sobre cómo acceder a la información de este Content Provider, y sin esta información es completamente imposible intervenirlo, ya que ni siquiera es posible saber qué tipo de información entrega.

Considerando los datos expuestos, solamente quedan dos caminos; el primero es crear un algoritmo de detección de actividades, y el segundo utilizar un algoritmo de un tercero. Como no es objetivo de esta memoria crear un algoritmo de detección, si no enfocarse en el sistema de monitoreo, se decidió buscar un algoritmo de detección de actividades desarrollado por un tercero.

Se hizo una pequeña investigación en la literatura sobre diversos algoritmos, pero muchos no indicaban cuál era el algoritmo completo que habían estudiado o eran muy complejos de comprender e implementar para este trabajo de memoria. Por otra parte, si se encontraron otros algoritmos en la web, pero se decidió considerar el algoritmo de código libre con mejor puntuación en el sitio GitHub, que además no posee restricciones para su uso.

3. Diseño

Para la aplicación desarrollada, se seleccionaron los requisitos mínimos para solucionar al problema expuesto. Por otra parte, se presenta la arquitectura del sistema y el diseño detallado del mismo.

3.1. Principales Requisitos

A continuación, se presentan los principales requisitos divididos por aplicación relacionada:

3.1.1. Requisitos para la Aplicación Móvil

Los requisitos de usuario asociados a la aplicación móvil son los siguientes:

Identificador	RU0001
Nombre	Registro de actividades
Descripción	Registrar las actividades físicas realizadas por la persona que utiliza el teléfono inteligente.
Prioridad	Crítica
Fuente	Cliente/Usuario
Estabilidad	Intransable
Estado	Cumple

Identificador	RU0002
Nombre	Registro de dispositivo para monitoreo
Descripción	Un usuario al instalar la aplicación, puede determinar de que este equipo será monitoreado desde su cuenta de usuario, para obtener resúmenes diarios de la actividad física del usuario del equipo.
Prioridad	Crítica
Fuente	Cliente/Usuario
Estabilidad	Intransable
Estado	Cumple

Identificador	RU0003
Nombre	Historial de actividades
Descripción	El usuario de la aplicación puede revisar un resumen de sus actividades diarias.
Prioridad	Crítica
Fuente	Cliente/Usuario
Estabilidad	Intransable
Estado	Cumple

Identificador	RU0004
Nombre	Sincronización de resúmenes diarios
Descripción	Los resúmenes diarios son sincronizados a través de internet con la aplicación web. No necesita la intervención del usuario.
Prioridad	Crítica
Fuente	Cliente/Usuario
Estabilidad	Intransable
Estado	Cumple

Identificador	RU0005
Nombre	Uso de batería
Descripción	El uso de batería por parte de la aplicación completa, debe permitir que el usuario utilice su teléfono de la manera habitual.
Prioridad	Deseable
Fuente	Cliente/Usuario
Estabilidad	Intransable
Estado	Cumple

3.1.2. Requisitos para la Aplicación Web

Los requisitos de usuario asociados a la aplicación Web son los siguientes:

Identificador	RU0005
Nombre	Monitoreo a través de la versión web
Descripción	El usuario puede ingresar a la aplicación Web para consultar los resúmenes de la actividad física de los equipos monitoreados.
Prioridad	Crítica
Fuente	Cliente/Usuario
Estabilidad	Intransable
Estado	Cumple

3.2. Diseño Arquitectónico de la Solución

La arquitectura del sistema está constituida por tres componentes: Un entorno para teléfonos inteligentes Android, un entorno para PC y el servidor web que se encarga de comunicar a ambos. Los tres componentes se comunican a través de internet (ver Fig. 1):

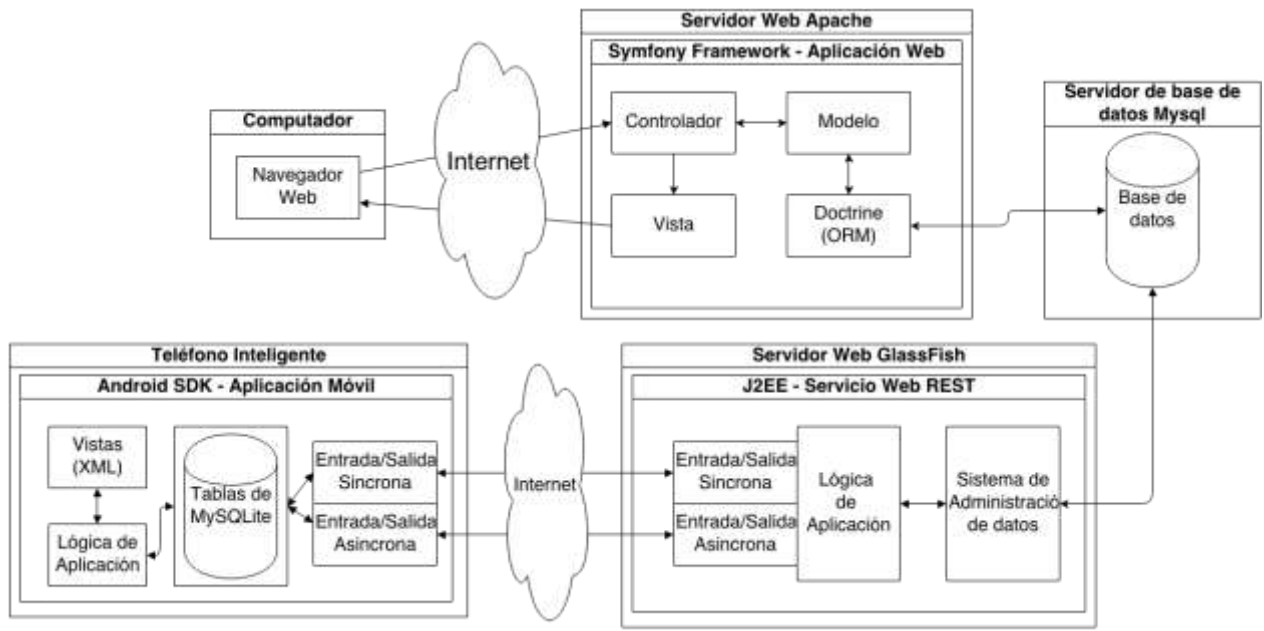


Figura 1. Diseño arquitectónico de la solución

En primera instancia, tenemos nuestra aplicación móvil donde las interfaces de usuario o vistas interactúan con la lógica de la aplicación. La lógica de la aplicación móvil, interactúa con los proveedores de datos presentes en la aplicación, esto consulta la información local disponible y se accede de manera síncrona y asíncrona a la información proporcionada por el servicio Web Rest. Toda información que se desea escribir en la base de datos global, también grabada en la base de datos local de la aplicación, para así acceder a estos datos aunque el dispositivo no esté conectado. Esta aplicación fue desarrollada usando el SDK de Android.

Por el lado de la aplicación Web se utilizó el framework Symfony, el cual está montado en un servidor Web Apache. Este framework, aplica el patrón de diseño Modelo-Vista-Controlador como muchos otros framework basados en PHP. El modelo y la base de datos están conectados usando las librerías de Doctrine, las que permiten mapear la base de datos relacional a objetos, lo que aliviana de gran manera el manejo de la persistencia y lectura de datos.

El servidor Web Rest recibe y envía información a la aplicación móvil. La información es enviada y recibida a través de request HTML. Este servidor es el encargado de procesar la información recibida, para obtener o escribir información directamente en la base de datos.

Un punto que llama la atención, es que hay dos conexiones distintas a la base de datos. Primero que todo, en este tipo de sistemas es muy buena idea incluir un servidor de tipo REST porque no es viable una conexión directa a una base de datos desde un teléfono móvil. La cantidad de cuelgues o fallas de conexión, haría inusable la aplicación. Por otra parte, se podría usar este mismo servidor para proporcionar información a la aplicación web, pero se decidió usar Doctrine, ya que es recomendado su uso para Symfony. Además, el programador posee entrenamiento y conocimientos de su uso.

Es importante recalcar, que el caso de la aplicación web es distinto a la del dispositivo móvil en términos de conexión a la base de datos, por el lado de la conexión el servidor web estaría en el mismo servidor físico que el servidor de bases de datos, con esto la conexión es directa y no es importante el intermediario.

3.3. Diseño Lógico

En las siguientes líneas, se presenta la estructura lógica de la aplicación con sus dos entornos: Entorno Web y Entorno Dispositivo Móvil. Además, se integran los servidores que permiten la comunicación con la base de datos.

El entorno de dispositivo móvil, representa la aplicación desarrollada. Por una parte, tenemos la interfaz de usuario y sus determinados módulos que permiten al usuario registrarse, acceder a la información de otros dispositivos asociados y a la información recopilada por el mismo. La información recopilada es la actividad física detectada por el servicio asociado a la aplicación. El servicio tiene como tarea emplear un algoritmo de detección de actividades, que utiliza como fuente de información el acelerómetro, permitiendo así llevar un registro de la actividad física. Podemos observar que el servicio se encuentra aparte de la interfaz de usuario, e incluso se podría decir que es un módulo completamente aparte, esto se da porque este servicio debe estar constantemente recopilando información desde que se enciende el teléfono, y su ciclo de vida no depende del ciclo de vida de la aplicación. La única interacción entre las vistas y el servicio es para mostrar recopilados, ya que se despliega la cantidad de actividad física para cada instante.

La aplicación móvil posee una interacción entre datos almacenados localmente, y datos obtenidos desde internet, siempre la primera opción es obtener datos a través de internet usando el módulo de cliente REST, pero si no está disponible la conexión se podría acceder a los datos almacenados localmente. Este servidor web funcionará de intermediario entre la base de datos y el dispositivo móvil. Este paso intermedio, permite que la información se envíe simplemente a través de *request* HTML y sean recibidas de la misma forma, todo esto gracias al módulo de cliente REST y al módulo de servidor REST en el servidor. Una vez recibida la información, se procesa por cada tipo de dato posible y se inserta en el servidor de base de datos, así también para el caso de lectura de información desde el mismo.

Por otra parte, tenemos la aplicación Web. Su principal tarea, es permitir a los usuarios acceder a la información recopilada a través de los dispositivos móviles. Considerando esto, sus módulos lógicos se basan en visualizar los dispositivos asociados a la cuenta del usuario y obtener historiales de la actividad física, recopilados por aquel dispositivo. También se pueden hacer cambios en los datos de registro de los usuarios o dispositivo. Para obtener y escribir datos, se comunica directamente con la base de datos global a través de Doctrine, encargada de realizar las conexiones y transformar las tablas recibidas a objetos para su fácil uso en la aplicación.

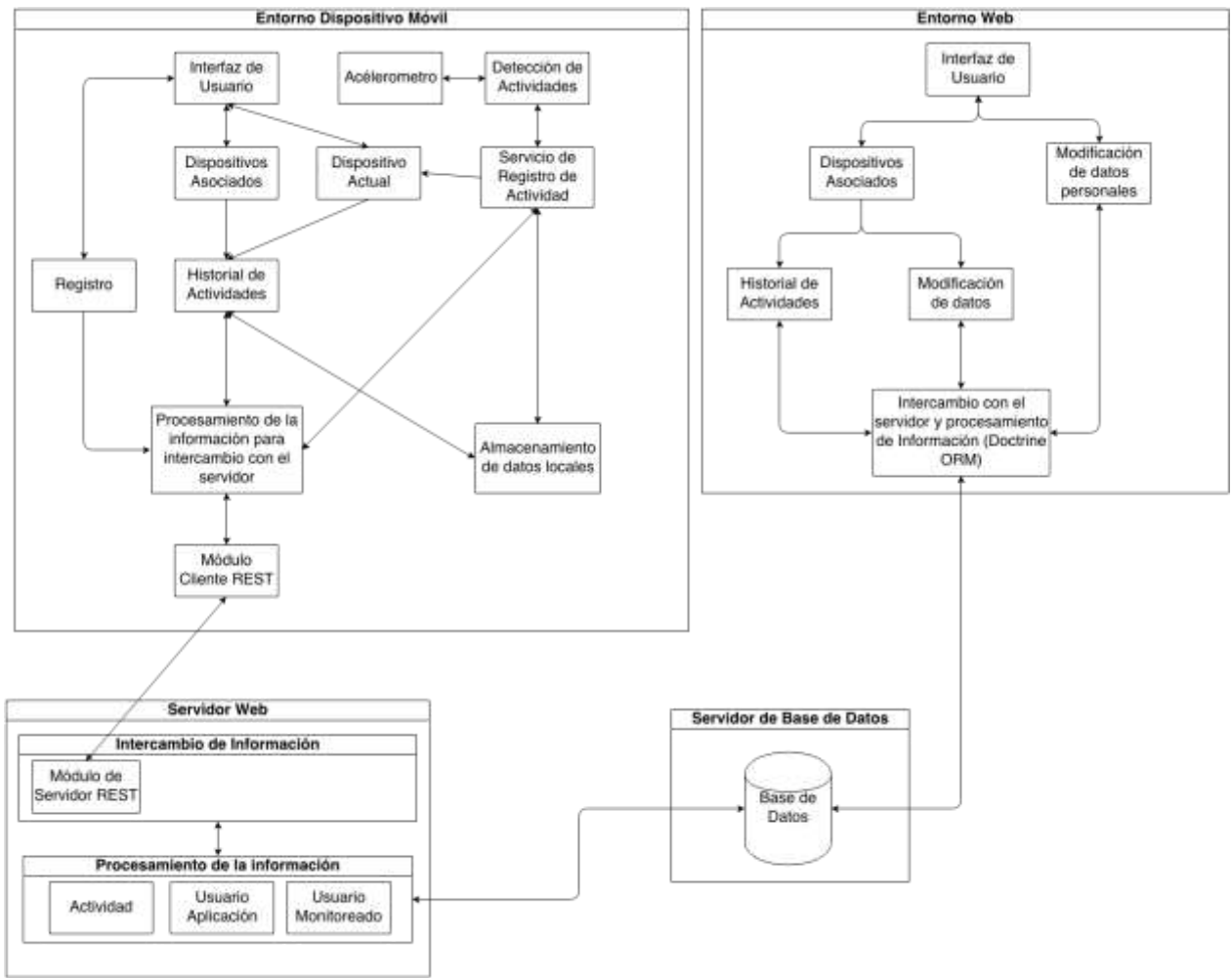


Figura 2. Diseño lógico de la solución

3.4. Modelo de Datos

En esta sección, se presenta una descripción del modelo de datos con sus características, una explicación de las tablas que lo componen y un diagrama con el resumen de lo descrito.

Todo usuario del sistema debe poseer un usuario en el mismo. Este usuario posee un email y password que le permiten ingresar al sistema, además de cierta información personal, como su Nombre, Apellido, Edad y Altura. La Edad y Altura se utilizan para determinar la cantidad de calorías que debe quemar al día, y determinar el largo de los pasos.

Todo usuario que ingresa al sistema por primera vez, se le consulta si desea usar la aplicación para monitorearse si mismo o a alguien más. En cualquiera de las dos opciones, se crea un usuario con los datos nombrados anteriormente, y si escoge la opción de monitorear a alguien

más, se crea un Usuario Monitoreado que está compuesto de los siguientes datos: Nombre, Apellido, Edad y Altura.

Otro punto importante, es que el sistema automáticamente crea en ambos casos un Dispositivo, este dispositivo representa al dispositivo en que se está realizando el registro, y está compuesto por un Label que se asigna automáticamente con el nombre del dispositivo seguido del Nombre del usuario (o usuario Monitoreado en el caso de ser asignado para monitoreo). Este Label permite identificar al dispositivo de parte del usuario.

Una actividad, es el registro del resumen de las actividades diarias. Hay 3 tipos de actividades, y para cada una de ellas se registra un resumen diario. Las actividades disponibles son: caminar, correr y andar en bicicleta. Se presenta como un resumen porque no se registra cada actividad que el usuario realiza, lo que se registra es un resumen diario del total de la actividad con actualizaciones cada una hora.

Toda actividad registrada está vinculada a un dispositivo en particular, y los usuarios monitoreados están vinculados al dispositivo que los monitorea. Por otra parte, todo dispositivo está vinculado a su respectivo usuario para que este pueda ver la actividad y administrar todos los dispositivos asociados a él. A continuación, se presentan las tablas del modelo que se pueden observar en la figura 3:

- *Gender*: Almacena los géneros disponibles para los usuarios, masculino o femenino.
- *User*: Almacena los datos del usuario registrado. La búsqueda de usuarios se hace usando el mail entregado, por lo que éste no puede estar duplicado.
- *Device*: Asocia el dispositivo con el usuario que lo monitorea.
- *MonitoredUser*: Almacena los datos del usuario monitoreado y la asociación con el dispositivo.
- *Activity*: Almacena el registro de todos los resúmenes diarios de las actividades registradas por el dispositivo.
- *ActivityType*: Almacena los tipos de actividades disponibles.

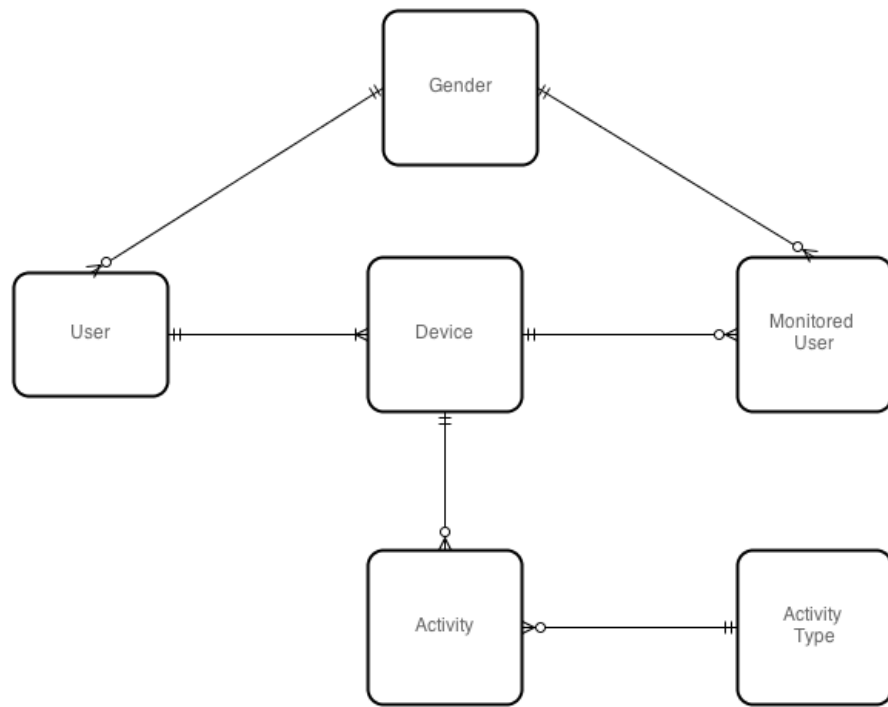


Figura 3. Modelo de datos

4. Implementación

En este apartado, se describen cada uno de los entornos que componen el sistema: Entorno de teléfono inteligente, entorno de PC, servidor Web. En primer lugar, se presentan las herramientas utilizadas, después se describen los distintos entornos en el orden en el que fueron desarrollados. Cada uno de los entornos, presenta la interfaz que permite entender de mejor manera el sistema descrito.

4.1. Herramientas de Desarrollo

Se utilizaron las siguientes herramientas de desarrollo:

- *ADT Bundle para Linux*: Este bundle se compone de Eclipse versión 4.2.1 y el SDK de Android.
- *Netbeans 7.4 IDE for J2EE*.
- *Git 1.8.4.1*.

4.2. Entorno Teléfono Inteligente Android

Inicialmente, se explica los tres puntos centrales en el desarrollo de la aplicación, luego se presenta el mapa de navegación y su descripción para pasar al detalle de las interfaces y sus funcionalidades.

4.2.1. Aspectos Esenciales del Desarrollo de la Aplicación

Los tres puntos centrales de esta aplicación son: la detección de actividades, el registro de datos y la visualización de equipos monitoreados.

4.2.1.1. Detección de Actividades

La detección de actividades, se realiza utilizando el acelerómetro de 3 dimensiones del teléfono inteligente. La información obtenida a través del acelerómetro, es procesada por un algoritmo de código abierto y libre para todo uso compartido en los repositorios Github. A grandes rasgos, el algoritmo compara los peaks mínimos y máximos extraídos por el acelerómetro, calcula su diferencia y la compara con el factor de sensibilidad para determinar si el movimiento detectado por el acelerómetro es un paso o no. Se realizan otro tipo de optimizaciones para mejorar el

cálculo del algoritmo, pero en gran parte estas optimizaciones vienen de pruebas empíricas realizadas por el desarrollador y no de un estudio científico.

Uno de los puntos más importantes de esta detección de actividades, es que debe cumplir como mínimo dos requisitos:

1. Detectar actividades a todo momento: No es necesario que el usuario mantenga la aplicación en funcionamiento para que se detecten actividades. Incluso, si el usuario reinicia el teléfono, no debe ser necesario que ejecute nuevamente la aplicación para que se comience a detectar las actividades. Esto es crucial para el monitoreo de terceros, ya que, ellos no saben que este servicio está en ejecución.
2. Permitir el uso normal de la batería: El requisito anterior implica que el acelerómetro funcione a cada momento, y así también el algoritmo que determina si el movimiento es una actividad o no. Es importante determinar, si el efecto de la detección de actividades permite que el usuario continúe con el uso normal de su dispositivo.

Para cumplir el primer requisito, es necesario construir un servicio que se ejecute constantemente en el uso diario del teléfono, que no dependa de que la aplicación se esté ejecutando y que se inicie cada vez que el teléfono se enciende. Para construir este servicio, se utilizó una clase de tipo Service que es la determinada para ejecutar operaciones en background, en particular que siga en ejecución sin la necesidad de tener la aplicación en ejecución. Un ejemplo de servicio, es la reproducción de música donde el usuario desea que siga reproduciéndose aunque salga de la aplicación. Por otra parte, para permitir que el servicio se ejecute cada vez que se inicia el teléfono se solicita un permiso particular llamado: `BOOT_COMPLETED` lo que le permite iniciarse en cuanto el teléfono está preparado para ser usado.

El segundo requisito, es un poco más complejo. En general, gran parte de las aplicaciones que buscan llevar registro de actividades físicas hacen uso del GPS y se enfocan en actividades particulares, porque se hace inviable utilizar el GPS durante un día completo, además no pueden reconocer actividades indoor. Por otra parte, la mayoría de las que hacen uso del acelerómetro también están enfocadas en actividades particulares, y no en el uso constante durante todo el día. En este sentido, se buscará minimizar el envío de datos a través de internet, enviando paquetes cada una hora con los datos recopilados y se estudiará el impacto en el consumo de batería del uso constante del acelerómetro.

4.2.1.2. Registro de Datos

Los datos de la aplicación son administrados por tres diferentes fuentes:

- *Datos de sesión local*: Los datos sesión local, permiten identificar al usuario que está monitoreando el dispositivo, es decir, utilizando la aplicación. Este enfoque (en el que un

solo usuario administra la aplicación), es recurrente en las aplicaciones móviles. Las sesiones se establecen para un usuario a la vez, y este usuario necesita desvincularse de la aplicación para que otro usuario pueda utilizarla. Se ha flexibilizado este punto, permitiendo que otros usuarios ingresen a la aplicación para consultar los datos de sus dispositivos asociados, pero sin la posibilidad de vincular este dispositivo a su cuenta. Para que otro usuario pueda vincular este dispositivo, el usuario propietario de la aplicación debe haberlo desvinculado antes. Los datos de sesión local son llamados SharedPreferences en Android, y son generalmente datos que requieren de rápido y reiterado acceso evitando las consultas a la base de datos, un ejemplo típico es el nombre de usuario.

- *Base de datos local:* Es la forma en que la aplicación guarda la información obtenida a través del servidor web. La base de datos local está creada como una base de datos SQLite. Un punto importante a considerar es la comunicación con la base de datos, para esto se consideró la creación de ContentProviders con la finalidad de comunicarse con esta base de datos de una manera simple y consistente. Los ContentProviders ofrecen formas de abstracción del contenido de la base de datos, y se manejan a través de URIs (Unique Resource Identifier) de forma similar a los recursos manejados a través de repositorios REST. Los datos se reciben en filas a través de la clase Cursor de Android. Esta implementación se divide en 3 partes:

- *Bajo nivel:* Una clase para cada tabla, encargada de crear su respectiva tabla a través de comandos de SQL. Además, se crea una función para la actualización del esquema de la tabla.
- *Alto Nivel:* Un ContentProvider para cada tabla. El ContentProvider está encargado de definir los métodos: Query, Insert, Update y Delete. Estos métodos permiten definir distintas URIs para acceder a la información en casos particulares. Por ejemplo, si quisiera consultar los datos de un determinado usuario solo sabiendo su email, se debe crear una URI del estilo: “content://memoria/user/email/*”, donde el * funciona como un wildcard para ingresar el mail a buscar. En particular, para saber los datos de usuario de alguien con un email: memoria@memoria.com, me basta invocar a Query entregando el siguiente URI:

Content://memoria/user/email/memoria@memoria.com

- *Conexión entre ambos:* Tenemos clases que crean cada una de las tablas, sin embargo, necesitamos una clase que cree la base de datos completa y se comunique con la implementación de alto nivel. Aquí aparece una subclase de SQLiteOpenHelper, que nos permite crear la base de datos y generar la instancia a

la cual accederá el ContentProvider para enviar sus consultas, inserciones, etc. Además también se define el método de actualización de las tablas.

- *Base de datos compartida en Internet:* Permite tener acceso a la última información disponible para refrescar los datos desplegados en la aplicación. Por otra parte, es necesario notificar al servidor de los cambios locales del registro de la actividad física (esta notificación se hace cada una hora). Así también, integrar el proceso de registro en la aplicación. La implementación se divide en las siguientes partes:
 - *Comunicación con el servidor:* La comunicación con el servidor es a través de requests HTTP. Se utilizan los métodos GET, POST, DELETE y UPDATE. Las request son dirigidas a las URI definidas en el servidor web. Los datos enviados al servidor son datos en el formato JSON (JavaScript Object Annotation).
 - *Modelos:* Para un mejor encapsulamiento de la información enviada y recibida, se generan clases que representen los datos a recibir como objetos. Estos modelos permiten manipular fácilmente los datos a enviar y administrar los recibidos.
 - *JSON:* Es el formato de intercambio de datos. Se utiliza la librería GSON, que es la encargada de transformar objetos de JAVA a JSON y viceversa. Los datos enviados a través del request HTTP son los modelos formateados a JSON por GSON. Una vez recibida la respuesta, si es necesario, se decodifica el JSON recibido usando nuevamente la librería GSON que me entrega los objetos de acuerdo a los modelos definidos.

Uno de los aspectos fundamentales de la aplicación, es la capacidad de monitorear dispositivos remotos. Se podría pensar de que la forma óptima de monitoreo, en cualquier ámbito, es obtener al instante la información de estado del objeto monitoreado, en este caso, obtener la cantidad de actividad física desarrollada hasta el preciso instante que se hace la consulta. ¿Cuál sería el problema de este enfoque? el alto consumo de batería, pues a cada instante se debe enviar la información obtenida de la detección de actividad física provocando un alto impacto en las horas efectivas de batería del teléfono. Considerando esto, se planteó generar resúmenes diarios que se actualizan solamente cada una hora. La implementación para este tipo de registro se desarrolló utilizando la clase AlarmManager del SDK de Android que permite generar alarmas que se repiten en determinados momentos del día y que a la vez son capturadas por el servicio de detección de pasos para registrar la información obtenida.

4.2.1.3. Visualización de Equipos Monitoreados

Como ya se ha presentado, se dispone de una versión web donde los usuarios podrán monitorear a los dispositivos que tienen asociados. Esto podría ser suficiente, pero hoy en día en nuestro

ajetreado mundo, las personas tampoco tienen tiempo para acceder a un computador, por lo que, se debe al menos presentar la posibilidad de desplegar la información de los dispositivos asociados desde un dispositivo móvil, y si se quiere un informe más detallado se puede visitar la versión web.

El usuario al ingresar a la aplicación deberá seleccionar si desea visualizar los dispositivos asociados. La aplicación internamente genera una petición asincrónica al servidor web para cargar la lista de dispositivos asociados al usuario, si el servidor no está disponible se obtiene la información desde la base de datos local. Una vez seleccionado el dispositivo, se obtienen los datos y un resumen de la cantidad total de pasos y calorías quemadas. Así también, se puede acceder a un historial de las actividades donde se puede ver el detalle de cada día el resumen total de la actividad física realizada. Esta lista de días junto con sus actividades, también es obtenida de manera asincrónica. Es importante mencionar que si esto no fuera así y ocurriera un fallo de red o una demora en obtener la información, la aplicación se mostraría completamente “pegada”, por lo que es importante generar hilos de ejecución que obtengan esta información y sean paralelos al hilo principal de la aplicación. Considerando esto, Android provee de la clase AsyncTask que permite desprendernos del manejo de thread, y hacer las consultas al servidor en background mientras el thread principal sigue preocupado de las vistas para que la aplicación no parezca colgada.

4.2.2. Mapa de Navegación

Primero que todo, la aplicación muestra un Login simple y un botón de registro para usuarios que no estén registrados (Pantalla de inicio/Login). En el caso de seleccionar registrarse, se le consulta al usuario si va a monitorear su actividad física personal o monitoreará a alguien más (Tipo de Uso). Ambas opciones lanzan a la misma vista de registro para usuarios, pero si escogió la opción de monitoreo, al término del ingreso de datos de registro se le enviará a una vista para ingresar los datos de la persona a monitorear (Registro de Usuario y de Usuario Monitoreado). Con el registro completado se presenta una vista para calibrar la detección de pasos, se le pide al usuario activar la calibración y dar 10 pasos con el teléfono en el bolsillo, después de esto presionar el botón que termina con la calibración (Calibración). Terminando con la calibración, se accede a la vista principal que presenta la cantidad de pasos realizados en el día, la velocidad de los pasos realizados y las calorías quemadas en el día, además permite ingresar al registro de actividades de días anteriores y desvincular el dispositivo (Vista Principal) .

Una segunda parte del mapa de navegación, corresponde a la vista después de que un usuario ingresa a través del Login (Lista de dispositivos). Cuando el usuario ingresa, se despliegan dos opciones: ver todos los dispositivos asociados a su cuenta o ver este dispositivo. Esta última opción sólo aparece disponible siempre y cuando el dispositivo actual esté asociado a su cuenta. Si selecciona ver todos los dispositivos y dentro de esa vista selecciona alguno, llegará a una nueva vista que es un resumen de la cantidad de pasos realizados y las calorías quemadas, además

de botones para ingresar al historial de actividades y para desvincular el dispositivo. Esta es la misma vista que se obtiene si selecciona ver los datos de este dispositivo y este es un dispositivo monitoreado, en cambio, si el usuario usa este dispositivo para monitoreo personal se le despliega una vista como la descrita anteriormente, donde se le presentan la cantidad de pasos y calorías actuales (Vista principal). A continuación, se presenta el mapa de navegación de la aplicación:

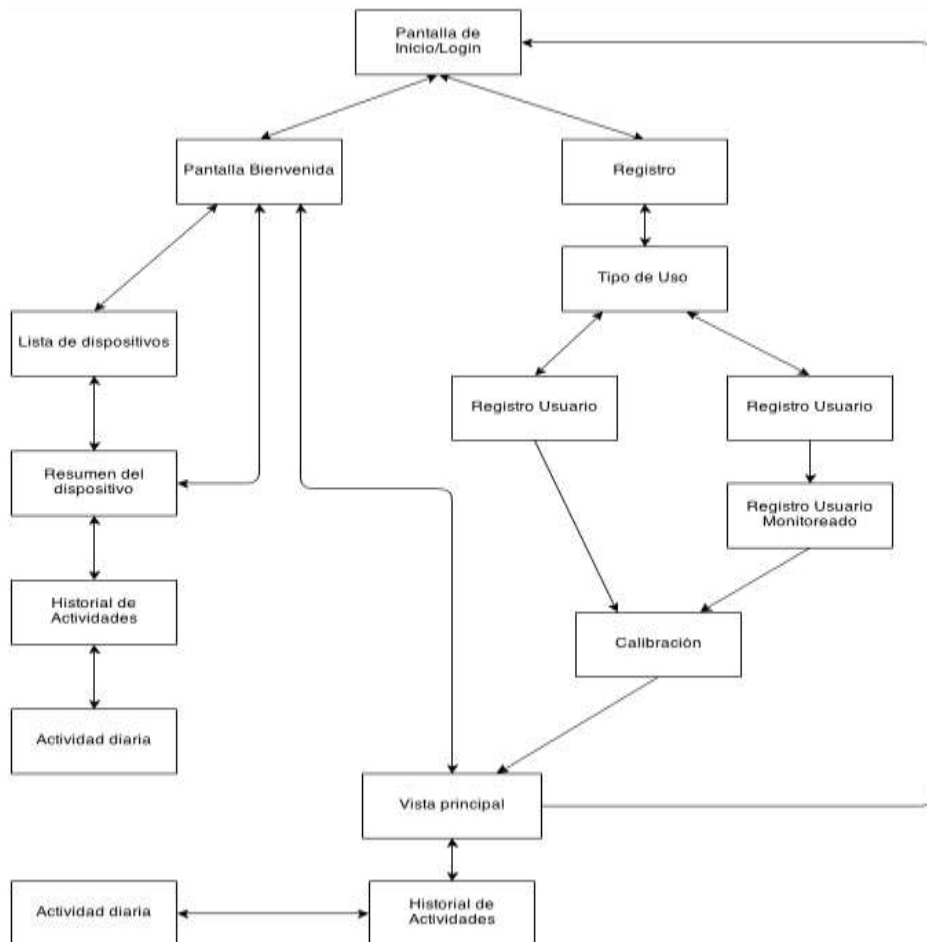


Figura 4. Mapa de navegación

Un detalle que es interesante mencionar, es que en el caso de que el usuario use la aplicación en su dispositivo personal, al momento de ingresar a la aplicación no deberá ingresar nuevamente a través del Login e inmediatamente saltará a la vista principal. Esto permite que aunque no tenga conexión a internet en un determinado momento, pueda ver el historial de actividades del dispositivo y verificar cuánta actividad ha realizado en el día. En el caso de que el usuario del dispositivo esté siendo monitoreado por un tercero, solamente verá el Login y no podrá registrarse, esto impide desvincular el dispositivo del monitoreo y permitir seguir recopilando información.

Como se puede observar, algunas vistas aparecen duplicadas en el mapa de navegación. Las vistas duplicadas son exactamente iguales para el usuario, pero aparecen duplicadas en el mapa para definir de manera correcta los lugares donde se puede dirigir un usuario y donde retorna.

4.2.3. Detalle de Interfaces y Funcionalidades

En esta sección, se describirá cada una de las interfaces presentadas en el mapa de navegación con sus respectivas funcionalidades. Además, se incluyen capturas de las interfaces para mejor entendimiento y algunos detalles de implementación utilizados.

4.2.3.1 Pantalla de Inicio/Login

Como se explicó en la descripción del mapa de navegación, el usuario tiene dos posibilidades: Ingresar a la aplicación si ya está registrado o registrarse si no lo ha hecho. Los datos ingresados por el usuario para hacer ingreso a la aplicación, son enviados a través de un request HTML de tipo POST, usando el protocolo REST al servidor, el cual verifica los datos con respecto a la base de datos para permitir el ingreso del usuario. Un aspecto importante de considerar, es que para poder ingresar a la aplicación a través de este medio siempre es necesaria una conexión a internet. Esto mantiene seguro el monitoreo del dispositivo. También obligaría al usuario a tener una conexión a internet constante, en el caso de querer verificar la información de la actividad del día o de días anteriores, pero cuando un usuario se registra para utilizar el dispositivo personalmente, se salta inmediatamente el inicio de sesión hacia la vista principal. Podría verse como una falla de seguridad el hecho de no verificar que la contraseña haya cambiado, pero esta verificación se hace apenas se tenga conexión a internet y el usuario intente ingresar nuevamente, además es común que las aplicaciones sean permisivas en este aspecto para aplicaciones móviles.

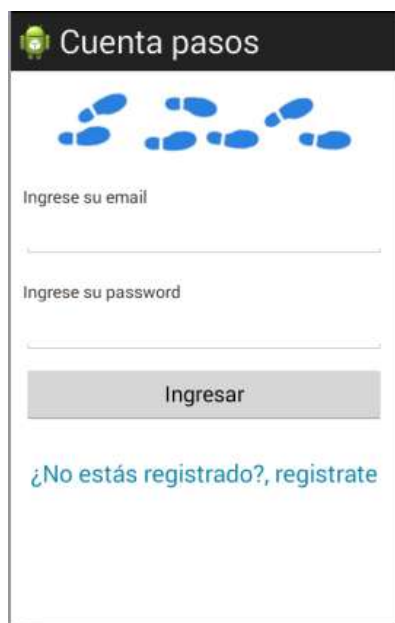


Figura 5. Pantalla de inicio/login entorno teléfono

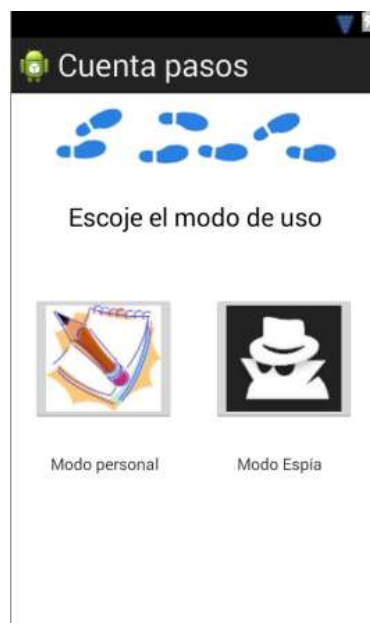


Figura 6. Selección de modo de uso entorno teléfono

4.2.3.2 Registro

Se presentan las interfaces del proceso de registro, y se describe sus funcionalidades en conjunto con el proceso realizado para este registro.

En primera instancia, el usuario que ingresa al registro se le consulta si desea usar la aplicación para su uso personal o en modo espía (Figura 6). El modo que se escoja se guarda en la sesión de aplicación o datos de sesión local. Esto es importante porque este dato es verificado para determinar algunas acciones posteriores en la aplicación.

Cuando el usuario ya escogió alguna opción, sin importar cuál de las dos opciones haya elegido, es necesario que el usuario se registre en la aplicación. La verificación de la creación de cuenta también, se debe hacer con una conexión a internet presente, esto se debe a que se verifica que el correo ingresado por el usuario no esté repetido en la base de datos global y a la vez se debe crear la cuenta en esta base de datos. El proceso de verificación de email y verificación de cuentas, se realiza a través de request HTML del tipo POST que envían el email o los datos del usuario, respectivamente para que sea el servidor el encargado de procesar las consultas. Se podría considerar que cada vez que se haga una consulta al servidor, se descargue completamente la base de datos de usuarios y después se verifique solamente en el teléfono si el email ya está siendo utilizado, pero esto sería muy costoso en términos de datos y totalmente innecesario. Una vez confirmados los datos del usuario, se crea también automáticamente el dispositivo asociado en la base de datos que tiene como *label*: “modelo del teléfono + nombre del usuario + apellido del usuario” lo que permite identificarlo en el caso de tener muchos dispositivos asociados.

Si el usuario ya registrado en la vista anterior, decidió utilizar la aplicación en modo espía se accede a la vista de la Figura 8. Debe ingresar los datos del usuario a monitorear. Estos datos también son enviados a través de un POST de HTTP utilizando la URI correspondiente. Un pequeño detalle que también se considera, es que se actualiza el nombre del dispositivo creado en el paso anterior, donde se cambia el *label* del mismo por: “modelo del teléfono + Nombre del usuario monitoreado + Apellido del usuario monitoreado”.


 Ingrese su nombre

 Ingrese su apellido

 Ingrese su email

 Género
 Hombre Mujer
 Ingrese su edad
 _____ años
 Ingrese su password

 Confirme su password

Figura 7. Registro de usuario entorno teléfono


 Ingrese los datos de la persona a monitorear
 Ingrese su nombre
 | _____
 Ingrese su apellido

 Ingrese su edad
 _____ años
 Género
 Hombre Mujer

Figura 8. Registro usuario monitoreado

4.2.3.3 Calibración

Una vez terminado el proceso de registro, se presenta la etapa de calibración de la detección de pasos. La idea es que el usuario presione el botón “Comenzar Calibración”, después de eso debe dar 10 pasos y al terminar estos pasos presionar el botón “Terminar Calibración”. El proceso de calibración lanza el servicio que detecta la cantidad de pasos. Se detecta la cantidad de pasos y se determina si la sensibilidad debe crecer o disminuir, en caso de que la cantidad de pasos sea menor a diez o mayor a diez respectivamente. El ajuste de dicha sensibilidad es lineal.

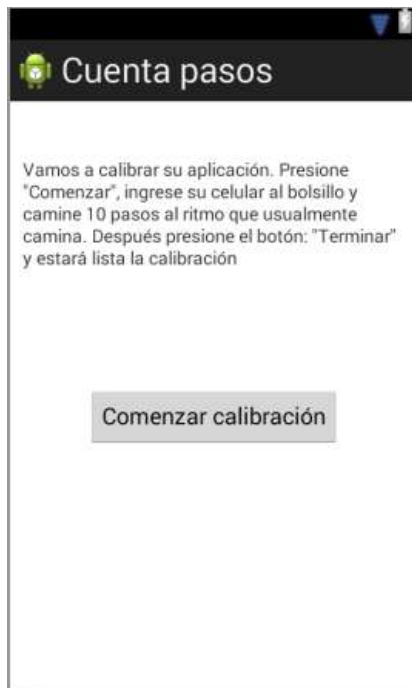


Figura 9. Interfaz de calibración

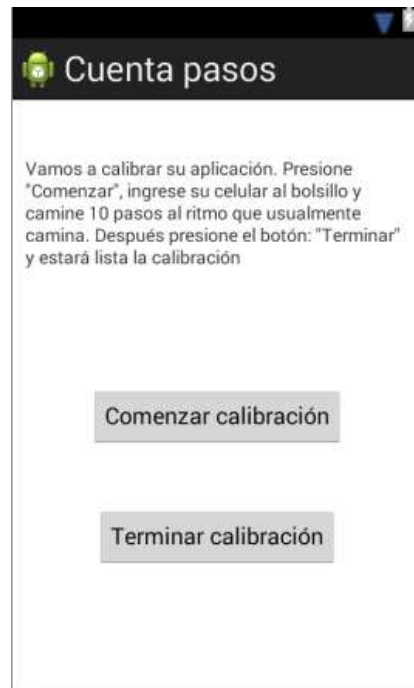


Figura 10. Interfaz de final de la calibración

4.2.3.4 Vista Principal

Ingresando a la vista principal, se presenta el nombre del usuario y dos parámetros:

- *Velocidad Media*: Indica la velocidad que tiene el usuario al caminar. El cálculo de la velocidad, se realiza solamente considerando la velocidad cuando el usuario ha realizado 4 o más pasos. Esto permite descartar los momentos en que el usuario no está caminando, y mostrar una velocidad más coherente con el movimiento realizado.
- *Número de pasos*: Presenta el número de pasos realizado durante el día. El número de pasos es enviado cada una hora al servidor web para la actualización y visualización desde el sitio web. Cada vez que se enciende el teléfono, se verifica si ha hecho un conteo de pasos del día, y si no se ha hecho, se resetea y se envía la creación del nuevo día de actividad en la base de datos. Esto se hace a través de un request HTML POST al servidor REST. Si se debe mandar una actualización de pasos, esta se hace a través del método de HTTP UPDATE que envía el identificador de la actividad con la cantidad de pasos, calorías y velocidad a actualizar. Si la conexión a internet no se encuentra disponible, no hay mayores problemas, pues en la siguiente hora se intentará nuevamente.

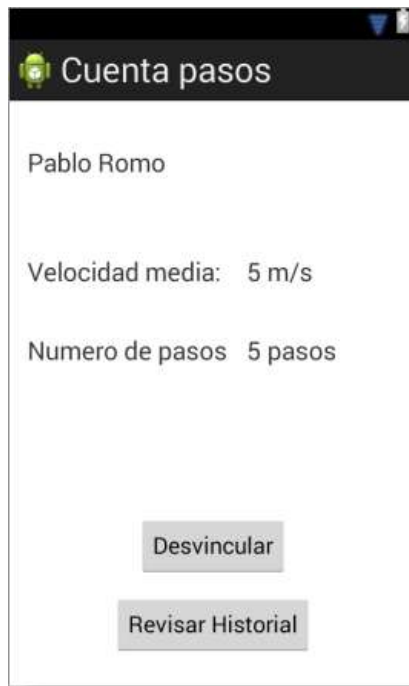


Figura 11. Interfaz principal del entorno teléfono

Ambos parámetros son administrados por el servicio de detección de pasos. Este servicio tiene como tareas: obtener la información de la cantidad de pasos desde el algoritmo de detección y enviar las actualizaciones a la base de datos. El servicio es lanzado inmediatamente al ingresar a esta sección de la aplicación, y cada vez que se ingresa se vincula nuevamente a esta vista. Esta vinculación es necesaria porque el servicio se ejecuta en paralelo a la aplicación, lo que permite que el servicio funcione a cada momento aunque la aplicación sea terminada por el sistema operativo.

Las otras acciones disponibles en la vista son: acceder al historial de actividades y desvincular el dispositivo. La desvinculación de dispositivo es importante, aunque se ejecuta de manera simple. Primero se borran los datos de sesión, lo que permite a otro usuario vincular este dispositivo y además también registrarse usando este dispositivo. Como segunda parte de la desvinculación se termina con el servicio de registro de actividades. Por último, se redirige al usuario a la vista de pantalla de inicio/Login.

4.2.3.5 Historial de Actividades y Actividad Diaria

Se despliegan las interfaces de historial de actividades y actividad diaria, además de una explicación de su funcionalidad e implementación.



Figura 12. Historial de actividades

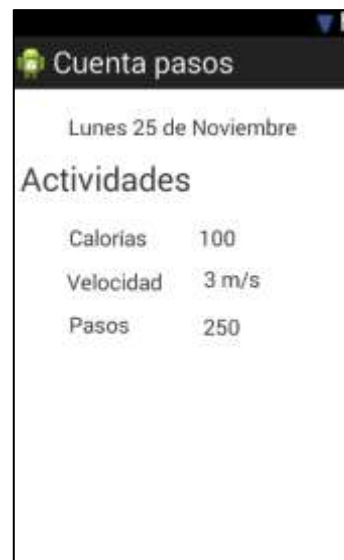


Figura 13. Detalle de actividades

El historial de actividades, muestra las fechas donde se ha registrado actividades. Estas fechas se presentan como una lista. Un punto interesante es la carga asincrónica de la información de esta lista. Como se ha presentado en distintas ocasiones en este documento, la información se obtiene a través de request HTML al servidor REST. Estas peticiones al igual que las anteriores son de tipo asincrónicas, pero en este caso es mucho más importante que sean así debido al volumen de información. La consulta asincrónica permite que, aunque el volumen de información sea grande, el usuario no vea la aplicación congelada durante su carga. Como se mencionó anteriormente, se utiliza la clase AsyncTask que nos permite dejar de lado el manejo de threads, se encarga de ejecutar la consulta y la espera de resultados sin interferir con el hilo de ejecución de la interfaz. En caso de no tener conexión disponible, se carga simplemente con los datos de la base de datos local. Los datos obtenidos a través de internet, son cargados inmediatamente a la base de datos local para su posterior lectura sin conexión.

Una vez seleccionado la fecha para la cual se quiere acceder al resumen de actividades, se ingresa a la interfaz de la figura 13. Se despliega la cantidad de pasos realizados, las calorías y la velocidad media. Estos datos son consultados a través del request HTML del tipo GET, y si la conexión no está disponible, se verifica si la información se encuentra disponible en la base de datos local. Se le notifica al usuario de esta carga local, para que esté al tanto que es posible que la información no esté actualizada.

Ambas interfaces se pueden acceder desde distintos puntos de la aplicación, como se puede ver en el mapa de la figura 4, pero la funcionalidad es la misma. Lo único que cambia es el usuario y el dispositivo asociado con el cual se consulta la información.

4.2.3.6 Pantalla de Bienvenida

La pantalla de bienvenida es la siguiente:



Figura 14. Interfaz de bienvenida



Figura 15. Interfaz de bienvenida alternativa

Esta interfaz, permite al usuario seleccionar entre los dispositivos que monitorea o ver los datos del dispositivo actual. Si selecciona el dispositivo actual y está utilizándolo para su uso personal, se despliega la vista principal, si no es así, se despliega el resumen del dispositivo. En caso de que el usuario que ingresó a la aplicación no es el usuario el cuál monitorea el dispositivo, será invisible el botón “este dispositivo”, como se puede ver en la figura 15. Simplemente se verifica que el usuario sea el registrado en los datos de sesión local.

4.2.3.7 Lista de Dispositivos y Resumen del Dispositivo

Las interfaces de la lista de dispositivos y el resumen del dispositivo, se presentan a continuación en conjunto con la descripción de su funcionalidad.

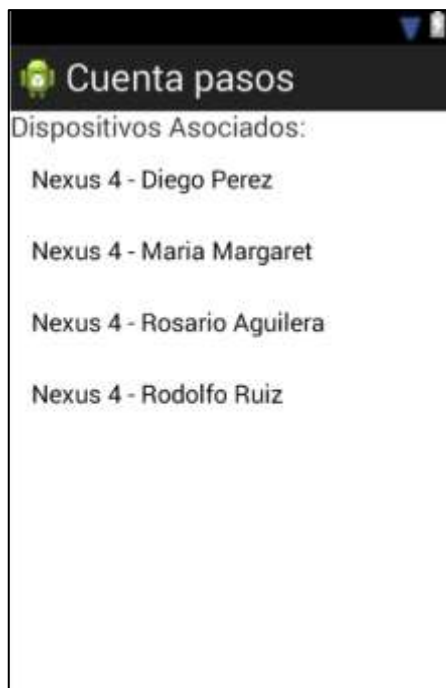


Figura 16. Lista de dispositivos asociados



Figura 17. Resumen del dispositivo

La lista de dispositivos asociados, se obtiene a través del método GET de HTTP al servidor web REST para así obtener los datos más actuales, en caso de que la conexión no esté disponible, se presenta la información guardada en la base de datos local. La lista presenta el nombre para cada uno de los dispositivos, que por defecto sería el modelo del dispositivo y el nombre del usuario. Una vez escogido un dispositivo se presenta un resumen de los datos del dispositivo.

El resumen del dispositivo despliega: la cantidad total de pasos y calorías, además del nombre del usuario y la opción de ir al historial de actividades. La cantidad total de pasos y calorías contadas, se consultan directamente al servidor web con una URI determinada dejando que la sumatoria la calcule en forma eficiente el servidor, evitando traer toda la información.

4.3. Entorno de la Aplicación Web

La aplicación web, fue desarrollada utilizando Symfony 2.0, un framework PHP basado en el patrón de diseño Modelo Vista Controlador. Symfony se caracteriza por ser fácil de instalar, configurar y a la vez, permitir desarrollos rápidos sin perder robustez. Además, es independiente del gestor de bases de datos. Se utilizó un servidor Apache 2.4, con una versión de PHP 5.3. La base de datos compartida es gestionada por MySQL 5.5.32. La conexión con la base de datos, es directa a través de Doctrine que provee servicios de persistencia y permite abstraerse a través de ORM (Object Relational Mapper), para beneficiar la orientación a objetos.

En términos de implementación de la interfaz, Symfony recomienda el uso de Twig, un engine de templates para PHP. Twig presenta una sintaxis concisa y sencilla de aprender, además de que provee seguridad y es desarrollado por los creadores de Symfony.

El objetivo del entorno de aplicación web es bastante específico: Permitir a los usuarios monitorear la actividad física de los dispositivos asociados. Considerando esto, podemos inferir que como mínimo esta aplicación debe permitir al usuario ingresar con su cuenta, verificar los dispositivos asociados a él y ver los datos entregados por el dispositivo móvil. También se pueden añadir algunas funciones adicionales para darle completitud al sistema, como por ejemplo: cambio de contraseña, editar los datos de usuario y obtener ayuda de la aplicación.

4.3.1. Aspectos Relevantes del Desarrollo

En términos de magnitud, el desarrollo de la aplicación web es muy inferior al desarrollo de la aplicación móvil, pero aun así, hay un punto bastante interesante que es el manejo de la seguridad en la aplicación. Si bien la seguridad en este tipo de desarrollos no es lo más importante, se debe tener en cuenta al menos en cosas sencillas y fundamentales, por ejemplo: ¿Cómo se puede impedir que un usuario obtenga acceso a información que no le corresponde? Una respuesta algo obvia, es pedirle que ingrese su usuario y contraseña cada vez necesite ingresar a una sección de información personal. Esto claramente es algo frustrante y no parece para nada ideal. Entonces, ¿Cómo se implementó la seguridad en esta aplicación? Claramente hay muchas formas de hacer esta implementación, pero se buscó la forma más sencilla, y a la vez robusta de hacerlo. En este sentido, se siguieron los patrones de diseño de Symfony, el cuál divide el aspecto de la seguridad en dos procesos: Autenticación y Autorización.

La Autenticación, es el proceso donde el usuario verifica que es quien dice ser. Acá la implementación es bastante sencilla, basta proveer un formulario de Login y configurar la autenticación. Symfony actúa como el firewall, activándose cada vez que el usuario hace un request y determina si es necesario que use el Login para autenticarse. Por otra parte, la definición del firewall es bastante simple, se define cuál es el área segura y el área de Login en un archivo “security.yml”. Un punto relevante de mencionar, es que el punto de Login no puede ser bloqueado por el firewall, si no, nadie podría ingresar en la aplicación. Es por esto, que se utiliza el usuario anónimo para permitir que accedan a la ruta del Login y que todo el resto de la aplicación esté protegida por este firewall. Esto permite que un usuario cualquiera llegue a la aplicación, y sin importar a que ruta se dirija, sea redirigido al Login para hacer su respectiva autenticación.

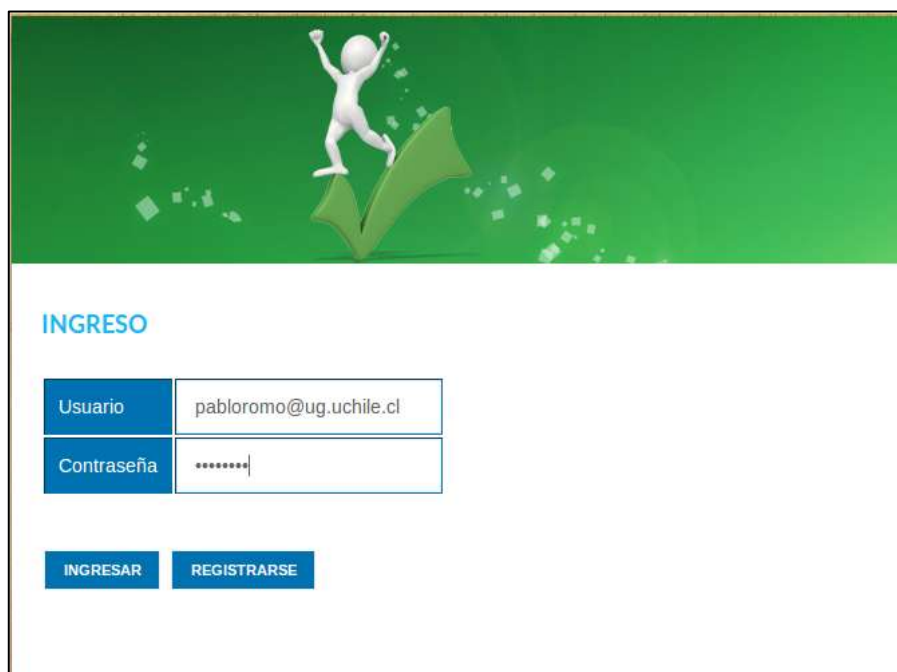
La Autorización, es el proceso que decide si el usuario tiene acceso a un determinado recurso. Aquí entran en juego los roles de usuario que se pueden observar en el diagrama de la base de datos de la figura 3. Similarmente a la autenticación, la definición de las rutas y los controles de acceso es bastante simple. Se define en el mismo archivo “security.yml” los permisos de los roles

en las rutas, permitiéndole a Symfony determinar en base a estas reglas, los accesos de un determinado rol de usuario a los recursos. En este desarrollo, sólo se utiliza el usuario normal y el usuario anónimo, pero a futuro se podrían implementar usuarios administradores u otros roles.

A modo de resumen, tanto la autenticación y autorización son procesos muy importantes en términos de seguridad, y Symfony provee la administración de ambos procesos de una manera sencilla y fácil de configurar. Mientras que por otra parte, se podría verificar en cada ruta que el usuario esté autenticado, verificar su rol y redirigirlo al Login en caso de no tener acceso al recurso. Esto claramente implicaría una cantidad inmensa de código duplicado, y frente a la propuesta de Symfony en términos de seguridad, no se hace complejo escoger esta última.

4.3.2. Detalle de Interfaces y Funcionalidades

Se presentarán las interfaces y se explicarán los detalles de funcionalidad relevantes en el desarrollo de esta aplicación web. Primero que todo, al ingresar a la aplicación, se ingresa a la sección de Login:



INGRESO	
Usuario	pabloromo@ug.uchile.cl
Contraseña	*****
INGRESAR	REGISTRARSE

Figura 18. Ingreso aplicación web de la solución

Para permitir que el usuario ingrese a la aplicación, se debe definir la entidad que representa a un usuario, en nuestro caso será la tabla de usuarios: User. Las entidades, no son más que la relación a objetos de las tablas de la base de datos. Definida esta entidad de usuario, se le indica a Symfony que propiedad representa al nombre de usuario (en nuestro caso el email). Con esto, el usuario ya es capaz de pasar el firewall e ingresar a la aplicación y a su información.

Una vez que el usuario ingrese a la aplicación, se desplegará una sencilla pantalla de bienvenida y se le presentarán al usuario las distintas funciones disponibles (ver Fig. 19).



Figura 19. Pantalla bienvenida aplicación web

Un aspecto interesante, es el uso de templates de TWIG. En este caso por ejemplo, necesitamos que el banner y el menú aparezcan en cada una de las secciones de la aplicación. Si se utilizara HTML básico, habría que copiar y pegar cada vez el código del menú para cada una de las vistas. Usando TWIG es diferente, pues se debe definir un template base y dentro de ese template base delimito a través de tags las secciones en donde insertaré el contenido, por ejemplo, en este caso todo el espacio donde escribo la bienvenida. Una vez definido ese template, toda vista que desee utilizarlo simplemente se llama a un “extends” del template base para incrustar el código empleando los tags correspondientes. Esto hace bastante sencillo el proceso de definir las vistas, permitiendo sólo preocuparse del contenido.

Como se mencionaba en los objetivos, la sección más importante de esta aplicación es permitir al usuario verificar los datos de los dispositivos monitoreados.

En seguida, se presenta la vista de “Mis dispositivos” (ver Fig. 20).



Figura 20. Lista de dispositivos en aplicación web de la solución

Antes que todo, el usuario debe escoger el dispositivo del cual visualizará la información. Este proceso es bastante simple, no es más que un formulario que a través de POST envía el id del dispositivo escogido. Podemos observar, que se despliegan los dispositivos con su respectivo nombre que incluye el modelo del teléfono y el nombre del usuario monitoreado.

Una vez escogido el dispositivo, se despliega la cantidad de pasos realizadas por el usuario en los últimos 30 días:

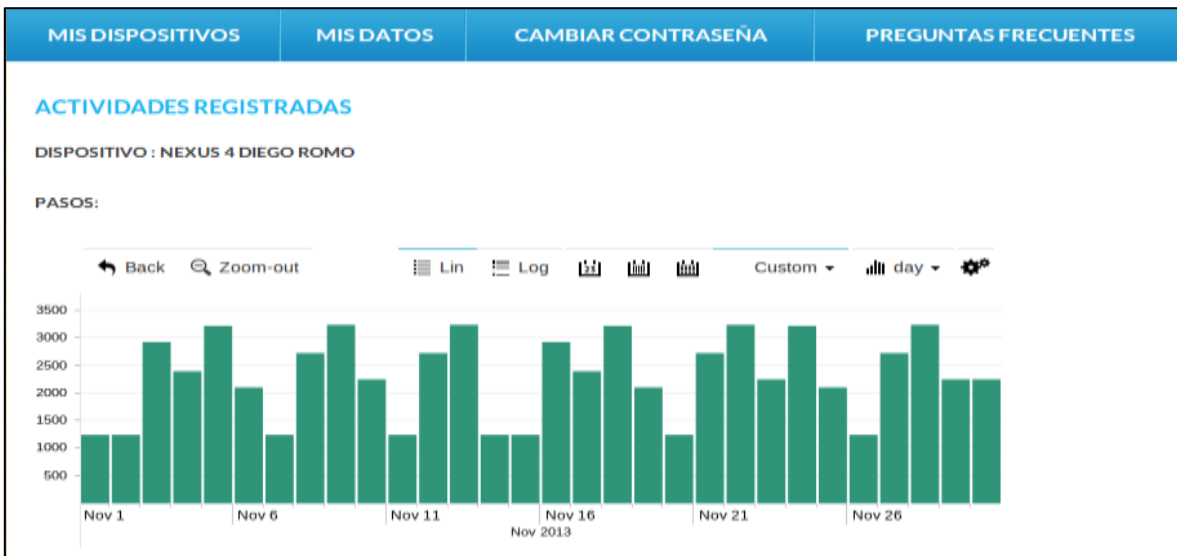


Figura 21. Actividades registradas en visualización por día

Así también se pueden cambiar las opciones de visualización, por ejemplo, para mostrar los últimos meses:

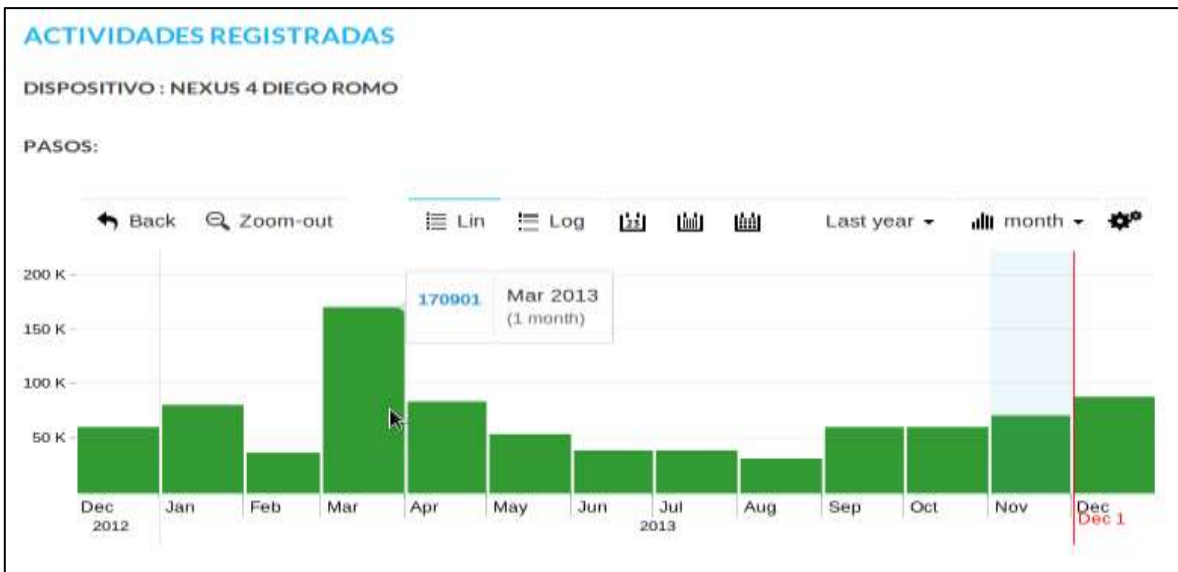


Figura 22. Actividades registradas en visualización mensual

También se pueden mostrar los últimos años:

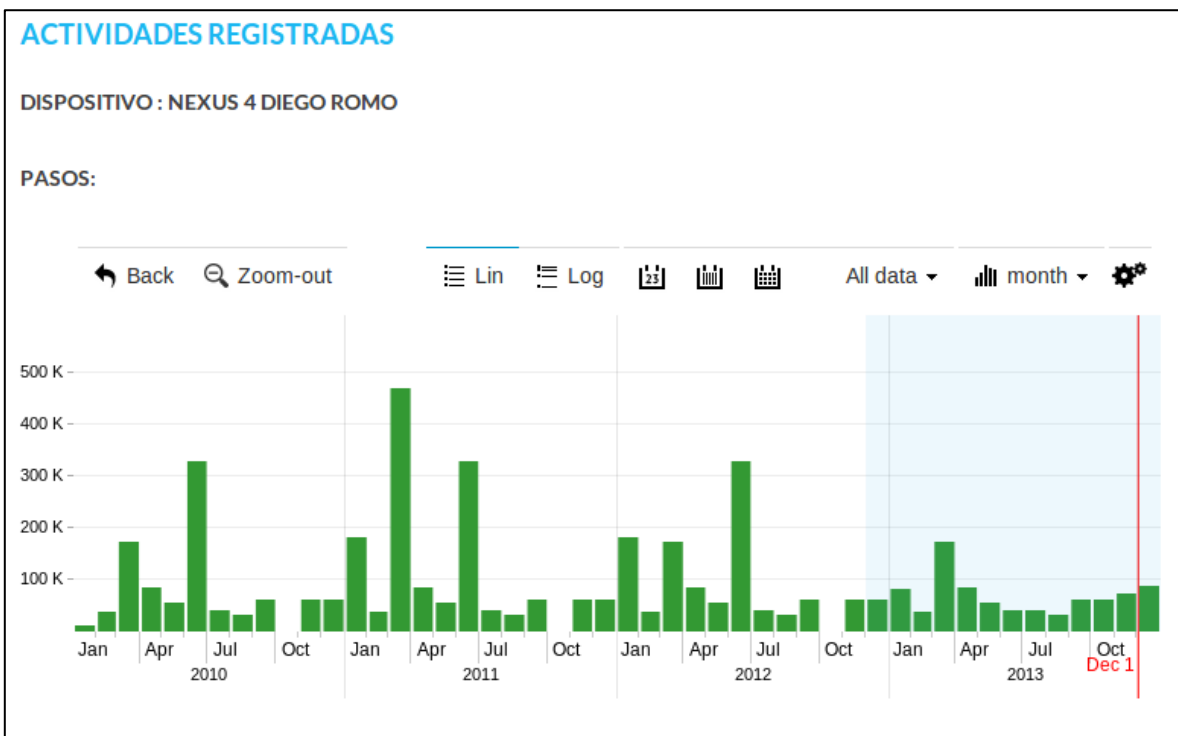


Figura 23. Actividades registradas en visualización anual

El proceso de despliegue de los gráficos, se realiza en varias partes. En primera instancia, se debe consultar a la base de datos por todas las actividades de la tabla Activity que tengan el mismo id que el dispositivo seleccionado. Esta consulta se realiza a través del EntityManager de Symfony, que permite abstraerse de la consulta SQL. Las actividades se entregan en forma de arreglo a la

vista. Como no es suficiente desplegar una lista gigante con los datos de todas las actividades recopiladas, se utiliza una librería de HTML5 llamada TimeChart. La función de la vista, es iterar sobre los elementos entregados para crear los valores a desplegar en el gráfico. Acá TWIG entra en juego, gracias a la capacidad de definir ciclos dentro de las vistas, así se genera un ciclo que recorre el arreglo y va llenando con los datos de las actividades a la librería. Del resto de las visualizaciones se encarga la librería TimeChart, que permite modificar el modo de vista de los datos a través del panel de control desplegado en el gráfico. Así tenemos una visualización intuitiva de la información requerida por el usuario y sencilla de implementar. De esta misma forma, se despliega la cantidad de calorías utilizadas por el usuario, el único cambio es que en la inyección de datos para el gráfico se cambia el dato de la cantidad de pasos por el de la cantidad de calorías. Por último, tenemos las vistas para la edición de datos:

MIS DISPOSITIVOS	MIS DATOS	CAMBIAR CONTRASEÑA	PREGUNTAS FRECUENTES	SALIR
EDITAR MIS DATOS				
Nombre:	Pablo			
Apellido:	Romo			
Edad:	25			
Altura:	177			
CONFIRMAR				

Figura 24. Edición de datos personales en interfaz web

MIS DISPOSITIVOS	MIS DATOS	CAMBIAR CONTRASEÑA	PREGUNTAS FRECUENTES	SALIR
CAMBIAR CONTRASEÑA				
Contraseña actual:				
Nueva Contraseña				
Repetir Nueva Contraseña				
CONFIRMAR				

Figura 25. Cambio de contraseña interfaz web

Ambas interfaces son bastante sencillas, y permiten al usuario modificar sus datos o su contraseña. En el caso de la información del usuario, se obtiene a través de los datos de sesión que tiene Symfony. De estos, simplemente se llama a los métodos definidos por la entidad usuario para acceder a su nombre, apellido, edad y altura. Estos datos son entregados a la vista para ser desplegados. Para el caso del cambio de contraseña, es simplemente un formulario POST, que comprueba que la contraseña entregada sea la que actualmente posee el usuario y que ambas contraseñas nuevas coincidan, después a través del EntityManager (que se encarga de la persistencia de datos) se escribe en la base de datos la nueva contraseña.

4.4. Entorno de Servidor Web

Las posibilidades para hacer un servidor web son innumerables. El primer punto relevante, es identificar en qué situaciones y qué tipo de comunicación se establecerá con el servidor. En este caso particular, toda la información proviene desde un dispositivo móvil que probablemente tenga una conexión itinerante. Considerando este último punto, una muy buena opción sería un servicio web muy liviano y fácil de manipular. Aquí entra en juego la API REST, que utiliza HTTP como protocolo de comunicación basándose en sus métodos POST, GET, PUT y DELETE para realizar las inserciones, consultas, actualizaciones o borrados de datos en la base de datos. A nivel de dispositivo móvil es fácil de consultar, ya que, simplemente se realizan request HTTP y los datos que son recibidos en forma de JSON o XML que cuentan con muchas librerías que permiten su fácil consumo, además se usa en muchas aplicaciones como Twitter, Flickr, entre otras.

Tomando en cuenta que ya tenemos decidida la API que implementará el servidor web, lo siguiente es elegir el lenguaje de programación y/o Framework en que se construirá esta API. Las posibilidades son muchas, pero considerando que el modelo de datos es simple, se podría considerar alguna forma de construir la API automáticamente a partir del modelo de datos. Este enfoque puede ser mucho más rápido que crear completamente la API en cualquier Framework, pero tiene como desventaja la falta de personalización de la misma. A partir de la investigación, se decidió considerar las herramientas entregadas por Netbeans, usando J2EE v7, JAX-RS (Java API for RESTful Web Services), Jersey y un servidor GlassFish versión 4.0.

El proceso de creación de la API utilizando Netbeans es sencillo. Primero que todo, hay algunos pasos de configuración para después hacer la conexión a la base de datos y se crean los siguientes elementos:

- *Entity Classes*: Estas clases típicamente representan una tabla de una base de datos relacional. Cada instancia de una entidad representa una fila de esa tabla.

- *Persistence Unit*: Consiste de un set de Entity Classes, el proveedor de la información, el proveedor de persistencia, y el nombre de la unidad de persistencia, como se define en el archivo XML que determina esta unidad.

¿Qué papel juega JAX-RS y Jersey en todo esto? JAX-RS no es más que la especificación de Java para servicios REST, y Jersey es la implementación de esta especificación. La especificación usa anotaciones para definir la relevancia en REST de las clases de JAVA, por otra parte, Jersey contiene básicamente un servidor REST y un cliente REST. Jersey se encarga de escanear las clases predefinidas para identificar los recursos REST, así también dispone de un servlet encargado de analizar el HTTP entrante y seleccionar la clase correcta y método para responder a esta consulta. Acá entran en juego las anotaciones que son prioritarias para escoger el método y las entidades, que son los objetos que se codifican para ser enviados como respuesta. Por último JPA (Java Persistence API), es el encargado de establecer la comunicación con la base de datos.

En concreto, una vez finalizado el proceso se obtienen las siguientes URI con su respectivo método HTTP (considerando una entidad llamada usuario):

- *URI*: http://localhost/usuario
 - *Método*: POST - Recibe a la entidad en forma de XML o JSON y la inserta en la base de datos.
 - *Método*: GET – Retorna la lista completa de usuarios en la base de datos en forma de XML o JSON.
- *URI*: http://localhost/usuario/{id}
 - *Método*: PUT –Recibe el id del usuario como su definición en XML/JSON y actualiza estos datos en la base de datos.
 - *Método*: DELETE –Se elimina de la base de datos al usuario del id indicado.
 - *Método*: GET – Retorna al usuario con el id correspondiente en forma de XML o JSON.
- *URI*: http://localhost/entity/{from}/{to}
 - *Método*: GET – Retorna todos los usuarios con id en el rango {from, to}.
- *URI*: http://localhost/entity/count
 - *Método*: GET – Retorna el total de usuarios de la base de datos.

Además se obtiene una interfaz de pruebas de la API:

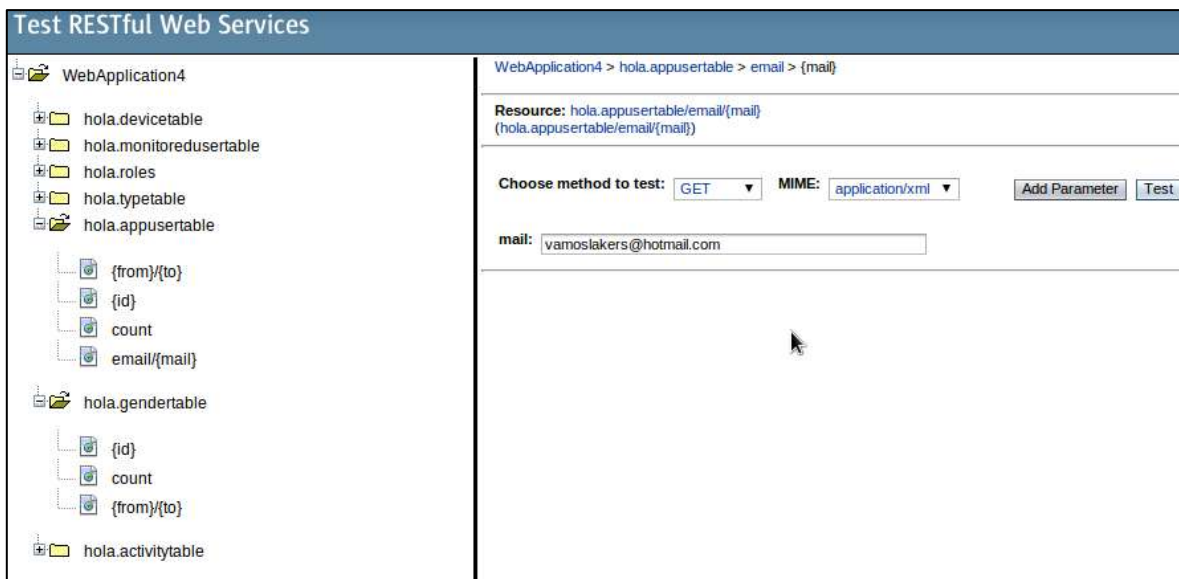


Figura 26. Interfaz de pruebas de API REST del servidor web

Como se puede observar, tenemos cada uno de los métodos disponibles y podemos obtener los datos en formatos correspondientes al tipo de información entregada, usualmente XML o JSON. Sin embargo, hay consultas (como el número de usuarios) cuyo resultado se entrega directamente sin un formato.

A grandes rasgos tenemos la base del servidor construida, pero hay varios métodos que serán muy útiles al momento de comunicarse con el servidor y no están definidos. Por ejemplo, si quisiera saber si un determinado email está siendo utilizado por un usuario, la única forma disponible, es pedir todos los usuarios de la base de datos y después comprobar uno a uno para ver si el email ha sido utilizado. Claramente esta forma no es óptima para realizar las consultas, y por lo tanto debiese ser el servidor el que se encargue de hacer ese trabajo, aprovechando las optimizaciones de la base de datos.

La definición de un método consta de dos partes. La primera, es que se debe crear la consulta a la base de datos. Esta consulta SQL se puede definir en el código del Entity y para esto se utilizan las Named Queries de JPA. Una Named Query no es más que una consulta definida por un string que no se puede modificar. Esto obliga a usar los parámetros de la consulta, lo que resulta en consultas más eficientes. Como ejemplo, podemos considerar la siguiente Named Query:

```
@NamedQuery(name="AppUserTable.findByEmail", query = "SELECT a FROM AppUserTable a WHERE a.email = :email")
```

Una vez definida la consulta a la base de datos, necesitamos definir las anotaciones que le permitirán a Jersey identificar el método correspondiente, definir la URI que identificará al recurso, definir el formato de los datos a consumir y el tipo de respuesta:

```
@GET
@Path("email/{mail}")
@Consumes({"application/xml", "application/json"})
@Produces({"application/xml", "application/json"})
public String getEmail(@PathParam("mail") String mail){ /* Implementación método */ }
```

Como podemos notar las anotaciones son sencillas de entender. Primero, se define que el tipo de método HTTP a utilizar, en este caso GET. Segundo, se aceptan datos del tipo JSON/XML y el retorno también son datos de tipo JSON/XML. También es muy importante la definición de la URI, que permite identificar al recurso y los datos que se reciben que funcionan como parámetros de la consulta. El uso de la Named Query en el método, nos permite abstraernos del código SQL en esta capa de definición de métodos.

5. Pruebas del Prototipo

En las secciones anteriores se explicitaron las distintas funcionalidades, pero hay dos puntos que son interesantes de estudiar: la precisión de la detección de pasos y el rendimiento de batería con el uso del monitoreo.

5.1. Precisión

Se busca comparar la precisión de la aplicación en el conteo de pasos, con otras aplicaciones presentes en el mercado. Esta sección presentará el experimento realizado, los resultados y una pequeña discusión sobre los mismos.

5.1.1 Experimento

Se busca probar la precisión de la aplicación para el conteo de pasos. Los factores que inciden en la precisión son variados. Por ejemplo, la posición del teléfono dentro de los pantalones es importante, también afecta el caso de que esté completamente suelto en el bolsillo. El experimento a realizar es el siguiente:

- Una persona con la misma tenida de ropa y el teléfono en su bolsillo del pantalón.
- Se realizan 10 caminatas de 10 pasos cada una.
- Las caminatas se realizan con 5 velocidades distintas: muy lento, lento, normal, rápido, muy rápido.
- Para cada velocidad se realizan dos caminatas de 10 pasos.
- Por cada caminata a una velocidad determinada, se anota la cantidad de pasos que se registraron en la aplicación.
- En el caso de la aplicación desarrollada, se realiza dos veces los experimentos. Uno sin la etapa de calibración inicial y el segundo con la calibración inicial realizada de manera normal.

Este experimento se realizó con todas las aplicaciones de la Tabla 1 y se comparan con los datos registrados con la aplicación desarrollada usando la calibración y sin el uso de calibración.

5.1.2. Resultados

Se presentan los resultados arrojados por el experimento para la velocidad lenta, normal y rápida. Se espera que las velocidades promedio de los usuarios estén cerca de estas tres velocidades por lo que, para el análisis sólo se considerarán estas tres velocidades. El resto de los resultados, se

presentan en los Anexos sección 8.2.1. A continuación, se exhibe una tabla con el nombre de cada aplicación y la letra asignada para una mejor visualización de los gráficos:

Tabla 2. Letras asignadas para despliegue de resultados en prueba de precisión

Nombre Aplicación	Letra Asignada
Runtastic Pedometer	A
Podómetro - Accupedo	B
Smart Pedometer	C
Noom Walk: podómetro	D
Me paseo podómetro	E
All-in Podómetro	F
Podómetro Plus	G
Beppi's Podómetro fácil	H
WalkLogger podómetro	I
New Pedometer Free	J
Endomondo	K
Aplicación Memoria con Calibración	L
Aplicación Memoria sin Calibración	M

El siguiente gráfico, presenta el error total de pasos registrados en las 3 velocidades (lento, normal, rápido) con sus 2 caminatas respectivas. El error se calcula tomando el valor absoluto de la diferencia entre los pasos esperados y los pasos registrados. Los pasos esperados son siempre 10 para cada caminata.

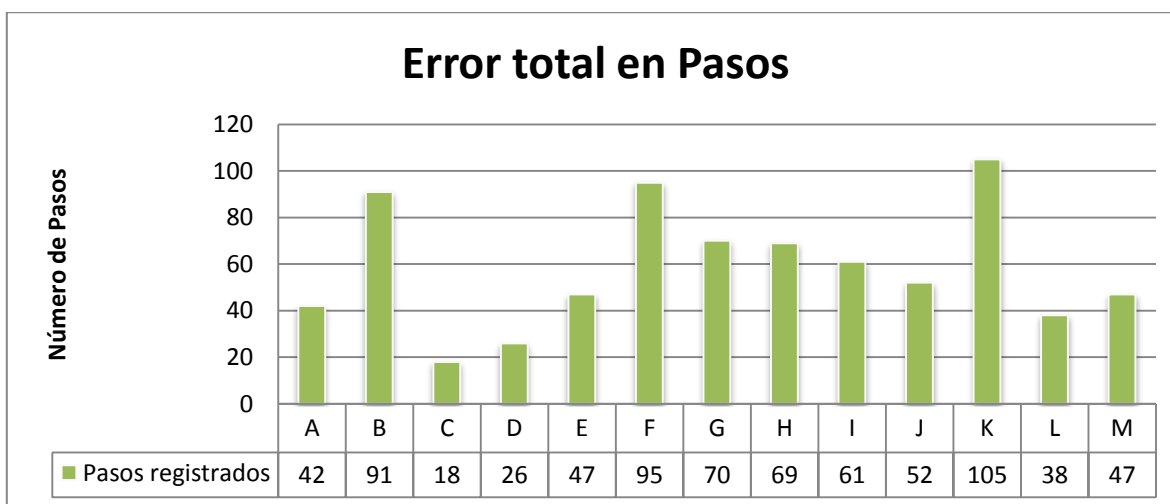


Gráfico 1. Errores totales en conteo de pasos para el experimento 1

Así también, se presenta el porcentaje de error con respecto a la cantidad total de pasos esperada:

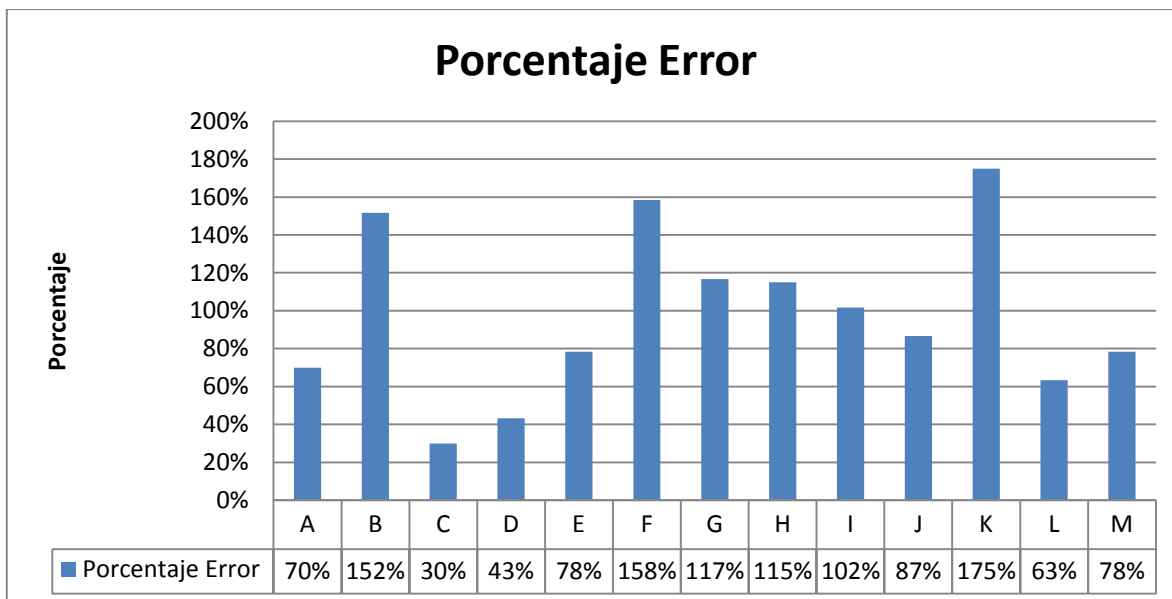


Gráfico 2. Porcentaje de error para el conteo de pasos en Experimento 1

Por último, se presentan los pasos registrados por cada una de las caminatas para la aplicación desarrollada.

Tabla 3. Pasos registrados por la aplicación desarrollada para prueba de precisión

Aplicación	Lento	Lento 2	Normal	Normal 2	Rápido	Rápido 2
L	17	6	17	22	14	14
M	10	11	17	6	21	34

5.1.2. Discusión

Podemos observar que los porcentajes de error son altos en todas las aplicaciones, el mínimo error obtenido es de un 30%. La aplicación desarrollada presenta una tasa de error del 63% que puede parecer bastante alta si se considera por sí sola, pero hay que los algoritmos de detección son bastante imprecisos, prueba de esto es que la mejor aplicación del mercado alcanza un 30% de error en la detección de pasos y este tampoco es un valor muy bajo con respecto a lo uno esperaría de una aplicación desarrollada comercialmente con 2 o 3 años en el mercado. Además de entre todas las aplicaciones probadas en el experimento la aplicación calibrada se ubica en el tercer lugar de entre las aplicaciones con menor tasa de error.

Por otra parte, si consideramos los datos de la tabla 3, tenemos que la aplicación en sólo dos ocasiones detecta menos que la cantidad de pasos realizados por el usuario. Esto es bastante importante porque siempre se debe contar al menos la cantidad de pasos que hace el usuario, ya que, sería muy frustrante el haber realizado una gran cantidad de pasos y que estos no sean contabilizados. Además de estos resultados podemos inferir que se debe amentar el umbral de

detección de los pasos. El umbral de detección es el que permite decidir al algoritmo de detección de pasos si un movimiento representa a un paso o no. Si el umbral es alto, se necesitan movimientos de pasos más bruscos para que sean contados como pasos. Entonces, como la aplicación está contando más pasos de los que debiese contar, se podría aumentar el umbral de detección y así descartar pasos que en realidad no son.

5.2. Consumo de Batería

Otro de los aspectos trascendentes para el funcionamiento de la aplicación desarrollada es el gasto de batería durante el uso diario. Si la aplicación consume grandes cantidades de batería nos veremos con un dispositivo sin la capacidad de monitorear por no tener carga suficiente. A continuación se presenta el experimento realizado, sus resultados y una discusión de estos resultados.

5.2.1. Experimento

El experimento realizado es bastante sencillo:

- Teléfono con su carga en 100%.
- Se inicia la aplicación y se anota la hora en que se inicia esta aplicación.
- Se deja el teléfono en stand-by por 6 horas.
- Al cabo de las seis horas se verifica en el mismo teléfono la cantidad de batería utilizada por el celular. Para este fin Android provee de una interfaz que despliega la cantidad de batería utilizada por las aplicaciones. Se anota el porcentaje de batería restante en el teléfono y el porcentaje de batería utilizado por la aplicación.

Un punto importante a mencionar es que los porcentajes de gasto de batería mostrados por Android, son relativos a la influencia en el gasto de batería de la aplicación, considerando la cantidad restante de batería. Por esta razón el teléfono debe quedar en stand-by durante el proceso de pruebas.

Otro punto relevante es que Android no despliega el uso de acelerómetro como gasto de batería de una aplicación específica, sino que se muestra en una categoría separada con su propio porcentaje. Considerando este punto se considera el gasto de batería de la aplicación como el gasto de batería de la aplicación y el gasto de batería de la categoría de sensor acelerómetro.

5.2.2. Resultados

El experimento se realizó en tres ocasiones. Primero se presentan los totales de batería utilizada por el acelerómetro y la aplicación.

Tabla 4. Consumo de batería por componentes de aplicación

Experimento	Acelerómetro	Batería Aplicación	Total
Experimento 1	8%	4%	12%
Experimento 2	6%	5%	11%
Experimento 3	5%	2%	7%

También se presenta el porcentaje de batería final después de las 6 horas del experimento y el porcentaje total utilizado por la aplicación.

Tabla 5. Impacto del consumo de batería de la aplicación en relación al consumo total

Experimento	Batería Inicial	Batería Final	Batería Aplicación
Experimento 1	100%	80%	12%
Experimento 2	99%	83%	11%
Experimento 3	100%	77%	7%

5.2.3. Discusión

El consumo de batería hoy en día es uno de los grandes problemas de los teléfonos inteligentes. A modo de ejemplo, podemos la duración de batería de distintos dispositivos en el caso de la navegación por internet:

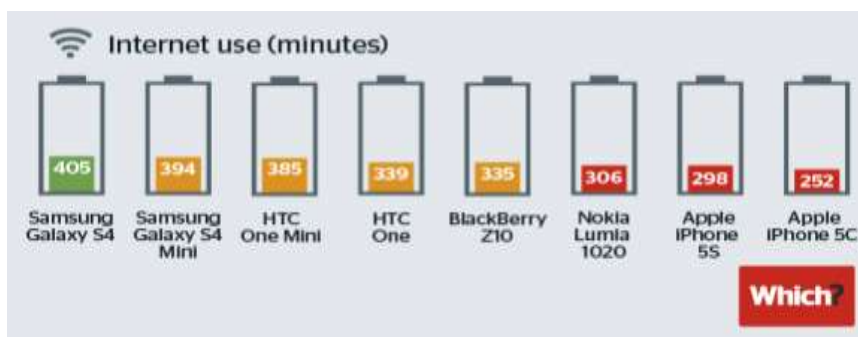


Figura 27. Consumo de batería utilizando internet en diversos teléfonos

Podemos observar que un teléfono como iPhone 5C consume completamente su batería en 4 horas si el usuario se dedica a navegar por internet. Por su parte, nuestra aplicación en un periodo de 6 horas consume el 10% del total de batería que se consume con el teléfono en stand-by con la aplicación activa. Esto quiere decir con la aplicación activa y el teléfono en stand-by se consume

un total de un 20% de batería durante 6 horas, en esto la aplicación desarrollada aporta un gasto de batería de un 10% de ese 20% consumido, por lo que, solamente aporta al gasto de un 2% de la batería total del teléfono en ese periodo de 6 horas. Esto en parte es gracias a la decisión de hacer el envío de información cada una hora lo que permite que el gasto de batería por parte de la aplicación sea mínimo y en gran parte su gasto se asocie al uso del acelerómetro. Aunque es claro que este 10% de gasto de batería puede resultar crítico para usuarios que utilizan constantemente para todas las tareas disponibles en un teléfono inteligente: navegación, videojuegos, música, entre otros. Un heavy-user carga su teléfono más de una vez durante el día (Ferreira 2011), mientras que en el caso de usuarios que usualmente no hacen un uso tan constante de su teléfono inteligente el porcentaje de batería utilizado por la aplicación es aceptable para su uso normal.

6. Conclusiones y Trabajo Futuro

La aplicación desarrollada en el marco de esta memoria, en términos generales, asume el mismo esquema de trabajo que la mayoría de las aplicaciones móviles que cuentan con una interfaz Web. Por lo tanto, tenemos una aplicación móvil que proporciona información, la cual queremos acceder desde una aplicación Web usando un navegador. Esto obliga a que exista un canal de información compartido, o sea, un servidor web que los pueda comunicar ambas aplicaciones a través de los datos.

Esta estrategia de funcionamiento es muy común hoy en día, y además es casi impensado tener una plataforma web que no tenga una contraparte móvil y viceversa. Por ejemplo, tenemos redes sociales como *Instagram* que nacieron simplemente como una plataforma para dispositivos móviles y que luego pasaron a una versión Web. Considerando esto, este esquema de funcionamiento es muy interesante como proceso general, y brinda conocimiento útil para otros proyectos futuros.

Ahora en términos específicos, el objetivo general de este trabajo de memoria fue recopilar la información de la actividad física de la persona usando un teléfono inteligente, y además permitir a los usuarios monitorear a otras personas a través de una plataforma web. En este sentido, se proporciona una solución que utiliza el acelerómetro del teléfono inteligente para determinar, de manera aproximada, la cantidad de pasos realizados por el usuario durante el día. En base a esa información la aplicación estima la cantidad de calorías quemadas durante el día. Además se proporciona un entorno Web donde los usuarios pueden acceder a esta información y monitorear a sus usuarios asociados.

Es importante mencionar que la solución, de acuerdo a los experimentos exhibidos, permite un uso normal del teléfono con respecto al consumo de batería durante el día, factor clave para la recopilación de información. O sea, en términos generales se cumplió el objetivo del trabajo planteado. Claramente hay muchos detalles específicos importantes, pero es muy relevante que se haya cumplido el objetivo propuesto.

Un factor que no se puede dejar de lado es el nivel de error en la cantidad de pasos contados. En este sentido la aplicación cumple con ser un indicador de la cantidad de actividad física, pero no es un contador preciso de la misma. Este es un punto que debe ser mejorado en trabajo futuro haciendo mejoras al algoritmo utilizado o utilizar un dispositivo de conteo externo que nos asegure esta precisión. Es importante mencionar que el sistema de detección de pasos está aislado y puede ser modificado para generar estas mejoras.

Indagando más específicamente en las metas logradas, es muy difícil que algún desarrollo cumpla a cabalidad cada una de las ideas surgidas en el mismo. Además hay muchas ideas interesantes que incluso aparecen al momento de escribir este documento de memoria. Considerando esto se presentarán algunas ideas para un trabajo futuro:

- Probar y utilizar la API de Google ActivityRecognition para determinar los distintos tipos de actividades. Esta es una de las mayores tareas pendientes que no influye directamente en el objetivo principal, pero es un interesante nicho de estudio.
- Mejorar la interfaz de todas las aplicaciones y realizar pruebas de usabilidad de las mismas.
- Realizar pruebas de la aplicación a lo largo de al menos un mes.
- Integrar las redes sociales y la posibilidad de compartir la información recopilada. Puede ser un factor importante para motivar a los usuarios a superarse en sus actividades físicas el hecho de compartir los datos registrados por el dispositivo.
- Mejorar la precisión del conteo de pasos y buscar formas de que el gasto de batería sea menor aún a lo expuesto en las pruebas preliminares. En este sentido, el último dispositivo creado por LG para Google integra un contador de pasos a nivel de hardware. Así se busca minimizar el gasto de batería para el monitoreo de actividades físicas, por tanto, sería muy interesante integrar esta aplicación en ese modelo de teléfono inteligente, para ver las mejoras en precisión y uso de batería.

7. Bibliografía y Referencias

- Alcalá, 2010.** <http://www.ind.cl>. *Instituto Nacional de Deportes*. [En línea] 2010. [Citado el: 12 de 09 de 2012.] http://www.ind.cl/estudios-e-investigacion/investigaciones/Documents/encuesta_nacional_habitos.pdf.
- Ferreira, D., Anind, D., Kostakos, V. 2011.** *Understanding human-smartphone concerns: a study of battery life*. <http://dl.acm.org/citation.cfm?id=2021978> : Pervasive'11 Proceedings of the 9th international conference on Pervasive computing, 2011.
- Firtman, M. 2010.** *Programming the Mobile Web*. <http://shop.oreilly.com/product/9780596807795.do> : O'Reilly Media, 2010.
- Fling, B. 2009.** *Mobile Design and Development: Practical concepts and techniques for creating mobile sites and web apps (Animal Guide)*. <http://shop.oreilly.com/product/9780596155452.do> : O'Reilly Media, 2009.
- Google A. 2013.** Dashboards. [En línea] 2013. [Citado el: 10 de 10 de 2013.] <http://developer.android.com/about/dashboards/index.html>.
- Google B. 2013.** Data Storage. [En línea] 2013. [Citado el: 10 de 10 de 2013.] <http://developer.android.com/guide/topics/data/data-storage.html>.
- Google C. 2013.** Content Provider. [En línea] 2013. [Citado el: 15 de 10 de 2013.] <http://developer.android.com/guide/topics/providers/content-providers.html>.
- Guo, Q. 2012.** *Android Health-Care App: Multi-function Step Counter*. <http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-16542> : Student Thesis, Mid Sweden University, Faculty of Science, Technology and Media, Department of Information Technology and Media, 2012.
- Liria, H. 2013.** Kantar World Panel. [En línea] 17 de 04 de 2013. [Citado el: 25 de 09 de 2013.] <http://www.kantarworldpanel.com/es/Noticias/Android-ya-est-en-9-de-cada-10-nuevos-smartphones>.
- Meier, R. 2012.** *Professional Android 4 Application Development (Wrox Professional Guides)*. <http://www.amazon.com/Professional-Android-Application-Development-Guides/dp/1118102274> : Wrox, 2012.
- Oner, M., Pulcifer-Stump, J., Seeling, P., Kaya, T. 2012.** *Towards the Run and Walk Activity Classification through Step Detection – An Android Application*. http://people.cst.cmich.edu/kaya2t/papers/Kaya_EMBC_12.pdf : Proceedings of the 34th Annual International Conference of the Engineering in Medicine and Biology Society, 2012.

Shin, J., Shin, D., Shin, D., Her, S., Kim, S., Lee, M. 2010. *Human Movement Detection Algorithm Using 3-Axis Accelerometer Sensor Based on Low-Power Management Scheme for Mobile Health Care System.* <http://www.springerlink.com/content/t44034618316kq58/> : Springer Berlin Heidelberg, 2010.

Six, J. 2011. *Application Security for the Android Platform.* <http://shop.oreilly.com/product/0636920022596.do> : O'Reilly Media, 2011.

Wargo, J. 2012. *PhoneGap Essentials: Building Cross-Platform Mobile Applications.* <http://www.informit.com/store/product.aspx?isbn=0321814290> : Addison-Wesley Professional, 2012.

Zigurd, M., Dornin, L., Meike, B., Nakamura, M. 2011. *Programming Android.* <http://shop.oreilly.com/product/0636920010364.do> : O'Reilly Media, 2011.

Anexo A. Especificación de Requisitos

En esta sección se presentan los requisitos detallados de la solución y éstos se dividen en requisitos de usuario y requisitos de software.

A.1. Requisitos Detallados de la Solución

En esta sección se presentan los requisitos preliminares de la solución a desarrollar.

A.1.1. Requisitos de Usuario

A continuación se detallan los principales requisitos de usuario (preliminares) de la aplicación propuesta.

- RU001. Recopilación de la información.
 - i. *Descripción:* Recopilar la actividad física desarrollada por el usuario en términos de:
 1. Tiempo de la actividad.
 2. Velocidad de la actividad.
 3. Tipo de actividad.
 - ii. El tiempo mínimo para registrar una actividad como válida será 1 minuto. Este tiempo.
- RU002. Filtros de actividades
 - i. *Descripción:* Generar un filtro que permita diferenciar entre actividades físicas y movimientos que no tengan que ver con alguna actividad física como por ejemplo: andar en automóvil.
- RU003. Conversión a calorías.
 - i. *Descripción:* Desplegar la conversión de la actividad física realizada a la cantidad de calorías quemadas.
- RU004. Calorías quemadas necesarias en un día normal.
 - i. *Descripción:* Desplegar la cantidad de calorías quemadas necesarias para un día normal y sus equivalentes en actividad física considerando la edad del usuario.
- RU005. Gráficos de actividades desarrolladas
 - i. *Descripción:* Mostrar un gráfico con la actividad física registrada diaria, semanal, mensual y anual.
- RU006. Recordatorios de actividad física mínima
 - i. *Descripción:* Permitir crear recordatorios que alerten al usuario si la cantidad física diaria no ha sido suficiente.

- RU007. Creación de cuentas de usuario.
 - i. *Descripción:* Creación de una cuenta de usuario para llevar el registro de la actividad física recopilada por el teléfono.
- RU008. Inicio de sesión y asociación de teléfono.
 - i. *Descripción:* Se debe permitir iniciar sesión en un teléfono y así asociarlo a esta cuenta de usuario.
- RU009. Conexión al servidor web de la aplicación
 - i. *Descripción:* Envío paulatino de los datos recopilados, al servidor web de la aplicación, para su visualización en la versión web de la aplicación.

Los requisitos de usuario para la versión spyware, para teléfonos inteligentes son los siguientes:

- RU0010. Mismas funcionalidades que la versión normal.
 - i. *Descripción:* Permitir las mismas funcionalidades que la versión normal.
- RU0011. Notificaciones de desactivación de GPS.
 - i. *Descripción:* Generar notificaciones si el GPS del teléfono ha sido desactivado por correo electrónico.
- RU0012. Aplicación no detectable.
 - i. *Descripción:* No desplegar ningún tipo de notificación de las acciones hechas por la aplicación, aunque haya incurrido en errores esta última.
- RU0013. Envío de la información recopilada.
 - i. *Descripción:* Enviar la información recopilada automáticamente si se encuentra con una red inalámbrica abierta o conocida.
- RU0014. Seguridad para el usuario de la aplicación y uso indebido.
 - i. *Descripción:* No permitir el registro de los lugares en donde se realiza la actividad física para no generar un dispositivo de seguimiento del usuario del teléfono del cuál no tenga conocimiento.

Los requisitos de usuario para la aplicación Web son los siguientes:

- RU0015. Cuentas de usuario.
 - i. *Descripción:* Permitir el ingreso de los usuarios usando la cuenta creada a través de esta versión Web o del teléfono.
- RU0016. Selección de teléfonos asociados.
 - i. *Descripción:* Permitir seleccionar entre los diversos teléfonos asociados al usuario.
- RU0017. Despliegue de estadísticas.
 - i. *Descripción:* Mostrar las estadísticas de la actividad física recopilada por el teléfono en gráficos que muestren actividad diaria, semanal, mensual y anual.

Los requisitos de usuario para el servidor Web se detallan a continuación:

- RU0018. Canal de comunicaciones.

- i. *Descripción:* Debe ser el canal de comunicaciones entre los dispositivos del usuario y la versión web de la aplicación.

A.1.2. Requisitos de Software

A continuación se presentan los requisitos de software para la aplicación móvil para teléfonos inteligentes:

- RS001. Filtro para la detección de una actividad válida
 - a. *Requisito Asociado:* RU003.
 - b. *Descripción:* Se debe generar un filtro que permita discernir entre actividades que se pueden realizar por un humano como: correr, andar en bicicleta a actividades como andar en un vehículo que no requieren actividad física.
- RS002. Nivel de error para una actividad válida.
 - a. *Requisito Asociado:* RU003.
 - b. *Descripción:* Uno de los posibles filtros es considerar la velocidad que se mueve un individuo, pero esto no sería suficiente, por ejemplo, si un auto se moviera lentamente.
- RS003. Recopilación de la velocidad y el tiempo de una actividad válida.
 - a. *Requisito Asociado:* RU002.
 - b. *Descripción:* Se debe registrar la velocidad y tiempo para una actividad válida. Este registro se genera cuando una actividad se ha desarrollado por al menos 1 minuto.
- RS004. Recopilación del tipo de actividad.
 - a. *Requisito Asociado:* RU002.
 - b. *Descripción:* Se debe usar la velocidad de la actividad para determinar el tipo de actividad desarrollada: Caminar, trote, bicicleta.
- RS005. Conversión a calorías.
 - a. *Requisito Asociado:* RU004.
 - b. *Descripción:* Se debe generar una tabla de conversión entre una actividad física (con velocidad, tiempo y tipo) a cantidad de calorías quemadas.
- RS006. Calorías mínimas diarias para una persona de una determinada edad.
 - a. *Requisito Asociado:* RU005.
 - b. *Descripción:* Se debe investigar sobre los datos de la cantidad de calorías mínimas diarias para una persona de determinada edad. También puede influir el género del usuario en este caso.
- RS007. Tipos de gráficos para actividades realizadas.
 - a. *Requisito Asociado:* RU006.
 - b. *Descripción:* Se deben generar 4 gráficos que representen los siguientes:

- i. Gráfico de la actividad del día: Muestra la cantidad de actividad física realizada durante el día en un gráfico tiempo vs cantidad.
 - ii. Gráfico de la actividad de la semana, mes y año: Mismo gráfico que el anterior, pero el espacio de tiempo es mayor.
- RS008. Calendario para gráficos de actividades realizadas.
 - a. *Requisito Asociado:* RU006.
 - b. *Descripción:* Se debe generar un calendario que le permita al usuario seleccionar un día, semana, mes o año para revisar el gráfico correspondiente a la actividad física desarrollada.
- RS009. Recordatorios
 - a. *Requisito Asociado:* RU007.
 - b. *Descripción:* Debe haber una opción para crear alarmas a una hora que el usuario escoja y los días que el usuario escoja, para recordarle si es que no ha desarrollado la actividad física mínima diaria.
- RS0010. Datos para cuenta de usuario.
 - a. *Requisito Asociado:* RU008.
 - b. *Descripción:* Los datos para la inscripción del usuario son los siguientes: Nombre y apellido, correo electrónico, edad, género, país.
- RS0011. Registro con Facebook.
 - a. *Requisito Asociado:* RU008.
 - b. *Descripción:* Permitir usar el registro a través de un botón de Facebook.
- RS0012. Asociación de cuenta con el dispositivo.
 - a. *Requisito Asociado:* RU009.
 - b. *Descripción:* Asociar la cuenta de usuario al teléfono en que se registró o que inició sesión. Esto permite que el usuario ingrese automáticamente a la aplicación y no sea necesario iniciar sesión cada vez que ingresa a la misma.
- RS0013. Disociar cuenta de aplicación con el teléfono.
 - a. *Requisito Asociado:* RU009.
 - b. *Descripción:* Si el usuario decide desloguearse, se borrarán todos los datos almacenados en el teléfono y se debe disociar este dispositivo de su cuenta de usuario.
- RS0014. Tipo de red para envío de datos.
 - a. *Requisito Asociado:* RU0010.
 - b. *Descripción:* El usuario puede seleccionar si enviar los datos en presencia de redes 3G, Wifi o cuando cualquiera de las dos esté disponible.
- RS0015. Tiempo para envío de datos.
 - a. *Requisito Asociado:* RU0010.
 - b. *Descripción:* El usuario puede escoger cada cuanto se envían los datos recopilados por la aplicación al servidor central: A cada instante, cada una hora, cada 3 horas, cada 12 horas, cada 24 horas.

Los requisitos de software para la versión espía de la aplicación, se detallan a continuación:

- RS0016. Funcionalidades para la versión espía de la aplicación.
 - a. *Requisito Asociado:* RU0011.
 - b. *Descripción:* La versión espía de la aplicación debe cumplir las mismas funcionalidades de la versión normal.
- RS0017. Notificación al usuario asociado por la desactivación del GPS.
 - a. *Requisito Asociado:* RU0012.
 - b. *Descripción:* Si algún dispositivo con la aplicación espía instalada se le desactiva el GPS, se le informará al usuario de la cuenta asociada a través de correo electrónico de este suceso.
- RS0018. No mostrar en la grilla de aplicaciones el icono de la aplicación.
 - a. *Requisito Asociado:* RU0013.
 - b. *Descripción:* La aplicación se debe instalar en el teléfono, pero se debe investigar la forma de no desplegarla en la grilla de aplicaciones instaladas en el teléfono.
- RS0019. Notificaciones de la aplicación.
 - a. *Requisito Asociado:* RU0013.
 - b. *Descripción:* No se debe generar ninguna notificación de la aplicación para la versión spyware.
- RS0020. Envío automático de información recopilada.
 - a. *Requisito Asociado:* RU0014.
 - b. *Descripción:* Si el teléfono se encuentra en una zona Wifi abierta, conocida o con redes móviles habilitadas, se hará automáticamente el envío de la información recopilada.
- RS0021. Tiempo para cada envío de información.
 - a. *Requisito Asociado:* RU0014.
 - b. *Descripción:* El envío de la información recopilada se hará cada 3 horas para prevenir el desgaste de la batería. Si la conexión no está disponible en esas 3 horas, se realizará apenas esté disponible y una vez que se realice volverán a transcurrir 3 horas para realizar el envío nuevamente.
- RS0022. No registrar los lugares que recopila el GPS.
 - a. *Requisito Asociado:* RU0015.
 - b. *Descripción:* Registrar solamente el tiempo, velocidad y tipo de las actividades realizadas. No el lugar.

Los requisitos de software para la versión Web de la aplicación son los siguientes:

- RS0023. Registro en la versión Web.
 - a. *Requisito Asociado:* RU0016.
 - b. *Descripción:* Los datos para la inscripción del usuario son los siguientes: Nombre y apellido, correo electrónico, edad, género, país.
- RS0024. Registro con Facebook en la versión web.

- a. *Requisito Asociado:* RU0016.
 - b. *Descripción:* Permitir el registro en la versión web de la aplicación usando Facebook.
- RS0025. Despliegue de teléfonos asociado.
- a. *Requisito Asociado:* RU0017.
 - b. *Descripción:* Al ingresar a la cuenta de usuario de la versión web, se deben desplegar los teléfonos asociados a esta cuenta y si no tiene ninguno registrado, se le debe indicar que debe registrar al menos un teléfono para usar esta versión.
- RS0026. Tipos de gráficos para actividades realizadas.
- a. *Requisito Asociado:* RU018.
 - b. *Descripción:* Se deben generar 4 gráficos que representen los siguientes:
 - i. Gráfico de la actividad del día: Muestra la cantidad de actividad física realizada durante el día en un gráfico tiempo vs cantidad.
 - ii. Gráfico de la actividad de la semana, mes y año: Mismo gráfico que el anterior, pero el espacio de tiempo es mayor.
- RS0027. Calendario para gráficos de actividades realizadas.
- a. *Requisito Asociado:* RU018.
 - b. *Descripción:* Se debe generar un calendario que le permita al usuario seleccionar un día, semana, mes o año para revisar el gráfico correspondiente a la actividad física desarrollada.

Los requisitos de software para el servidor web son los siguientes:

- RS0028. Servidor basado en una arquitectura del tipo REST.
- a. *Requisito Asociado:* RU0019.
 - b. *Descripción:* Se debe crear un servidor basado en una arquitectura del tipo REST para permitir la comunicación simple entre los dispositivos móviles y la versión web de la aplicación. Además permite a futuro portar la aplicación a otros sistemas operativos o dispositivos sin la necesidad de cambiar la arquitectura del servidor.

Anexo B. Resultados Extendidos de los Experimentos

A continuación, se presentan las tablas con todos los resultados de los experimentos de la sección 6. En primera instancia, se presenta la lista de aplicación con los pasos registrados para las primeras tres velocidades, y después para las dos velocidades restantes.

Tabla 6. Pasos registrados por las aplicaciones seleccionadas para 3 velocidades

Aplicación	Muy Lento 1	Muy lento 2	Lento 1	Lento 2	Normal 1	Normal 2
A	9	11	13	12	18	23
B	20	10	10	30	29	22
C	0	0	5	9	8	16
D	5	16	4	11	6	4
E	22	20	30	17	17	13
F	9	14	18	34	15	25
G	15	8	14	7	34	36
H	35	20	36	12	28	21
I	23	7	14	7	34	22
J	15	19	18	22	18	20
K	5	3	30	25	30	20
L	10	3	17	6	17	22
M	8	6	10	11	17	6

Tabla 7. Pasos registrados por las aplicaciones seleccionadas para 2 velocidades

Aplicación	Rápido	Rápido 2	Muy Rápido	Muy Rápido 2
A	19	17	16	14
B	33	27	27	13
C	8	12	11	11
D	5	6	4	5
E	13	17	17	13
F	24	39	35	17
G	10	23	34	20
H	12	20	30	20
I	14	24	9	24
J	14	20	14	20
K	30	30	34	35
L	14	14	30	31
M	21	34	23	22