UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL CÓRDOBA
DIRECCIÓN DE POSGRADO

TESIS PARA MAESTRÍA EN INGENIERÍA EN
SISTEMAS DE INFORMACIÓN

Autor:
Ing. Maximiliano Abrutsky

Año 2018
“Aplicación de relojes inteligentes y otros sensores, en la recopilación de datos sensibles para el control de rendimiento de atletas de resistencia”

**Director:** Dr. Ricardo Medel

-----------------------------------

**Miembros del Tribunal de Tesis**

*Mgter.:* Cynthia Corso

-----------------------------------

*Mgter.:* Juan Francisco Giró Martín

-----------------------------------

*Mgter.:* Hugo Frías
Prólogo

La prevención de lesiones en los atletas entrenados por profesionales de las ciencias del movimiento es un aspecto crítico de su labor que requiere mucho control y seguimiento, actividades que se dificultan frente al vertiginoso incremento de atletas principiantes en deportes de resistencia como ciclismo, natación y carrera. Los entrenadores están obligados a tener mediciones más precisas de las sesiones de entrenamiento de sus atletas.

Este trabajo de investigación aplicada tiene como ejes la identificación de las variables más importantes a medir no invasivamente en ámbitos deportivos, la evaluación de las tecnologías o dispositivos “ponibles” que actualmente se utilizan para ello, la comparación entre los mayores fabricantes y principalmente el desarrollo de una aplicación que se ejecute en un modelo previamente seleccionado. El resultado es una aplicación que se ejecuta en un reloj inteligente, tecnología donde el ciclo de vida de desarrollo y técnicas de diseño, distan considerablemente de los medios tradicionales como la web o celulares inteligentes. El aprendizaje de las limitaciones, bondades, restricciones, buenas prácticas y aspectos relevantes inherentes a la tecnología es el objetivo que persigue este trabajo.
**Tabla de Contenidos**

Listado de Figuras
Listado de Tablas

**Introducción**
- Justificación .................................................. 1
- Objetivo ............................................................... 3
- Marco de trabajo .................................................. 4

**Estado del Arte “Wearables y la actividad deportiva”** ............................................... 7
- El Internet de las Cosas (IoT) ........................................ 8
- Tipos y usos de wearables ......................................... 10
- Aplicaciones de los wearables a la salud y al deporte .......... 11
- Resumen del capítulo ............................................... 13

**Fundamentos de aplicación** ........................................ 15
- Lo que no se mide, no se puede mejorar .......................... 15
- Enfoque sistémico .................................................. 15
- Dosificación ........................................................... 17
- Mediciones ............................................................... 19
- Rendimiento y ciclo de mejora continua .......................... 23
- Resumen del capítulo ............................................... 24

**Marco Tecnológico** .................................................. 25
- Introducción al desarrollo de wearables .......................... 25
- Cómo se ve una app en un wearable: UX Guidelines para wearables .......................... 26
- Cómo se comunican los wearables: ANT+ .................................................. 28
- Qué datos generan los wearables: Estructuras de datos .......................... 30
- Opciones de hardware: Marcas Principales .................................. 42
- Selección de hardware: Garmin ........................................ 45
- Plataforma de desarrolladores GARMIN .................................. 47
- Garmin Connect IQ .................................................. 48
- Tipos de aplicaciones que se pueden desarrollar .................. 50
- Lenguaje de programación: Monkey C ............................ 51
<table>
<thead>
<tr>
<th>Nro.</th>
<th>Título</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Modelo de caja negra.</td>
</tr>
<tr>
<td>2</td>
<td>Atleta bajo un sistema de entrenamiento.</td>
</tr>
<tr>
<td>3</td>
<td>Pirámide de jerarquía de necesidades de entrenamiento de resistencia.</td>
</tr>
<tr>
<td>4</td>
<td>Ciclo de mejora continua de Deming.</td>
</tr>
<tr>
<td>5</td>
<td>Archivo FIT a la computadora personal.</td>
</tr>
<tr>
<td>6</td>
<td>Listado de perfiles FIT estándares.</td>
</tr>
<tr>
<td>7</td>
<td>Estructura de un archivo FIT.</td>
</tr>
<tr>
<td>8</td>
<td>Archivo FIT y Perfil FIT Global.</td>
</tr>
<tr>
<td>9</td>
<td>Perfiles FIT: global y de productos.</td>
</tr>
<tr>
<td>10</td>
<td>Protocolo FIT.</td>
</tr>
<tr>
<td>11</td>
<td>Ejemplo de fragmento del esquema TCX.</td>
</tr>
<tr>
<td>12</td>
<td>Ejemplo de fragmento del esquema GPX.</td>
</tr>
<tr>
<td>13</td>
<td>Círculo de experiencia de desarrollo Garmin.</td>
</tr>
<tr>
<td>14</td>
<td>Diagrama despliegue desde sensores a servidores.</td>
</tr>
<tr>
<td>15</td>
<td>Diagrama despliegue Garmin Connect Push API.</td>
</tr>
<tr>
<td>16</td>
<td>Prototipo pantalla principal.</td>
</tr>
<tr>
<td>17</td>
<td>Prototipo selección de Test.</td>
</tr>
<tr>
<td>18</td>
<td>Prototipo presentación de Test.</td>
</tr>
<tr>
<td>19</td>
<td>Prototipo Test en ejecución.</td>
</tr>
<tr>
<td>20</td>
<td>Prototipo resultado de Test.</td>
</tr>
<tr>
<td>21</td>
<td>Aplicación Test en ejecución.</td>
</tr>
<tr>
<td>Nro.</td>
<td>Título</td>
</tr>
<tr>
<td>------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>Matriz preponderada de fabricantes de wearables y características.</td>
</tr>
<tr>
<td>2</td>
<td>Test de valoración de rendimiento por deporte.</td>
</tr>
<tr>
<td>3</td>
<td>Fragmento archivo FIT resultante.</td>
</tr>
</tbody>
</table>
Introducción

Justificación
El incremento del número de atletas en deportes de resistencia en las últimas décadas ha sido de gran magnitud, independientemente del nivel de los participantes que se inician como amateurs pero con alta tasa de posterior profesionalismo.

Los principales motivadores de esta tendencia nacen de la toma de conciencia social de los ritmos sedentarios de vida y su impacto negativo en la salud, como así también de regulaciones estatales que promocionan el ciclismo y la carrera (Castillo, 2007).

La facilidad para iniciarse en estos deportes individuales radica en que no se requiere conformar un grupo y la motivación intrínseca de autosuperación impulsa la constante práctica deportiva y abre las puertas a aquellos que se profesionalizan para competir contra terceros.

Sin embargo, hacer ejercicio no necesariamente significa salud y el riesgo de lesión es tan alto o más aún que el sedentarismo (Pérez et al., 2015). La conciencia y responsabilidad en el deporte previene daños y lesiones (desde pequeños traumatismos hasta consecuencias irreversibles). El seguimiento profesional y capacitado a los atletas es la vía para conducir la evolución de éstos.

Para los profesionales de las ciencias del movimiento, el seguimiento riguroso de sus atletas es una actividad clave dentro de la planificación, seguimiento, control y mejora continua de sus planes de entrenamiento. Independientemente del nivel del atleta y de su objetivo final, (desde amateurs -cuya meta es la reducción de peso- hasta el profesional que se prepara para competencias internacionales) es clave el poder administrar y controlar de cerca la evolución lo más rigurosamente posible, con foco en la obtención de datos claves para adecuar el entrenamiento.
A su vez, históricamente la habilidad de los entrenadores para armar planes de entrenamiento con el objetivo de lograr el mayor rendimiento de los atletas puede ser atribuida a los años de experiencia personal. Actualmente, un enfoque más moderno es adoptar métodos científicos en el desarrollo de programas de entrenamiento óptimos.

Así entonces, este trabajo de investigación aplicada, se enfocará en la identificación, recolección y análisis de datos sensibles y útiles para: 1) la cuantificación de la carga en el entrenamiento (la relación entre el volumen, intensidad y densidad del entrenamiento); 2) el control del proceso de entrenamiento; y 3) el análisis exacto del efecto del entrenamiento, permitiendo la comparación entre lo programado previamente y lo logrado por el atleta. En dicha medición de datos, se han propuesto, por un lado, métodos subjetivos de esfuerzo tales como observación directa o cuestionarios diarios; y por otro lado métodos más objetivos como el monitoreo fisiológico y los índices de stress de entrenamiento (Borressen y Lambert, 2009).

El crecimiento tecnológico vertiginoso en la última década impactó significativa y hasta disruptivamente en varias industrias y ramas de las ciencias y esto no excluyó a las ciencias del movimiento, en particular a la planificación del entrenamiento deportivo en atletas de alto rendimiento. Incluso el deporte amateur también experimentó un crecimiento exponencial, impulsado en parte por la fuerza comunicacional de las redes sociales, las aplicaciones orientadas al deporte y la oferta de dispositivos tecnológicos. En la actualidad, la empresa Garmin, al igual que decenas de empresas más, comercializa una gama de cientos de productos para entrenamiento y bienestar, desde relojes y pulseras inteligentes, mini computadoras GPS de alta capacidad de procesamiento hasta hardware específico según el tipo de deporte (Garmin, 2017).

En resumen, los profesionales de las ciencias del movimiento necesitan información, para el seguimiento y control de los programas de entrenamiento de sus atletas. La obtención de ésta, ya sean datos objetivos o subjetivos, en tiempo y...
forma, es de dificultosa ejecución práctica. Disponer de los mismos, sin las vías prácticas y visuales para procesarlos, hacen que la interpretación y posterior toma de decisiones, no sea la más eficiente.

Finalmente, para validar el resultado del presente trabajo, se aplicará y evaluará un caso particular. Consecuentemente a esta experiencia, se obtendrán una serie de “buenas prácticas” en recolección de datos que, a modo de recomendación, serán aplicables y adaptables a diferentes sistemas de entrenamiento.

**Objetivo**

El objetivo del presente trabajo es la creación de una aplicación de referencia para deportistas de resistencia que, a través de una serie de buenas prácticas metodológicas, permita extraer, no invasivamente, datos sensibles de sus sesiones de entrenamiento.

**Objetivos secundarios**

- Describir el estado actual de dispositivos “ponibles” (del inglés *wearables*) aplicados a deportes de resistencia.
- Identificar las variables más importantes en ámbitos deportivos que sean viables de medir de manera no invasiva.
- Evaluar y seleccionar algún fabricante y las herramientas de desarrollo ofrecidas para la creación de aplicaciones personalizadas.
- Desarrollar una aplicación para relojes inteligentes o dispositivos seleccionados, con una implementación concreta en algún deporte.
- Definir una serie de buenas prácticas para el desarrollo en dispositivos “ponibles”.
Marco de trabajo

Metodología y actividades

El desarrollo del proyecto se basa en procesos de investigación utilizando técnicas deductivas y exploratorias propias de la investigación aplicada.

La primera etapa es exploratoria, donde se hace un análisis crítico del estado del arte, realizando un estudio bibliográfico para consolidar fundamentos propios de la disciplina deportiva involucrada.

Se acompaña la fase exploratoria con un estudio de mercado sobre equipamiento, hardware específico, software actuales, herramientas de análisis de datos y progresos de entrenamiento, relevando sus fortalezas y limitaciones.

De este análisis de la ciencia del ejercicio, y del estudio de las tecnologías disponibles, surgirá una propuesta de desarrollo de aplicación para captura de datos sensibles, que permitan un eficiente seguimiento del progreso del atleta.

Una vez concluida la solución propuesta, se realizarán evaluaciones de los resultados obtenidos, y así definir las buenas prácticas metodológicas que se persiguen como objetivo.

1. Estudio Bibliográfico.

Realizar un análisis crítico del estado del arte actual, estudiando trabajos relacionados y analizando soluciones actuales de relojes inteligentes y dispositivos similares, aplicados a la recolección sensorial de datos útiles, para el seguimiento de progreso en deportes de resistencia.

Determinar el impacto y adopción actual de estas nuevas tecnologías.

Presentar diferentes posibilidades de aplicación y ventajas de relojes inteligentes y dispositivos similares, en ámbitos de la actividad física.
2. **Definiciones**

Evaluar la importancia de la medición para el control de la evolución del entrenamiento.

Proponer una forma sistémica de vincular las mediciones con el rendimiento dentro de un sistema de entrenamiento deportivo.

Identificar datos críticos necesarios para el análisis del progreso del entrenamiento.

Evaluar la dificultad de obtención de las distintas mediciones útiles en el control.

3. **Estudio tecnológico**

Analizar las tecnologías más apropiadas para desarrollar el proyecto, marca de relojes inteligentes, computadoras para bicicletas, hardware específico de la industria, etc.

Identificar aspectos claves y principios a tener en cuenta en la programación de aplicaciones para dispositivos ponibles.

Análisis de posibles limitaciones impuestas por la madurez actual de la tecnología.

Determinar qué estructuras de datos generan estos dispositivos, y cómo se comunican entre ellos: protocolos y estándares existentes en la industria.

Presentar las marcas principales actualmente en mercado de fabricantes de hardware.

Evaluar las alternativas de hardware, y seleccionar una junto con el entorno de desarrollo y plataforma necesaria.

4. **Implementación tecnológica**

Seleccionar, adaptar o definir, un ciclo de desarrollo de software.

Identificar el alcance funcional de la aplicación que se va a construir, procurando que el mismo sea lo suficientemente amplio para aprender buenas prácticas.
Definir las características principales que va a tener el proyecto, es decir, qué estructuras de datos se obtendrán, cómo se van a procesar y cómo se presentarán en los dispositivos.

Programar la aplicación que cubra el alcance definido anteriormente, desarrollando cada fase del ciclo de desarrollo optado.

5. Documentación

Redacción y escritura de la tesis.
El término “wearable” es una palabra inglesa que significa literalmente “que se puede llevar puesto”. Así pues, la tecnología wearable, o tecnología ponible, es aquella electrónica diseñada para ser vestida, bien como complemento o bien como parte de algún material usado en la ropa. Según Ordóñez (2016), la tecnología wearable es habitualmente referenciada en la jerga tecnológica como WT (Wearable Technology), y los dispositivos asociados como WD (Wearable Devices) o simplemente wearables. En castellano, es habitual acompañar a estos dispositivos con los adjetivos “tecnológico” o “inteligente”: chaqueta tecnológica, reloj inteligente, etcétera.

Para Muñoz (2015), los wearables o “ponibles” son dispositivos informáticos de pequeñas dimensiones, fáciles de utilizar y que, según de que aparato se trate, permiten llevar con el usuario, toda la información médica necesaria; pueden ser dispositivos electrónicos o biosensores generalmente inalámbricos, integrados en ropa o accesorios sobre el cuerpo para recabar y mostrar datos de constantes vitales y actividad física.

WT puede desempeñar muchas de las tareas de los ordenadores y los dispositivos móviles, y son más sofisticados, ya que incorporan sensores para interactuar con el cuerpo y con el entorno. Una característica fundamental de WT y WD es la capacidad de conectividad inalámbrica (en particular, pero no sólo, a Internet), tal que permiten al portador acceder a diversa información en tiempo real. En muchos casos, permiten además la introducción de datos por parte del usuario, así como el almacenamiento de información en local en el dispositivo. El propósito general de WT es crear de forma adecuada, constante, transparente y portable el acceso a la electrónica y computación en la vida diaria de los individuos (Ordóñez, 2016).

Durante este siglo han proliferado todo tipo de pulseras, anillos, camisetas, zapatillas, cinturones, gorras y cualquier otro elemento "ponible" que incluye
sensores para registrar los movimientos y los parámetros corporales: pulsaciones, ritmo cardíaco, ritmo respiratorio, presión arterial, nivel de glucosa, saturación de oxígeno en sangre, temperatura, sudoración, etcétera. Aparentemente muchos de estos dispositivos parecían formar parte de una moda pasajera; pero en concordancia con Muñoz (2015), nada está más lejos de la realidad.

La tecnología está revolucionando todos los aspectos cotidianos, y el deporte no es una excepción. Al respecto, Pérez y Requejo (2016), apuntan que cada vez se desarrollan más aplicaciones y dispositivos que permiten contrastar el entrenamiento diario y tener una información notable sobre el rendimiento y la progresión de los deportistas, y también de los propios aficionados. Zapatillas con GPS, pulseras de monitorización, smartwatches, aplicaciones de geolocalización; el fenómeno de Internet de las cosas (IoT) a través de wearables, es una realidad que cada día tiene más relevancia en el mundo del deporte.

Así entonces, hoy día, grandes corporaciones tecnológicas, están enfocadas en incorporar estos mecanismos en el campo de la salud y asociados a la actividad física del consumidor. En este sentido, Apple va a la vanguardia, seguida de firmas como Google, Nike, Samsung, entre otras, las cuales han identificado las necesidades de los usuarios, así como las de los deportistas, pacientes y médicos; han comenzado a crear dispositivos que ayudan a obtener datos de forma más sencilla y de mayor calidad, además de dar los principios para obtener una fotografía completa de la salud de poblaciones a gran escala (Psyma Group AG, 2016).

Muchos expertos aseguran que durante los próximos años, el mercado de los wearables será uno de los más prolíficos y rentables. Según un estudio de Morgan Stanley (citado por Muñoz, 2015) sobre Wearables Devices se espera que en 2017 la Tasa de Crecimiento Anual Compuesto, alcanzará el 154%, respecto al 2013. Más del doble de las expectativas del mercado. En complemento, datos de la agencia MaRS Market Insights (citada por Psyma Group AG, 2016), estiman que
el mercado de los dispositivos wearables tendrá un crecimiento anual de 485 millones en envíos de equipos para 2018.
El ingreso global de mercado de la salud para este tipo de dispositivos se calcula que tuvo 1.1 billones de dólares en 2014 y que para 2018 crecerá a 2.9 billones de dólares. Igualmente, para 2015, el sector de los wearables en el ámbito deportivo ya había alcanzado los 2.4 mil millones de dólares y se prevé que en 2018 se sitúe en los 5.1 mil millones (Gilibets, 2015). La mayoría de los ingresos del segmento de ambos mercados (ingresos en la categoría salud y de acondicionamiento físico) viene de América del Norte (52%), seguido de Europa (26%), Asia-Pacífico (16%) y el resto del mundo (7%).

El Internet de las Cosas (IoT)

La tecnología wearable, a criterio de Ordóñez (2016) se considera una parte integrante del nuevo paradigma de Internet de las Cosas (IoT, Internet of Things), una porción de una red de objetos físicos embebidos con electrónica, software, sensores y conectividad que permiten el intercambio de datos con fabricantes, operadores y otros dispositivos sin intervención humana.

IoT permite la identificación e interconexión de todos los objetos a través de Internet, de manera que conecta el mundo tradicionalmente offline al mundo online, para mejorar procesos, aumentar eficiencia y disminuir riesgos. Así, cualquier objeto con sensores y conectividad formará parte en un futuro cercano de IoT, siendo WT el puente natural para conseguirlo. IoT se compondrá de casas inteligentes (Smart houses), ciudades inteligentes (Smart cities), fábricas inteligentes (Smart factories), etcétera. Para ello, el primer paso es dotar a la vida diaria de objetos inteligentes (Smart gadgets), parte de los cuales (aquellos que se pueden llevar puestos) son los WD.
Tipos y usos de wearables

Son muchos los tipos de wearables que se encuentran en el mercado, cuyo diseño y colocación en el cuerpo humano dependerá del uso que se les pretenda dar. Existen varios criterios para categorizar estos dispositivos, siendo la clasificación siguiente, la más habitual (Soto, 2014):

- Relojes inteligentes (smartwatches): este tipo de dispositivos ofrece las funciones básicas de un reloj, incluyendo además, aplicaciones propias de un dispositivo IOS o Android. Permite por ejemplo, gestionar llamadas, abrir correos, transcribir mensajes, monitorizar la actividad física, conectarse a redes sociales e, incluso, realizar pagos.

- Gafas inteligentes (smartglasses): dentro de esta categoría destacan las Google Glass, las cuáles pretenden cambiar la manera de comunicarse y de ver el mundo. Pueden utilizarse sin necesidad de realizar ningún movimiento ya que únicamente se controlan por voz. Se pueden aplicar en diferentes ámbitos, como pueden ser la educación, el turismo, la medicina o el marketing. Desde su lanzamiento, este tipo de dispositivos ha tenido que enfrentarse a numerosas debilidades. Algunos de los problemas que retrasaron su salida al mercado hacen referencia a la privacidad, la poca duración de la batería, la escasa visibilidad de la pantalla en condiciones de mucha luz o la poca fiabilidad del control por voz (Alviz, 2013).

- Pulseras inteligentes (smartbands): se trata de bandas cuya función es la monitorización de la actividad del usuario. Su uso se puede trasladar al ámbito deportivo, o al día a día del usuario. Pueden almacenar información sobre el ritmo cardíaco, horas y calidad del sueño, los pasos andados, calorías quemadas, etcétera.

- Anillos inteligentes: son uno de los wearables más básicos y su pequeño tamaño hace que sus funciones estén muy limitadas en comparación con el
resto de dispositivos. Sin embargo, no se quedan atrás, ya que por ejemplo, permiten controlar el smartphone sin necesidad de sacarlo del bolsillo.

- Ropa inteligente: se trata de prendas de ropa a las que se les han añadido nuevas prestaciones dada la tecnología existente. Su uso es muy diverso, desde chaquetas que se adaptan a los cambios del tiempo y monitorizan el pulso a la hora de hacer deporte; hasta las específicamente deportivas, que miden el gasto calórico para tener una dieta equilibrada y un estilo de vida saludable. Un caso particular de mucha utilidad en campo deportivo, es el dispositivo Stryd, que se utiliza con las zapatillas deportivas que el atleta prefiera, otorgándole la capacidad de medir potencia y comunicarla a través de red inalámbrica (Stryd, 2017).

Aplicaciones de los wearables a la salud y al deporte

La actividad física y la salud se encuentran estrechamente ligadas, elemento que pareciera justificar la insistencia permanente de la sociedad en la importancia de practicar algún deporte o realizar algún tipo de ejercicio. Actualmente, hábitos como la práctica del running o el comer de forma más sana, están de moda. En asociación a ello, se ha extendido entonces, la inclusión en la cotidianidad de los individuos, de dispositivos cuantificadores de la actividad física, lo cual trae consigo una serie de ventajas (Ounae, 2015):

- Ayudan al usuario a controlar su actividad física haciendo un seguimiento diario del ejercicio realizado y sus indicadores de salud, como pueden ser el ritmo cardíaco, el número de calorías quemadas, las horas de sueño o las calorías ingeridas.

- Alejan al usuario de determinados hábitos de vida poco recomendables, como el sedentarismo, que se encuentra asociado a numerosas patologías
cardiovasculares. Ejercen aquí un rol previsor mediante la evaluación personal y el auto-control.

- Ofrecen al usuario un diagnóstico rápido de su estado de salud, dado que monitorizan diversas variables de su vida diaria que tienen especial relevancia en la misma. La fiabilidad varía en función del dispositivo.

- Introducen el concepto de e-salud y devuelven al usuario el control sobre su propia condición física, mediante el análisis de los datos recopilados por estos dispositivos sobre dichas variables. Facilitan al usuario una serie de recomendaciones adaptadas a su estado de salud.

- Anima al usuario a aumentar su actividad física diaria, a hacer más ejercicio y a perder peso. Lo logra a través de la “gamificación”, que trata de motivar al usuario mediante la entrega de premios virtuales en función de los kilómetros recorridos, pasos dados o en función de la posición en la clasificación en un ranking con los amigos.

Así entonces, el ejercicio se está convirtiendo en un juego gracias a estos dispositivos. La actividad física ya no significa sufrimiento, sino que se convierte en un reto para superar a los amigos y alcanzar los objetivos previstos. En adición a tales ventajas, los dispositivos destinados al cuidado de la salud se encargan de la medición de variables biológicas que se transmiten a aplicaciones específicas de dispositivos móviles, donde los datos se guardan y se procesan. Actualmente, los fabricantes tratan estos dispositivos como accesorios para el usuario, dentro de la “moda”, ya que intentan que éstos combinen con la ropa como si se tratase de otro complemento más. El gran uso de los wearables en la salud se basa en las siguientes causas (Ávila, 2015):

- La facilidad de medición de las variables biológicas, ya que hay tradición y experiencia en aparatos para su medida con límites estandarizados.
- Independientemente de la edad de las personas y de si están sanas o enfermas, la salud es un tema que preocupa a la gran mayoría.

- Existen determinadas situaciones, por ejemplo en el deporte, en los que la monitorización de ciertas variables puede tener una importancia más relevante a la hora de conseguir planes de entrenamiento personalizados en entrenamientos de élite.

Estos dispositivos son capaces de medir prácticamente todas las variables biológicas del usuario (ritmo cardíaco, oxígeno en sangre, encefalograma, respiración, temperatura corporal, fases del sueño, etcétera). Además, proporcionan una serie de informaciones adicionales (López, 2015), como pueden ser:

- Localización geográfica del sujeto: esto es importante en el caso de que las personas monitorizadas tengan un cierto riesgo de pérdida (demencias) o pongan su vida en peligro (en deportes extremos).

- Monitorización continuada y transmisión de datos: la aplicación va registrando la variable medida a lo largo del tiempo y en casos determinados, se puede transmitir utilizando el teléfono en el que se registran los valores.

Finalmente, las variables que más miden los usuarios son la distancia (45%), las calorías consumidas (21%) y los objetivos establecidos (8%) (Salah, McIntosh, y Rajakulendran, 2014).

**Resumen del capítulo**
En consecuencia a lo anteriormente expuesto, pudiera concluirse que el impacto de estos nuevos dispositivos es innegable y cada vez hay más empresas que aprovechan la tecnología para lanzar propuestas atractivas, tanto para deportistas profesionales como amateurs. No obstante, aunque los wearables no mejoran directamente el rendimiento físico, son muy beneficiosos desde el punto de vista...
de la salud. Estos gadgets no pueden suplantar el consejo médico pero a través de la lectura de datos, pueden indicar cuándo ha llegado el momento de detenerse en el entrenamiento, un aviso fundamental para los deportistas de élite o las personas que padecen alguna enfermedad. De hecho, la nueva generación de dispositivos inteligentes ya no se limitará a recoger los datos sino que también los analizará y podrá sugerir alternativas de ejercicios más adecuados en base a objetivos propuestos.
Fundamentos de aplicación

En este capítulo se abordarán los conceptos básicos relacionados al deporte de resistencia, y a la ciencia del movimiento en sí, que sirvan de base a la comprensión del campo de aplicación de la tecnología wearable.

Lo que no se mide, no se puede mejorar

“Lo que no se mide, no se puede mejorar”. Frase muy popular atribuida a unos de los máximos referentes en administración, a Peter Ferdinand Drucker, donde manifiesta la importancia de la medición, aplicable en cualquier índole donde se tenga la mejora como objetivo. Ahondado sobre los orígenes de la frase, es posible remontarse al Lord William Thomson Kelvin (Lord Kelvin), físico y matemático británico quien exponía más explícitamente: “Lo que no se define no se puede medir. Lo que no se mide, no se puede mejorar. Lo que no se mejora, se degrada siempre”.

“Definir”, “Medir” y “Mejorar”, son claves en el desarrollo de este trabajo, conceptualmente hay que definir que es un sistema de entrenamiento, cómo se mide la dosificación de carga de entrenamiento, y que impacto ocasiona para poder mejorararlo.

Enfoque sistémico

Según lo señalado en Investopedia (2017), el modelo de caja negra muy utilizado en ámbitos informáticos y computacionales, indica que una serie de procesamientos internos desconocidos, transforman una entrada en una salida, siendo esta última dependiente de la entrada (Figura 1, en la página siguiente).

Por otro lado, la anatomía humana, ciencia dedicada al estudio de estructuras macrocómpicas del cuerpo humano, clasifica a conjunto de órganos asociados en una función en común, en sistemas y aparatos según los mismos estén formados
predominantemente por los mismos tipos de tejidos. A su vez, un aparato puede estar compuesto por varios sistemas, como lo es el aparato locomotor que integra sistema muscular, óseo, articular.

**Figura 1:** Modelo de caja negra.
**Fuente:** Investopedia (2017)

Entonces cada aparato y sistema en la anatomía humana es en sí misma, un sistema de altísima complejidad, todos están relacionados entre sí y en conjunto, conforman al cuerpo humano como uno de los sistemas complejos de mayor interés de estudio. La teoría general de sistemas, permite así, abordar el estudio de sistemas complejos adoptando que están compuestos a su vez, por sub-sistemas de menor complejidad (Drake et al., 2015).

Trasladando estos fundamentos al área de estudio de interés en el presente trabajo, que es el entrenamiento deportivo, y con base en el modelo de caja negra, se asume que el cuerpo humano responde con diferentes rendimientos, en concordancia a distintas dosificaciones de entrenamiento, tal como se aprecia en la Figura 2.

**Figura 2:** Atleta bajo un sistema de entrenamiento.
**Fuente:** Elaboración propia.
En estos modelos donde se proponen al atleta como un sistema, asumen que es éste quien recibe como entrada la carga de entrenamiento, y el rendimiento, es la salida que produce. Si se sobrecarga al sistema, se puede lesionar, si la carga es baja, su rendimiento no es al máximo, qué medir del sistema, cómo medirlo, entender las mediciones y posteriormente cuantificar con criterio, es el resumen del proceso científico de un entrenador de alto rendimiento. El análisis riguroso y exacto de la respuesta del atleta a la dosis de entrenamiento le permite al entrenador lograr la máxima respuesta (rendimiento) con el mínimo estrés (entrenamiento)

**Dosificación**

Pese a que en el deporte de resistencia se acostumbra a cuantificar la carga de entrenamiento, en realidad sigue siendo un problema por resolver. Actualmente muchos entrenadores diseñan el programa de entrenamiento, pero no cuentan con herramientas para cuantificar la carga a la que realmente se expone el atleta, y así poder controlar cuidadosamente la dosis (entrenamiento) y su respuesta (rendimiento). Se asumen que todos los métodos de cuantificación de la carga son imperfectos (Esteve y Cejuela, 2011).

Todo sistema de programación de entrenamiento en deportes de resistencia, requiere de cuantificación y medición de carga, independientemente de las imperfecciones propias de cada uno. De hecho, en toda fase del entrenamiento es fundamental la correcta manipulación de las variables críticas (esencialmente volumen o cantidad del entrenamiento, e intensidad o calidad del entrenamiento).

Así, las variables básicas e indispensables, comunes a cualquier sistema o enfoque de entrenamiento, son el volumen e intensidad, y una variable más relacionada a la carga global, es la densidad. El primero es muy sencillo de medir, ya que para el volumen se aplican unidades estándares de tiempo o distancia para su cuantificación. Por ejemplo, metros en natación y kilómetros en ciclismo o carrera, o bien, minutos y horas pedaleando en la bicicleta en una determinada
zona de esfuerzo del atleta. Existen diferentes propuestas para determinar los límites de cada zona de entrenamiento.

No obstante, en relación a la intensidad del entrenamiento, existen muchas posibilidades, habiendo propuestas basadas en la percepción subjetiva del esfuerzo como la señalada por Foster (2004), que es muy simple, ya que no requiere ningún tipo de tecnología, hasta el modo más utilizado por los atletas de resistencia que es la frecuencia cardíaca (FC) (o el % de la FC máx.), así como otras opciones que requieren invadir al atleta como es el caso de la medición de lactato sanguíneo, o usar tecnología costosa, y solo aplicable al marco de la investigación científica como es el caso de la medición del consumo de oxígeno (VO$_2$) (o el % del VO$_2$ máx.).

Una simplificación a tres zonas de intensidad de entrenamiento para deportistas de resistencia, basadas en los umbrales ventilatorios y correlacionándolos con los niveles de lactato en sangre es lo que propuso Seiler (2009). Esta propuesta de tres zonas (modelo trifásico) es la referencia actualmente en los deportes de resistencia y es ampliamente citada en la literatura científica específica (Arroyo, 2013). Por ende, el mencionado modelo trifásico de Sieler, se adopta como necesidad en el entrenamiento de resistencia, el mismo, se refleja en los tres primeros niveles básicos de la pirámide de jerarquía de necesidades en el entrenamiento de resistencia, según se muestra en la Figura 3 de la página siguiente.

La importancia de cuantificar, con toda su dificultad inherente, para poder dosificar cuánto se incrementa o reduce la carga total de entrenamiento, es crítico en la búsqueda de resultados significativos; básicamente se puede incrementar demasiado poco de una sesión a otra, o bien incrementarse en exceso. Lo primero producirá una menor adaptación (un mero mantenimiento del nivel alcanzado hasta entonces, es decir, una falta de estímulo eficaz), y lo segundo un alto riesgo de sobrepasar las posibilidades de respuesta del atleta y así conducir adaptación negativa (un estímulo excesivo). El atleta que trabaja sistemáticamente con cargas de entrenamiento muy elevadas para sus posibilidades ingresa primero en un
estado de overreaching (o sobrecarga), que primero es funcional (y necesario en cierta medida para mejorar), pero que luego se hace no funcional (non functional overreaching), y está asociado a mala adaptación, lesiones y fatiga crónica (síndrome de sobreentrenamiento).

Figura 3: Pirámide de jerarquía de necesidades de entrenamiento de resistencia.  

Mediciones
Es condición imperante, que la medición de cada sesión de entrenamiento del atleta, se realice en tiempo real y continuamente durante todo el entrenamiento; de esta manera, se podrá corroborar que lo dosificado sea realmente lo que realiza el atleta, y no que éste, por falta de información o desconocimiento, o bien por aspectos motivacionales o de cualquier índole, hubiere aumentado o disminuido la carga y/o intensidad indicada en la sesión de entrenamiento. He aquí, donde se aprovechan al máximo los wearables y su capacidad de lecturas de datos sensibles, para el seguimiento de cómo está respondiendo el atleta, al estímulo indicado. Entre las mediciones importantes, destacan:
**Tiempo y distancia:** constituyen las variables clave para determinar la cantidad del entrenamiento (lo que los coaches denominan volumen). Son variables de gran utilidad y sencillez en la medición; cualquier cronómetro puede medir tiempo y la distancia, es lo más básico en cualquier dispositivo con GPS o acelerómetro. Relacionada a esta variable, está también el desnivel acumulado, que indica los metros que el atleta ganó y perdió, durante el entrenamiento. Por desnivel acumulado, se hace referencia a los metros ganados o ascendidos, y el desnivel perdido, indica los metros que el atleta “perdió” durante el entrenamiento.

Las variables relacionadas a tiempo y distancia son:
- Distancia total
- Tiempo total
- Desnivel acumulado
- Desnivel perdido
- Pendiente (%) (es el desnivel acumulado dividido entre la distancia recorrida x 100)

**Potencia:** constituye uno de los indicadores clave de la intensidad del entrenamiento. Es el cociente entre el trabajo realizado (Joules) dividido entre el tiempo que implicó realizar el mismo (1000 J/10 s = 100 W).

Se utiliza para controlar la intensidad del entrenamiento en ciclismo y triatlón, y nuevas tecnologías wearables, están introduciendo el entrenamiento por potencia también en el running con el mencionado dispositivo Stryd. La enorme utilidad de la potencia radica en que permite realizar valoraciones en el campo, e incluso construir un perfil de rendimiento del atleta en base a entrenamientos y competiciones.

Se debe medir utilizando potenciómetros o powermeters, los más conocidos son SRM y Powertap; el primero es demasiado caro para que esté accesible para el atleta amateur, el Powertap, no obstante, es accesible, ya que por 750 USD cualquiera puede hacerse de uno; están apareciendo de muchas marcas Quark,
Garmin (va en los pedales y es costoso), Stages (es una palanca), iAero (estima la potencia en base a varias variables), el dispositivo Stryd es específico para el running.

Las variables relacionadas a la potencia son:
Potencia media
Potencia máxima
Mean Maximal Power o potencia media máxima (para diferentes tiempos, 5 s, 30 s, 1 min, 2 min, 3 min, 5 min, 10 min, 20 min, 30 min, 60 min)
NP (potencia normalizada)
IF (NP/FTP)
TSS (IF x IF x 100 x tiempo (horas))
Chart de potencia vs. tiempo (ideal con la posibilidad de hacer zoom en diferentes secciones)
Time in zones (en zonas predeterminadas o una individualizada por el coach)

**Cadencia:** consiste en el número de ciclos por minuto del gesto de los deportes de resistencia. En el ciclismo se habla de revoluciones (una vuelta completa de pedal) por minuto; en la carrera, se habla de frecuencia de pasos por minuto. Pero en la natación, no se habla de cadencia, sino de frecuencia de brazadas (el número de brazadas por minuto).

Se la utiliza analizando simultáneamente otras variables como potencia y FC (frecuencia cardíaca), y permite determinar así, si el atleta entra en fatiga, o si tiene limitaciones técnicas. La miden los potenciómetros, y dispositivos GPS que incluyen en el caso del ciclismo sensores de cadencia. Por lo general, se mide cadencia media, máxima, y distribución de cadencia en el tiempo.

**Frecuencia cardíaca:** es el número de veces que el corazón late en un minuto. Es uno de los indicadores más utilizados por todos los deportistas de diferente nivel para controlar la intensidad. Tiene, no obstante, muchas limitaciones que deben ser tenidas en cuenta, como diferentes factores que incluyen la altura, el calor,
enfermedad, ansiedad, las horas de ejercicio, deshidratación, etcétera; que pueden modificarla. Los dispositivos GPS, la miden utilizando una banda que el atleta usa en el pecho; otros, la miden también desde la muñeca.

Las variables relacionadas a la frecuencia cardíaca son:
FC media
FC máx.
Mean Maxima FC o Frecuencia Media Máx. (para diferentes tiempos, que son similares a los de la potencia)
Time in Zones (en zonas predeterminadas e individualizadas por el coach)
Chart de FC vs. tiempo (ideal con la posibilidad de hacer zoom en diferentes secciones)

**Temperatura**: puede tener aplicaciones en relación, sobre todo, a los atletas que compiten en el calor, y deben entrenar en esas condiciones. Es interesante en el caso del agua, ya que puede permitir buscar condiciones específicas a las que nadará el atleta en aguas abiertas. Por lo general, se mide temperatura media y distribución de la temperatura en el tiempo.

**Consumo de oxígeno**: es la cantidad de oxígeno que el organismo utiliza por minuto. El VO2 máx., o máximo consumo de oxígeno, constituye un indicador clave del rendimiento, y determinante del mismo en todos los deportes de resistencia, por ende, se utiliza para diagnosticar el nivel de rendimiento del atleta.

La desventaja del VO2 máx. es que para medirlo se requieren equipos costosos y recursos humanos capacitados para utilizarlos. Equipos accesibles como dispositivos GPS realizan una estimación del mismo, y es posible generar aproximaciones a partir de diferentes tests, utilizando ecuaciones. En las valoraciones realizadas en laboratorios, se puede ver el VO2 en el tiempo, y luego el valor máximo (VO2 máx.) que tiene ciertos criterios para ser definido.
**Lactato en sangre**: el lactato es un metabolito de la vía glucolítica, la misma constituye uno de los sistemas de producción de energía del organismo. El lactato es un muy buen indicador de la intensidad del entrenamiento, y puede complementarse a la frecuencia cardíaca, potencia y cadencia.

Se mide con dispositivos portables tales como el Lactate Plus o Lactate Pro, que miden su concentración en sangre capilar con una pequeña muestra de sangre (5 micro litros). Existen dispositivos que lo estiman en base a una medición infrarroja pero se desconocen publicaciones científicas que validen la precisión de las lecturas.

**Rendimiento y ciclo de mejora continua**

Dosificado el entrenamiento, realizado por el atleta con un seguimiento controlado por las mediciones de los sensores de los wearables, hay que verificar los resultados obtenidos para tal estímulo; para lo cual, se realizan diferentes test de valoración de rendimiento. Los entrenadores con su experiencia, determinan la periodicidad de realización acorde a la planificación macro del entrenamiento, y las fechas de competencia con las que se guía el entrenamiento.

![Ciclo de mejora continua de Deming](image)

**Figura 4**: Ciclo de mejora continua de Deming.

**Fuente**: (Garcia, Quispe y Ráez, 2003)
El accionar profesional del entrenador de deportistas de rendimiento, es cíclico, desde la planificación del entrenamiento con dosificaciones de carga, intensidad y periodización de las mismas, medición de la actividad física del atleta, verificación de resultados por medio de valoraciones de rendimiento, y accionar correctivo y/o perfeccionativo del plan de entrenamiento. Claramente aplica el ciclo de mejora continua de Deming popularmente conocido como el padre de la calidad tal como se visualiza en la Figura 4.

Resumen del capítulo
Los datos son de primordial importancia crítica para poder medir que el entrenamiento dosificado al atleta, sea efectivamente el que está realizando. Los resultados obtenidos se deben verificar para poder realizar las correcciones necesarias. En este contexto, los wearables tienen aplicación en la medición de datos, quedando claramente fundamentado el valor agregado que aportan estos dispositivos en la ciencia de movimiento presentada, por lo que el objetivo perseguido en el presente trabajo queda contextualizado y justificado.
Marco Tecnológico

Introducción al desarrollo de wearables
El desarrollo en plataformas “wearables” es todo un nuevo desafío, los usuarios exigen experiencias e interfaces de uso dinámicas, comparables a lo que están acostumbrados a acceder en sus smartphones o páginas webs responsivas, visualizadas en pequeñas pantallas. Sin embargo; el limitante del consumo energético impone que si la app es muy dinámica, la batería se consume rápidamente, más agravante aún, es que cuando el dispositivo se está cargando, implica que el usuario definitivamente no está usando la aplicación, a diferencia de lo que ocurre con un celular o una tablet.

La heterogeneidad de capacidades y especificaciones de hardware de dispositivos wearables, incluso, cuando los mismos pertenecen al mismo fabricante y a la misma línea de productos, hacen que la labor de programación, tanto en código como diseño, sea mucho más desafiante. Por ejemplo, la misma app, con el mismo diseño y experiencia de usuario, con el mismo código fuente, se debe ejecutar tanto en un reloj de pantalla redonda y en otro de pantalla cuadrada; en uno que tiene touch screens, como en otro que se usa con botones, y no todos equipados con los mismos sensores, ya que la implementación de éstos, dependerá del propósito del dispositivo; un smartwatch para natación es considerablemente distinto que uno para carrera.

Es necesario para desarrollar aplicaciones para dispositivos de cualquier índole, inclusive wearables, que se disponga del kit de desarrollo de software o “SDK”, del inglés Software Development Kit, es éste quién provee todas las herramientas para poder programar aplicaciones propias que se ejecuten en los dispositivos, esto implica desde el lenguaje de programación, hasta intérpretes y compiladores del código. Por otro lado, tenemos el IDE (Integrated Development Environment; en español, Entorno de Desarrollo Integrado o Entorno de Desarrollo Interactivo), que son plataformas de software que proporcionan servicios integrales que
facilitan el desarrollo de software al programador (Ramos y Lozano, 2000). Entonces se resume que el SDK es indispensable y que debe ser provisto por el fabricante del dispositivo, o en su defecto, por el fabricante de su sistema operativo, en cambio el IDE no es indispensable pero sí deseado por ser un facilitador en el proceso de desarrollo de software.

Cómo se ve una app en un wearable: UX Guidelines para wearables

En el diseño de wearables, hay que tomar en cuenta aspectos asociados a los diferentes tipos de pantallas: rectangulares, redondas, semi-redondas, etcétera; diferente resolución; diferentes tipos y cantidad de botones disponibles; posibilidad de touchscreen, cuando tienen, la superficie de contacto es limitada; recomendaciones de Layout de datos, acorde a cantidad de campos a mostrar y dispositivos.

Así entonces, el diseño de wearables puede ser considerado por muchos, como un desafío. Al respecto, FJORD (2017) sugiere cinco principios en la creación de un diseño para tecnología portátil.

1. Balance público y personal: los wearables son usados para expresar públicamente el sentido de la moda y el estilo del usuario, pero al mismo tiempo, pueden revelar datos muy personales, como conversaciones, relaciones, e incluso la salud. A diferencia de los teléfonos inteligentes que se pueden ocultar en la privacidad de los bolsillos, llevar encima la tecnología, puede hacer que estos dispositivos sean aún, más íntimos y públicos. En función de ello, se debe tener en cuenta la consideración, entre públicos y personales. Se sugiere atender la manera en la que se aborda inicialmente el dispositivo y el contenido que deberá tener la pantalla en consecuencia: vibración e imagen. El usuario deberá tener el control, pero con valores predeterminados considerados.
2. Que sea glanceable: generalmente el espacio de la pantalla es extremadamente limitado, por lo que, la exposición de los usuarios a la menor cantidad de datos posible para ayudarles a alcanzar sus objetivos, es la clave. Diseños sobrecargados con información detallada requieren demasiada atención, puede distraer a los usuarios o competir con su contexto social. Se sugiere que cada pantalla muestre una información a la vez y con el mayor nivel de sencillez.

3. Las reglas del juego no visual: las pantallas portátiles son mucho más pequeñas de lo que usualmente se acostumbra. Sin embargo, de la limitación viene la oportunidad. Para compensar su falta de tamaño, a través de las interacciones basadas en visuales, muchos dispositivos portátiles ofrecen nuevas y diferentes formas de entrada y recepción de información. El reconocimiento de gestos, los patrones de golpeteo, datos de salud y comunicación vibratoria, son algunas de las capacidades que apuntan al futuro de nuevas interacciones. Por lo tanto, es conveniente el uso de nuevas entradas, como gestos, la voz, y el golpeteo; notificar con matices de comentarios, vibraciones y sonidos.

4. Cuidado con la avalancha de datos: hoy día, se tiene acceso ilimitado a los datos y las comunicaciones, por lo que, priorizar es esencial. Hay que centrarse en ofrecer sólo la información más importante cuando el usuario la necesite, creando experiencias que apoyan en lugar de abrumar. El objetivo es crear un sistema que preste apoyo a tareas específicas y que permita interactuar con los usuarios sólo en los momentos adecuados, a decisión propia.

5. Atención a las lagunas: la tecnología wearable, representa la más reciente adición a la vida digital del usuario, y al igual que otros dispositivos, van a experimentar problemas de conectividad en sumatoria al reto de su integración con los dispositivos existentes en ese ecosistema. Por lo tanto, los diseñadores tendrán que esforzarse por proporcionar modos fuera de línea útiles y transiciones sin fricción entre wearables. Se sugiere
entonces, siempre proporcionar la funcionalidad de núcleo en el modo fuera de línea y hacer transiciones imperceptibles cuando la gente cambie de dispositivos.

**Cómo se comunican los wearables: ANT+**

En el proceso de creación de un producto portátil, es imprescindible considerar la tecnología inalámbrica a elegir. La forma en que funcionan las API modernas, sugieren que los desarrolladores no necesariamente necesitan conocer los detalles técnicos; pero tener una comprensión básica de la tecnología subyacente, les ayudará a hacer aplicaciones que sean más eficientes, más receptivas y más fáciles de usar.

Para establecer la comunicación de un wearable, la mejor opción dependerá de los requisitos particulares de ancho de banda, consideraciones de potencia y alcance. Las seis tecnologías inalámbricas más comunes a considerar, son: comunicación de campo cercano (NFC), Bluetooth Baja Energía (BLE), ANT, bluetooth classic (originalmente llamado bluetooth), wifi y celular. Algunos wearables combinan diferentes tecnologías inalámbricas para aprovechar los diferentes aspectos de cada método.

Según afirma, Soderholm (2016), la tecnología de red inalámbrica ANT es una buena opción para los usuarios en el campo deportivo. Como una filial de Garmin, ANT es enorme en el segmento de deportes y fitness, donde se utiliza típicamente para el control de la frecuencia cardíaca, la potencia en el ciclismo, la distancia y la velocidad. Muchas de las marcas actuales de ciclismo y fitness utilizan ANT para sus wearables. De hecho, casi se ha convertido en un estándar en esta área. En ANT, los sensores y otros nodos actuarán como esclavos o maestros dentro de una red inalámbrica. Cada nodo puede transmitir, recibir o incluso funcionar como un repetidor para aumentar el alcance de la red, permitiendo que ésta sea configurada para pasar largos períodos en el modo de suspensión de bajo consumo, gastando muy poca energía.
Comunicación ANT+ vs. Bluetooth

Según Jepson (2017), ANT y Bluetooth Low Energy (BLE) pueden parecer normas competidoras, pero cada una tiene sus puntos fuertes en diferentes aplicaciones. En ANT, cada nodo tiene capacidades iguales, mientras que en BLE, las redes son asimétricas, con un enfoque basado en el hub (a menudo con su teléfono o computadora en el centro).

Debido a que ANT es un modelo simétrico, los requisitos para una red son simplificados. Puede tener varios nodos con capacidades computacionales relativamente bajos y similares entre ellos, mientras que BLE requiere un dispositivo de concentrador con una potencia computacional significativa, propio de un modelo de hub.

BLE utiliza la red tipo hub a partir de un modelo de red en estrella con un dispositivo concentrador / maestro en el centro, que coordina cada uno de los otros dispositivos de esa red. ANT es capaz de acomodar ese modelo, pero también incluye redes de malla. Esto significa que ANT es capaz de escalar significativamente en cantidad de dispositivos, algo muy útil para que se utilicen más sensores a la vez, permitiendo también a dichos sensores hablar entre sí directamente, sin necesidad de comunicarse primero con el concentrador.

Aunque BLE teóricamente no tiene límite en el número de dispositivos que pueden participar en una red, existen limitaciones específicas de la implementación. Por ejemplo, Android 4.4 está limitado a 7 conexiones simultáneas, y 5.0 le permite subir hasta 15.

Si bien hay muchos casos de uso superpuestos entre BLE y ANT, esta última es particularmente idónea para soluciones que contienen gran cantidad y heterogeneidad de sensores, algo muy frecuente en sistemas para la salud, entrenamiento y basados en wearables.
Por lo tanto, un claro ejemplo de diferenciación de utilizar ANT, es en equipamiento para ciclismo, donde se cuentan con un gran número de sensores de potencia, cadencia, frecuencia cardíaca, temperatura, y otros. Otro caso sería aplicado a deportes en conjunto, por ejemplo el preparador físico comparando datos de frecuencia cardíaca de todos los jugadores de su equipo de fútbol, cada uno de ellos utilizando una banda sensorial de frecuencia cardíaca compatible con ANT.

**Qué datos generan los wearables: Estructuras de datos**

Las mediciones que los wearables vayan realizando sobre el deportista, deberán persistirse o grabarse, poder transferirse entre dispositivos de diferentes índoles, y ser interpretadas por los destinatarios. Para ello, es necesario disponer de estructuras de datos definidas y/o protocolos para poder codificar, decodificar y manipular las mediciones para generar la información necesaria.

Los fabricantes de hardware pueden optar por definir sus propios protocolos y estructuras de datos, procurar imponerlos como estándares del mercado si es que no existieren, o bien adoptar los que ya estén estandarizados para el mercado. Al momento de optar por desarrollar un software, es importante priorizar el uso de estándares fundamentados en una gran cantidad de ventajas, entre las que destacamos la reducción de la dependencia del fabricante de hardware, mayor compatibilidad multi-dispositivo, facilidad de mantenimiento y escalabilidad.

Por un lado, se usarán estándares, por el otro, en este momento de definición y durante todo el ciclo de desarrollo, toma un papel preponderante la heterogeneidad de capacidades que tienen los diferentes dispositivos. No poseen los mismos sensores un smartwatch para natación, que otro enfocado para corredores de maratón; mucho más distante es aún en el ciclismo, donde el dispositivo tiene gran capacidad computacional. Por dicha variedad, es que existen diferentes estándares.
Para abordar la diversidad de dispositivos, se hará foco en los dos estándares principales en la industria, señalados por Dynastream Innovations Inc. (2017). Primero el FIT que está diseñado para trabajar con wearables de pocos recursos computacionales como smartwatches (natación y carrera), por lo que es binario para eficientizar al máximo, el consumo de procesamiento y almacenamiento; además de ser flexible, para manipular diferentes tipos de mensajes como se detallará en los próximos apartados. Segundo, el TCX que está orientado a dispositivos de mayor capacidad como minicomputadoras (ciclismo), por lo que la sencillez de interoperabilidad es priorizada sobre la eficiencia de recursos, lo que justifica que sea basado en XML. Por último se mencionará el GPX, ya que si bien su campo de aplicación supera los de este trabajo, es conocido y mencionado en la industria del ciclismo.

**FIT - Transferencia de datos flexible e interoperable**
Este protocolo está diseñado específicamente para el almacenamiento e intercambio de datos que se originan en los dispositivos de deporte, fitness y salud. El protocolo FIT define un conjunto de plantillas de almacenamiento de datos (mensajes FIT) que se pueden utilizar para almacenar información, tal como perfiles de usuario, y los datos de actividad en los ficheros. Está específicamente diseñado para ser compacto, interoperable y extensible.

ANT-FS proporciona un marco sólido para la transferencia de archivos de forma inalámbrica entre dos dispositivos. Tal como se presentó anteriormente en la sección “Cómo se comunican los wearables”. La Figura 5, ilustra el protocolo FIT que se utiliza para transferir información personal de monitorización adquirida durante el ejercicio, para una base de datos de Internet.
El protocolo FIT define el proceso para que los archivos se transfieran. Normalmente, los datos de difusión se recogen mediante un dispositivo de visualización. El dispositivo de visualización, entonces, codificaría los datos en el formato de archivo FIT de acuerdo a su perfil de producto. El archivo FIT se transfiere asimismo a otro dispositivo, que a continuación decodificaría los archivos recibidos de acuerdo a su propio perfil de producto.

Los sensores ANT+ recogen parámetros de medida, tales como la frecuencia cardíaca y la velocidad de carrera. Los datos se transmiten en tiempo real, utilizando formatos de datos interoperables ANT+. Los eventos de sesión y los datos de la actividad en tiempo real, se guardan en un archivo que se muestra en un dispositivo de visualización.

De la misma forma, el archivo FIT se transfiere a la PC a través de recursos compartidos de archivos ANT (ANT-FS). Acá la opción de integración más completa, es que la transferencia se haga a un smartphone, y desde el mismo, a un servidor (PC) para ser utilizados por las aplicaciones de internet. El protocolo FIT ofrece un formato coherente, permitiendo que todos los dispositivos posteriores puedan compartir y utilizar los datos.
**Perfiles de dispositivos ANT+**

Cada dispositivo tiene un perfil que especifica los parámetros de canal y otros requisitos mínimos para la interoperabilidad.

![Figura 6: Listado de perfiles FIT estándares. Fuente: (Dynastream Innovations Inc., 2017c).](image)

Como ejemplo, se hace referencia al monitor de frecuencia cardíaca. Los monitores de frecuencia cardíaca ANT+, se utilizan principalmente para medir el ritmo cardíaco del usuario (latidos por minuto - ppm) durante una determinada actividad en tiempo real.

El monitor de frecuencia cardíaca ANT+ está diseñado para funcionar como un sensor de difusión. Este tipo de sensor reduce la complejidad de la red, simplifica la interacción del usuario con el dispositivo, y reduce los requisitos de la batería. Un sensor de difusión utiliza una topología de red de uno a muchos. Esto permite que múltiples dispositivos de visualización puedan recibir datos de frecuencia cardíaca desde el mismo monitor del ritmo cardíaco. Para lograr este dispositivo, cada pantalla debe ser configurada para recibir desde el monitor de frecuencia cardíaca.

**Estructura de los archivos FIT**

Un archivo FIT está compuesto de uno o varios registros, cada registro, a su vez, contiene un encabezado (indica el tipo del mensaje) y el contenido (datos). Todos los archivos FIT tienen la misma estructura que consiste en un encabezado de archivo, una sección principal de registros de datos que contiene los mensajes FIT codificados, seguido de un CRC de 2 bytes, tal como se aprecia en la Figura 7.
Es importante señalar lo siguiente:

- El encabezado del archivo proporciona información sobre el archivo FIT.
- Los registros de datos en el archivo FIT son el contenido principal y el propósito del protocolo FIT.

Por otra parte, hay dos tipos de registros de datos:

- Mensajes de definición: tal como lo indica su nombre, precisa los próximos mensajes de datos; específicamente, definirá un tipo de mensaje local y lo asociará a un mensaje FIT específico.
- Mensaje de datos: campos de datos poblados en el formato descrito por el mensaje de definición anterior. El mensaje de definición y sus mensajes de datos asociados tendrán tipos de mensajes locales coincidentes.

Asimismo, todos los mensajes de datos se gestionan localmente y los mensajes de definición se utilizan para asociar tipos de mensajes de datos locales al perfil de mensajes FIT global (Figura 8). Por ejemplo, un mensaje de definición puede especificar que los mensajes de datos de tipo de mensaje local 0 sean mensajes de "vuelta" FIT globales. El mensaje de definición también especifica, cuáles de los campos de "vuelta" se incluyen en los mensajes de datos (start_time, start_position_lat, start_position_long, end_position_lat, end_position_long) y el formato de los datos en esos campos. Como resultado, los mensajes de datos
pueden optimizarse para contener sólo datos y el tipo de mensaje local se hace referencia en el encabezado.

Figura 8: Archivo FIT y Perfil FIT Global. 
Fuente: (Dynastream Innovations Inc., 2017a).

Perfiles FIT: global y de producto
El perfil global tiene la colección completa de mensajes FIT, configuraciones del sistema, campos y datos. Por su parte, el perfil de producto es un subconjunto específico de aplicaciones del perfil global que sólo tiene los mensajes y datos necesarios de acuerdo a la arquitectura del producto. Dos dispositivos FIT diferentes pueden utilizar diferentes perfiles de producto o versiones del perfil global completo. Esto puede hacer que un dispositivo reciba un mensaje FIT que no reconoce. Cuando esto ocurre, el archivo FIT se mantiene en su totalidad y cualquier mensaje no reconocido es simplemente ignorado por el decodificador sin interrumpir el funcionamiento del dispositivo receptor o causar errores.

Del mismo modo, si un dispositivo no recibe datos que puede esperar, simplemente llenará esos campos con un valor no válido, en lugar de crear errores. De esta manera, el protocolo FIT garantizará la compatibilidad entre los dispositivos que pueden no tener exactamente los mismos perfiles implementados. Estos procesos de compatibilidad se analizan con más detalle en secciones posteriores, los diferentes perfiles se pueden ver en la Figura 9.
Protocolo de archivo FIT

Los datos entrantes tales como ajustes, eventos y datos de sensores se escriben en campos de mensaje FIT de acuerdo con los formatos definidos. El proceso de codificación FIT está optimizado, de tal manera que sólo se escriben campos válidos en el archivo. El archivo se puede transferir a otro dispositivo FIT. Cuando los datos son utilizados por el dispositivo receptor, se decodifica, para eso se relacionan los mensajes FIT recibidos con la lista global de mensajes FIT. Los valores decodificados se pasarán entonces como estructuras u objetos a la aplicación. Este circuito y flujo de información se presenta visualmente en la Figura 10.

Si hay una diferencia en la versión de perfil entre los dos dispositivos, los datos que faltan se establecerán en valores no válidos o predeterminados como se definen en el protocolo FIT y cualquier mensaje o datos desconocidos se ignorarán. El archivo FIT se mantiene en su forma original para su transferencia a otros dispositivos, si lo desea (Dynastream Innovations Inc., 2017a).
**TCX - Training Center XML**

Los archivos TCX se utilizan para el intercambio de datos; implementados por el software Garmin Training Center, están basados en XML (W3C, 2008). Estos archivos tratan a cada pista como una actividad, permitiendo guardar más que una serie de puntos; también almacenan datos de fitness, como el tipo de deporte, tiempo de vuelta, la distancia recorrida, calorías, frecuencia cardíaca, cadencia, entre otros. El ejemplo de fracción del esquema de un TCX, puede ser visualizado en la Figura 11.
Figura 11: Ejemplo de fragmento del esquema TCX.

Fuente: Garmin (2017c).
**Esquema de un TCX**

El Esquema XML es un lenguaje de esquema basado en XML utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML. O sea, un esquema XML es quien verifica si un mensaje XML entrante o saliente es válido acorde a lo que se espera que tenga su contenido.

El esquema TCX entonces será de gran utilidad y aplicación, para conocer qué datos puede contener cada mensaje TCX, cuáles de ellos son opciones u obligatorios, cómo se les denomina a cada uno, qué tipo de valores puede tener, etcétera. En el marco del desarrollo de software, en la fase de análisis y diseño, el esquema XML se utiliza para definir cómo serán los mensajes XML a intercambiar entre dispositivos. En la fase de programación, tener el esquema XML definido, otorga gran facilidad para corroborar si los mensajes recibidos son válidos. En este caso, para corroborar si un mensaje TCX recibido es válido, se lo contrarresta contra el esquema en cuestión. El esquema total tiene casi 900 líneas, en la Figura 11 se presenta sólo un fragmento del mismo.

**Estructura de un plan de entrenamiento. Elementos que contiene**

La estructura de un plan de entrenamiento puede ser ampliada, agregando elementos de otros esquemas. Sin embargo, pueden mencionarse, los siguientes elementos (Garmin, 2017d):

- Carpetas, con entrenamientos, circuitos y historiales.
- Actividades.
- Entrenamiento. Incluye los pasos.
- Circuito.
- Historial. Incluye carpetas de running, bicicleta y otros deportes.
- Carpetas con deportes múltiples. Indica qué semana, y qué deportes se realizan.
- Semanas, con día de inicio.
- Múltiples deportes por sesión. El primer deporte y el próximo deporte.
- Primer deporte. Indica la actividad.
- Próximo deporte. Contiene la transición y la actividad.
- Deporte. Puede ser running, bicicleta u otro.
- Actividad. Creación, vuelta de actividad, deporte, notas.
- Dispositivo. Contiene producto y versión.
- Aplicación. Contiene fecha de creación, lenguaje y número de serie.
- Ejercicio rápido. Tiempo total en segundos y distancia en metros.
- Plan. nombre, tipo de plan (circuito, entrenamiento), entrenamiento intervalado.
- Vuelta de actividad. Incluye tiempo total en segundos, distancia en metros, máxima velocidad, calorías, intensidad, cadencia, puntos, y notas; como extensión se le agrega la potencia.
- Punto. Contiene tiempo, posición (latitud y longitud), altura en metros, distancia en metros, cadencia.
- Repeticiones. Cantidad mínima y cantidad máxima.
- Pasos. Nombre, duración, intensidad.

**GPX – Formato de intercambio GPS**

GPX, es un esquema XML diseñado como un común formato GPS. Puede ser utilizado para describir los puntos, pistas y rutas. Los datos de ubicación (y opcionalmente la elevación, el tiempo, y otra información) se almacena en las etiquetas y se pueden intercambiar entre los dispositivos GPS y software.

**Tipos de datos**

Se distinguen los siguientes tipos de datos (GPX, s.f.):

- **wptType**. Es un punto entre una colección de puntos sin relación secuencial. Consiste en el WGS 84 (GPS) coordenadas de un punto y posiblemente otra información descriptiva.
- **trkType**. Es una pista, hecho de puntos que contienen al menos un segmento, es decir, una lista ordenada de puntos que describen un camino.
Un segmento de pista contiene una lista de puntos de seguimiento que están conectados lógicamente en orden.

- **rteType.** Es una ruta, una lista ordenada de punto de rutas que conducen a un destino.

Las propiedades mínimas para un archivo GPX, son latitud y longitud de cada punto. Todos los demás elementos son opcionales. Algunos proveedores, como Humminbird y Garmin, usan las extensiones del formato GPX para el registro de dirección, número de teléfono, la temperatura del aire, la profundidad del agua, y otros parámetros. Otras propiedades del esquema GPX se pueden visualizar en la Figura 12.

**Esquema de un GPX**

De igual manera que el esquema para TCX que se especificó anteriormente, el esquema para los archivos GPX detalla el contenido obligatorio, opcional, y estructura válida de un mensaje GPX. Ejemplo de fracción del esquema de un GPX, puede ser visualizado en la Figura 12. A groso modo, los datos más importantes que se detallan en el esquema son:

- Punto
- Pista
- Ruta
- Email
- Persona. Nombre de persona u organización
- Link a recurso externo
- Latitud
- Longitud
- Otras extensiones
- Proximidad
- Temperatura
- Profundidad
- Dirección (calle, numero, ciudad, estado, código postal)
- Número de teléfono
- Distancia en metros

Figura 12: Ejemplo de fragmento del esquema GPX.
Fuente: Garmin (2017e)

Opciones de hardware: Marcas Principales
En el proceso de selección de dispositivos donde realizar la implementación técnica, se evaluaron las marcas más importantes del mercado. Entre ellas se pueden mencionar las empresas entre las específicas del sector: Garmin, Polar, Suunto, Tomtom, Apple Watch, Samsung Smartwatches y Microsoft Band, las tradicionales del sector tecnológico con amplia presencia en el mercado de smartphones: Apple, Samsung y Microsoft; y por último, las referentes en
manufactura de relojes de precisión y lujo, como Tag Heuer que tiene una línea de smartwatches.

**Garmin**

Garmin, es la empresa líder mundial en navegación por satélite, dedicada al desarrollo de artículos GPS, como también de dispositivos conectados para aquellos usuarios que les gusta hacer deporte al aire libre, que permiten medir del rendimiento de su actividad deportiva.

Garmin fabrica dispositivos como pulsómetros, cámaras y minicomputadoras para todo tipo de actividades: automovilismo, náutica, aviación, ciclismo, running, natación, golf, excursionismo, multideporte y mascotas. Cada dispositivo posee un perfil diferente, de acuerdo al deporte para que se use. Pero el 15 de marzo, Garmin lanzo su nueva línea de dispositivos Fénix, que poseen un perfil centralizado y se adaptan a cualquier tipo de deportes, midiendo las variables principales de ellos.

Los dispositivos cuentan con Garmin Connect, una herramienta de entrenamiento en línea que te permite guardar, analizar y compartir todas tus actividades deportivas, conectando tu smartphone con el dispositivo Garmin. Garmin también permite que otras aplicaciones puedan interactuar con sus dispositivos, a través de Connect IQ (herramienta para desarrolladores), estas aplicaciones además de mostrar información de los dispositivos, podrán interactuar con ellos (Garmin, 2017b).

**Polar**

Empresa líder en innovaciones tecnológicas y en monitores de frecuencia cardíaca, que ofrece una amplia gama de productos en formato de pulsómetros, que le dan al atleta información precisa de su rendimiento durante el entrenamiento. Los pulsómetros se pueden sincronizar con Polar Beat, la
aplicación que permite al atleta analizar su entrenamiento y realizar un seguimiento de sus progresos a través de su Smartphone o de la web (Polar Flow).

Polar, permite que además otras aplicaciones de terceros puedan obtener información de los dispositivos Polar a través de AccessLink (herramienta para desarrolladores) (Polar, 2015).

**Suunto**
Empresa que diseña relojes, brújulas e instrumentos deportivos, para escalada, caza, pesca, ciclismo, fitness, running, natación, triatlón, entre otros; pero se destaca en la creación de dispositivos de buceo de calidad superior. Las aplicaciones que usan los dispositivos Suunto para compartir sus datos a través del Smartphone del atleta son Suunto Movescount, compatible con alguno de los dispositivos (Suunto, 2017).

**Fitbit**
Empresa destinada a crear productos usados para la medición del rendimiento del entrenamiento del atleta, como smartwatchches, wereables. Fitbit posee una aplicación propia, para compartir los datos del entrenamiento, permitiendo al atleta, obtener un análisis y seguimiento de sus datos. También posee una aplicación para desarrolladores, de manera que aplicaciones externas puedan obtener los datos de los dispositivos Fitbit (Fitbit, 2017).

**Tomtom**
Desarrolla pulsómetros, cámaras, para medir el rendimiento deportivo del atleta en fitness y golf. Para almacenar los datos del entrenamiento emplea, Sports App, que permite manipular y procesar los datos obtenidos del dispositivo vinculado. Además, Tomtom posee una API de geolocalización que consiente que
aplicaciones terceras recopilen datos del dispositivo Tomton, de su ubicación y parámetros geográficos (TomTom, 2017).

**Apple Watch**
Apple desarrolla relojes inteligentes que permiten medir y guiar el entrenamiento del usuario, sin importar el deporte que realice, además de contar con sensor óptico de frecuencia cardíaca, y los últimos modelos con sensor GPS integrado.

**Samsung Smartwatches**
Diseña pulseras inteligentes que además de medir frecuencia cardíaca, ritmo, guían al usuario en su entrenamiento, gran variedad de modelos que lideran la oferta en mercado a usuarios de celulares con sistema operativo Android (Samsung, 2016).

**Microsoft Band**
Pulsera inteligente con integración optimizada a celulares con sistema operativo Windows Phone, posee todas las funcionalidades esperadas de esta clase de dispositivos, sensor de frecuencia cardíaca, registro de actividades y medidor de calidad de sueño. (Microsoft, 2017). Las herramientas para programadores tienen soporte discontinuado desde Microsoft (Microsoft, 2017b).

**Selección de hardware: Garmin**
Para la selección del hardware a utilizar se planteó una matriz que valora los criterios que se consideran importantes y que deben ser tomados en cuenta para la toma de decisión. Tales criterios son los siguientes:

- Integración: capacidad de tomar las lecturas que hacen los dispositivos.
- SDK: entorno para desarrollar aplicaciones propias para el dispositivo.
- Hardware: variedad de equipos que se fabrican y comercializan.
Mercado: volumen de usuarios.

En función de éstos, se establece la respectiva valoración: 1 (poca/baja), 2 (regular), 3 (normal), 4 (muy bueno), 5 (excelente), – (en caso de si no aplica o no provee). Finalmente, se totalizan los datos, en términos de las marcas anteriormente expuestas.

| Matriz preponderada de fabricantes de wearables y características. |

<table>
<thead>
<tr>
<th></th>
<th>Integración</th>
<th>SDK</th>
<th>Hardware</th>
<th>Mercado</th>
<th>-</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suunto</td>
<td>1</td>
<td>-</td>
<td>4</td>
<td>3</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Garmin</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>Polar</td>
<td>3</td>
<td>-</td>
<td>5</td>
<td>4</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Fitbit</td>
<td>3</td>
<td>-</td>
<td>4</td>
<td>5</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Apple</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Samsung</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>Microsoft</td>
<td>1</td>
<td>-</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

Fuente: Elaboración propia.

Introducción: Garmin

Garmin es la línea de hardware seleccionada, ya que se impuso claramente sobre las alternativas. Tiene la línea de dispositivos preferida por los entrenadores, el 74% de entrenadores que recomiendan a sus atletas la utilización de wearables, lo hacen con la línea de productos Garmin (TrainingPeaks, 2016). Si bien la adopción en el mercado es un factor importante, la principal razón por la cual se define la selección, es que Garmin es el único que posee un ecosistema para desarrolladores maduro (Garmin, 2017a).

Un ecosistema para desarrolladores maduro, implica principalmente la posibilidad de desarrollar aplicaciones propias y personalizadas para diferentes gamas dispositivos, para lo cual, se utiliza un lenguaje de programación propio Monkey-C (que será detallado, en páginas sucesivas). Otro factor relevante, es el
ecosistema ANT, con el cual se extienden las posibilidades de conectividad a un sinfín de integraciones (como se verá más adelante), el acceso a APIs de las plataformas de software que ofrecen, y la comunidad de programadores.

Las posibilidades de desarrollo mencionadas y demás componentes del ecosistema, se pueden apreciar en la Figura 13, “Círculo de experiencia de desarrollo Garmin”.

![Figura 13: Círculo de experiencia de desarrollo Garmin. Fuente: Garmin (2017).](image)

Plataforma de desarrolladores GARMIN

Garmin Connect

Garmin Connect es una herramienta de entrenamiento en línea que permite guardar, analizar y compartir todas las actividades deportivas del usuario, sólo son necesarios algunos pasos para configurar el dispositivo y la cuenta respectiva.

Garmin Connect es la única comunidad en línea creada específicamente para la mayoría de los dispositivos de acondicionamiento físico desarrollados por Garmin y que funciona perfectamente con ella. Garmin Connect almacena y analiza datos de usuarios finales y admite más de treinta tipos de actividades diferentes,
incluyendo: running, ciclismo, natación, excursionismo, entrenamiento de fuerza, caminata, entre otros. Como características específicas de esta aplicación, se señalan (Garmin, 2017c):

- Almacenamiento y análisis de múltiples puntos de datos del usuario final, entre ellos: velocidad, distancia, paso, calorías, cadencia, poder, GPS, hora.
- Acceso potencial a los datos de millones de usuarios finales en todo el mundo.
- El acceso a los archivos de datos completos con datos a nivel de segundo y minuto, casi en tiempo real, arrojados directamente al servidor del desarrollador (es responsabilidad desarrollador de tener servidores accesibles para la recepción de datos).

La API de Garmin Connect, ofrece a los desarrolladores de terceras partes autorizadas el acceso a este conjunto de datos robustos con el consentimiento del usuario final. Además del acceso tradicional por medio de la API, Garmin también ofrece un servicio de sincronización de sus servidores con los del desarrollador, poniendo a disposición de éste, el archivo original de datos (FIT, GPX o TCX, dependiendo del dispositivo) a través del método PUSH. Así entonces, con la API de conexión, las posibilidades de innovación son infinitas. A continuación, algunos ejemplos de lo que se puede hacer con la aplicación:

- Automatizar el seguimiento por instructores/entrenadores.
- Proporcionar análisis detallado actividad para mejorar el rendimiento.
- Crear gráficos interactivos para seguir el progreso.
- Proporcionar planes de formación a medida.
- Crear fuentes de actividad de redes sociales.
- Proporcionar almacenamiento de datos en los teléfonos inteligentes.

**Garmin Connect IQ**

Connect IQ, según afirman Plana y Garrido (2015), es una plataforma abierta desarrollada por Garmin, con la que los usuarios de la marca pueden descargarse variedad de aplicaciones y herramientas que les permitirán personalizar su reloj.
Garmin según sus gustos, preferencias y necesidades, creando así un reloj único adaptado a cada estilo de vida. Básicamente, es como una “app store” abierta donde terceras partes pueden desarrollar y publicar aplicaciones para los relojes-gps y ciclocomputadores de Garmin.

Una de las claves del buen funcionamiento de esta plataforma Connect IQ (LBDC, 2017), se debe al conjunto de herramientas que Garmin ofrece a los desarrolladores: un lenguaje de programación relativamente sencillo, manuales, un foro activo para resolver dudas, un SDK (kit de desarrollo de software) con el que poder simular las aplicaciones creadas y una API (de pago) para comunicar la aplicación del reloj con la web o aplicación móvil y, además, no coarta la posibilidad de monetizar el trabajo, ya sea con donaciones vía paypal (la mayoría) o con versiones premium o suscripciones anuales. Pero la realidad es que el éxito proviene principalmente de la utilidad real de las aplicaciones que se han ido subiendo, ya que éstas ofrecen la posibilidad de mejorar de cualquier dispositivo mediante:

- Personalización: más allá de la propia configuración que ofrece el dispositivo, tanto en la esfera principal, como en los perfiles deportivos o los distintos widgets.
- Actualización del dispositivo: añadiendo funciones a relojes de gama media que sólo encontrábamos en los de gama alta.
- Gratuidad (o casi): la mayoría de aplicaciones son gratuitas, y las que no los son tienen un precio muy reducido y una versión gratuita para testear la aplicación.
- Simplicidad: tanto a la hora de buscar y elegir las aplicaciones, como en la instalación (guía básica de instalación).

En líneas generales, se tiene que Connect IQ es el SDK exclusivo para dispositivos Garmin compatibles. Esto significa, que éste provee todo el entorno para poder programar aplicaciones propias que se ejecuten en los smartwatches y otros dispositivos de la marca, para lo cual se programa en el lenguaje Monkey C,
como se detallará posteriormente en este trabajo (Garmin, 2017b). Los tres pilares con los que Garmin diseñó Connect IQ son:

1. Portabilidad: máxima eficiencia en el consumo energético, seguimiento de actividades e integración con sensores ANT, esto significa mayor tiempo usando y disfrutando del dispositivo, que cargándolo.

2. Localización: el seguimiento de la ubicación es importante, pero los productos no deben volverse inútiles sin un Smartphone; aunque con uno sea mucho más útil.

3. Ecosistema: permitir a los desarrolladores a extender sus aplicaciones dentro del ecosistema.

La instalación, configuración y utilización de Connect IQ se abordará en la implementación tecnológica.

**Tipos de aplicaciones que se pueden desarrollar**

Se pueden desarrollar diferentes tipos de aplicaciones para los wearables, acordes al tipo de interacción que se pretenda realizar con el dispositivo. Esta categorización en cuatro tipos según Garmin (2017b), es la siguiente:

- **Watch Faces**: este tipo de aplicación reemplaza “la cara” principal del wearable, y son la pantalla principal (home screen) del mismo.

- **Data Fields**: Data fields o campos de datos, se ejecutan dentro de las actividades, permitiendo a los programadores a computar valores basados en la actividad física actual que está realizando el atleta, como el ciclismo, carrera, natación, etcétera.

- **Widgets**: estos son páginas a pantalla completa que pueden estar embebidos dentro del carrousel informativo de la página principal del wearable. Los widgets se cargan al momento de ser impresos en pantalla, y se apagan luego de un timeout, un período de inactividad específico del dispositivo, o bien, cuando el usuario cambia al siguiente widget.
Wearable Apps: es una aplicación completa propiamente dicha para el dispositivo wearable, entiéndase por esto, que la app tiene control completo de la interfaz de usuario, pudiéndose implementar desde actividades de entrenamiento personalizadas hasta juegos.

**Lenguaje de programación: Monkey C**

Para el desarrollo de aplicaciones nativas en dispositivos wearables, Garmin (2017b) creó un lenguaje de programación orientado a objetos desde cero, al cual llamaron Monkey C; la aplicación que se desarrolla en el marco del presente trabajo, es justamente escrita en dicho lenguaje. Los autores de este nuevo lenguaje indicaron que los motivos que persiguieron al diseñarlo, fueron:

a) Que la programación se centre más en cliente y menos en las limitaciones de recursos.

b) Optimización de uso memoria a través de limpiezas automáticas de referencias.

c) Facilidad de manipulación de imágenes y fuentes para adaptación a diferentes dispositivos.

El lenguaje Monkey C tiene una sintaxis de fácil comprensión para programadores familiarizados con lenguajes dinámicos, como lo son Java, PHP, Ruby, o Python.

**Diferencias con otros lenguajes de programación**

De manera similar en que el idioma italiano y el castellano derivan del latín, Monkey C deriva fuertemente de lenguajes de programación más antiguos que influenciaron su diseño, ellos son C, Java, JavaScript, Python Lua, Ruby, and PHP.

A modo ejemplificativo, Monkey C compila a byte code que es interpretado por una máquina virtual, igual que Java; de hecho, el manejo de memoria y limpiado de la misma, funciona similarmente en ambos. Monkey C posee módulos que
tienen la misma finalidad que los paquetes en Java, salvo que dicho módulos pueden tener variables y funciones; no sólo clases como Java, permitiendo tener funciones y variables estáticas a nivel de módulo, no de clase. Luego, una diferencia importante entre ellos, es que Monkey C no tiene tipos de datos primitivos, tanto enteros, flotantes, caracteres son objetos y tienen métodos como cualquier otro objeto.

Monkey C no es tampoco un lenguaje tipado, o sea que el programador no debe declarar los tipos de datos para todos los parámetros ni valores retornados. De haber una operación inconsistente entre dos variables de diferente tipos de datos, la excepción de identifica y dispara en tiempo de ejecución (Garmin, 2017b).

Por ejemplo:

```javascript
function add( a, b ) {
    return a + b;
}

function estaFuncionUtilizaAdd() {
    var a = add( 1, 3 ); // Retorna 4
    var b = add( "Hola ", "Mundo" ); // Retorna "Hola Mundo"
}
```

En comparación a Ruby y Python, donde los objetos son tablas hash y tienen muchas de las propiedades de dichas tablas, funciones y variables pueden agregar a objetos en tiempo de ejecución, esto no sucede en Monkey C; donde todos los objetos son compilados y no se pueden modificar en tiempo de ejecución, todas las variables se tienen que declarar antes de ser usadas, ya sean locales a la función, en la instancia de la clase o en el módulo padre.

Se podrían listar más similitudes y diferencias con otros lenguajes populares en la industria como Javascript, PHP, etcétera. pero dicha profundización escapa del foco del presente proyecto.

Despliegue utilizando Garmin Connect IQ

Figura 14: Diagrama despliegue desde sensores a servidores.
Fuente: Elaboración propia.

Despliegue utilizando Garmin Connect Push API

Figura 15: Diagrama despliegue Garmin Connect Push API.
Fuente: Elaboración propia.
Otras API
Garmin (2017b), claramente manifesta una estrategia corporativa de construir un ecosistema de desarrolladores para sus dispositivos, tanto para los wearables como se ha visto anteriormente, como así también sus líneas para náutica, navegación, mapas, cámara VIRB, sensores y dispositivos que implementen ANT+ para compatibilidad con hardware Garmin, etcétera.

A modo referencial se menciona cada una de las API ofrecidas; de cara a futuros trabajos, es importante es conocer la funcionalidad y marco de aplicación de las mismas.

- Wellness Program: API orientada al desarrollo de aplicaciones de salud abastecidas por datos de ejercicios obtenidos de los dispositivos Garmin.
- Fleet Management: interfaces, protocolos y API para la gestión de flota vehicular.
- Custom Maps: herramientas de diseño, construcción y personalización de mapas para ser utilizados en dispositivos Garmin.
- Custom POIs & Voices: herramientas de creación de puntos de interés personalizados (POI por sus siglas en inglés), guías de viaje y voces para una amplia variedad de dispositivos.
- VIRB Program: API que permite controlar, configurar y transmitir vía stream en las cámaras de acción VIRB.
- ANT Wireless Networks: herramientas para conexiones wireless en sensores y dispositivos para deportes, ejercicios y otros.

Garmin: Selección de dispositivos
El espectro de opciones de hardware de Garmin (2017b) es muy variado, las diferencias principales entre los dispositivos se basan en relación al deporte de resistencia para el cual se utilizan (carrera, natación o ciclismo); sensores con los que viene equipado (frecuencia cardíaca, GPS, acelerómetro, etcétera); características técnicas que lo definen (tamaño de pantalla, gama de colores de la pantalla, interfaz de usuario touchscreen o por botones, entre otros).
Connect IQ (el SDK de Garmin) no está disponible para todos los dispositivos, esto significa que en estos equipos que se siguen comercializando pero corresponden a una generación anterior de hardware, es imposible programar una aplicación nativa que se ejecute en el dispositivo; por lo contrario, sólo se podrán usar las aplicaciones que ofrece el vendedor a través de su tienda.

Es condición que el dispositivo sea compatible con Connect IQ para poder desarrollar una aplicación propia utilizando Monkey-C. Adicionalmente, en el siguiente enlace se puede consultar todos los dispositivos compatibles: https://developer.garmin.com/connect-iq/compatible-devices

Es importante reducir la cantidad los dispositivos candidatos donde se procederá a desarrollar, dado que el esfuerzo de programación en wearables es dependiente de la cantidad de equipos distintos de hardware donde se ejecutará. Con base en el mercado local, se optó por seleccionar un dispositivo por deporte; en complemento se hace mención al enlace donde se pueden acceder los detalles asociados a cada uno de ellos.

- **Running**: Smartwatch: “Fenix 5”: es un reloj GPS multideportivo con medición de frecuencia cardíaca en la muñeca, características deportivas avanzadas y bandas intercambiables que le permiten al usuario, ir desde el trabajo a su sesión de entrenamiento sin interrupciones. Sin importar qué deporte se desee seguir, el Fénix 5 lo abarca gracias a perfiles de actividad incorporados y métricas de rendimiento, además de notificaciones inteligentes.

- **Swimming**: Smartwatch: “Garmin Swim”: reloj de natación diseñado para uso en piscina que se encarga de calcular distancia, ritmo, número de brazadas y mucho más.

- **Cycling**: Minicomputer GPS: “Edge 820”: computadora para bicicletas con GPS, compacta y con pantalla táctil; ofrece monitoreo de rendimiento
avanzado, navegación giro a giro específica para ciclismo, nuestra nueva función GroupTrack y más.


**Resumen del capítulo**
En este capítulo se investigaron y se plasmaron los conceptos claves relacionados al desarrollo de wearables; desde los fundamentos, el diseño de interfaz, la comunicación entre ellos, los datos que generan, y arquitecturas de software para gestionar los mismos.

Posteriormente, se identificaron los fabricantes más importantes del mercado, se evaluaron de acuerdo a los criterios más relevantes en marco de este trabajo, y se seleccionó uno de ellos; la diferencia con el resto, fue principalmente la posibilidad de generar aplicaciones nativas para su sistema operativo.

Tras profundizar en el entorno de desarrollo del hardware seleccionado, quedaron definidos los fundamentos para comenzar con la implementación técnica, abordada en todo su alcance en el siguiente capítulo.
Implementación Tecnológica

Ciclo de desarrollo propuesto

El proceso de desarrollo de un nuevo producto de software también se conoce como SDLC (Software Development Life Cycle) o Ciclo de Vida del Desarrollo de Software, y puede considerarse una subcategoría del ciclo de vida de desarrollo de sistemas. Existen varios modelos de SDLCs y se pueden estandarizar bajo la ISO/IEC 12207, la cual enumera todas las tareas que deben formar parte del desarrollo y mantenimiento de software. Normalmente, este ciclo de vida del desarrollo de software, incluye las siguientes fases (Pressman, 2005):

a) La fase de planificación y análisis.

b) Fases de implementación, pruebas y documentación del código.

c) Fases de despliegue y mantenimiento del software.

En relación al desarrollo del software involucrado en el presente trabajo, bajo la consideración de que desarrollar en IoT es un proceso de descubrimiento, se escogió el “paso a paso” de un ciclo de vida de las metodologías ágiles (Larman, 2004), considerado por como un ciclo de vida iterativo e incremental, con iteraciones cortas (semanas) y sin que dentro de cada iteración tenga por qué haber fases lineales (tipo cascada).

Para más de detalle citamos a (Iparraguirre, 2016) donde expresa: incremental = añadir, iterativo = retrabajo. En este ciclo de vida, se van liberando partes del producto (prototipos) periódicamente, en cada iteración, y cada nueva versión, normalmente, aumenta la funcionalidad y mejora en calidad respecto a la anterior.

El desarrollo ágil de software envuelve un enfoque para la toma de decisiones en los proyectos de software, que se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan con el tiempo según la necesidad del proyecto (Chin, 2004). Así, el trabajo es realizado mediante la colaboración de equipos auto-organizados y
multidisciplinarios, inmersos en un proceso compartido de toma de decisiones a corto plazo.

Cada iteración del ciclo de vida incluye: planificación, análisis de requisitos, diseño, codificación, pruebas y documentación. Teniendo gran importancia el concepto de "Finalizado" (Done), ya que el objetivo de cada iteración no es agregar toda la funcionalidad para justificar el lanzamiento del producto al mercado, sino incrementar el valor por medio de "software que funciona" (sin errores). Estas iteraciones son cortas en el tiempo, de pocas semanas, lo que permite una adaptación a cambios de requerimientos menos costosa, pagándose en contrapartida con retrabajo.

Optar por esta metodología se basó concretamente que por tratarse de un proceso exploratorio y de aprendizaje, se podrá ir mejorando el desarrollo a manera que se adquiera mayor conocimiento de la tecnología, permitiendo avances significativos en cada iteración.

Para cada iteración se proponen fases que la componen. Las fases 1 y 2 a diferencia de las restantes no se va a ejecutar en cada iteración, por lo contrario tienen mucha preponderancia en la primera iteración por tratarse de investigación, definición de tecnología y configuración inicial del entorno necesario para poder desarrollar, estos aspectos serán prácticamente estáticos en el transcurso del proyecto, pero por simplicidad, en el este trabajo se los incluye en el proceso iterativo. En contrapartida, las fases 3 a la 7 si serán muy dinámicas, ya que el alcance puede ampliarse a medida que se conozcan más la onerosidades de la tecnología, los prototipos pueden mejorar al descubrir nuevos componentes y tras las repetidas pruebas, y la programación en sí es la fase más iterativa de todas por ir incorporando funcionalidad incrementalmente.
### Fases de desarrollo propuestas para cada Iteración

<table>
<thead>
<tr>
<th>Fase</th>
<th>Observación</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Investigación y selección de tecnologías. Selección de arquitectura Garmin.</td>
</tr>
<tr>
<td>2</td>
<td>Configuración del entorno de desarrollo. IDE de desarrollo, compilador, etcétera.</td>
</tr>
<tr>
<td>3</td>
<td>Análisis de requerimientos y definición de alcance. Especificación del alcance de la aplicación.</td>
</tr>
<tr>
<td>4</td>
<td>Mockups o prototipos de interfaz. Boradores de pantalla</td>
</tr>
<tr>
<td>5</td>
<td>Implementación de interfaces y experiencia de usuario en emulador del dispositivo. Visualización de componentes en pantallas. Navegabilidad a través de las pantallas.</td>
</tr>
<tr>
<td>6</td>
<td>Generación y persistencia de datos. Obtención y persistencia de datos necesarios en la aplicación.</td>
</tr>
<tr>
<td>7</td>
<td>Pruebas. Ejecución de casos de prueba</td>
</tr>
</tbody>
</table>

### 1. Investigación y selección de tecnologías

Tras la investigación de hardware y tecnologías actuales desarrollada en el capítulo anterior de marco tecnológico, se concluyó seleccionando a los wearables de Garmin para la implementación de esta solución.

### 2. Configuración del entorno de desarrollo

La instalación es muy sencilla, sólo requiere de tener al entorno de ejecución de Java versión 8 o superior (JRE 1.8+). Consiste básicamente en descargar el SDK de connect IQ que se provee empaquetado acorde al sistema operativo. Se descomprime, se actualiza el PATH a la carpeta bin del SDK, y listo.

**Comandos.**

Los tres comandos que se utilizan del SDK son:

- **Monkeyc.** Es el compilador. Convierte el código fuente en un ejecutable .prg
- **Connectiq.** Es el simulador. Se utiliza para testear las apps en la computadora.

- **Monkeydo.** Es el ejecutador de programas ya compilados en el simulador.

Como por ejemplo, para llevar a cabo los siguientes pasos:

1. **Compilar**
2. **Lanzar simulador**
3. **Ejecutar programa en simulador**

Los comandos a utilizar, serían:

```
> monkeyc -o myApp.prg -m manifest.xml myApp.mc
> connectiq
> monkeydo myApp.prg
```

El único punto extra a tener en cuenta, es que el compilador requiere una clave de desarrollador para firmar las aplicaciones, que debe ser del tipo “RSA 4096 bit private key”. La generación con OpenSSL es muy sencilla, y la finalidad de esto es la protección del código, y seguimiento posterior a su subida al store, ya que sólo podrá ser actualizado si se cuenta con la firma con la que se subió el programa.

Cada desarrollador puede utilizar la IDE de desarrollo que desee y luego compilar como se mostró anteriormente. No obstante se recomienda utilizar Eclipse IDE, ya que en ella se provee un plugin donde se manipulan gráficamente las herramientas principales del SDK, se reconocen los tipos de archivos, tiene autocompletado, etcétera.

El IDE no es dependiente de un lenguaje de programación, en cambio se pueden utilizar diferentes a través de la IDE Eclipse, inclusive Monkey-C configurando algunos componentes extra que así lo hacen posible, tal como se explica en el tutorial en línea sugerido por Garmin (2017e).
3. Análisis de requerimientos y definición de alcance

La amplitud de posibilidades para desarrollar apps para wearables es muy grande; con intención de que el alcance sea controlable, pero a su vez lo suficientemente amplio para poder cumplir con el objetivo del proyecto, se opta por la implementación de un test de valoración de rendimiento, los cuales no sólo son de gran utilidad en la labor diaria de los entrenadores, sino que también se pueden mejorar y optimizar con el uso de la tecnología en cuestión, en comparación a los tradicionales cronómetros que se emplean.

Los test de valoración en su gran mayoría están compuestos por tres fases: entrada en calor, bloque principal y vuelta a la calma; en ellas a su vez se dosifica una o varias actividades físicas que el atleta debe realizar, y se definen por su duración y alguna métrica dependiente al tipo de actividad.

Hay decenas de tests en cada uno de los tres deportes de resistencia. En la Tabla 2 de la página siguiente, se presenta un ejemplo para carrera, ciclismo y natación.

Para el caso de ciclismo, si bien el ciclo de desarrollo de la aplicación, el lenguaje de programación y recomendaciones de usabilidad son los mismos que para cualquier reloj inteligente, el dispositivo no es un wearable en sí, dado que se trata de una mini computadora que se coloca sobre la bicicleta. Por ende, no se prioriza el test “Protocolo Cronos 3 y 5” para ciclismo, por contrapartida se presenta dicha implementación como un trabajo futuro y se detalla en el Anexo I “Implementación de test para Ciclismo”.
### Tabla 2: Test de valoración de rendimiento x deporte.

<table>
<thead>
<tr>
<th>Ciclismo</th>
<th>Warm-Up</th>
<th>Main</th>
<th>Cool Down</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocolo Cronos 3 y 5 min</td>
<td>1. 10 min en Zona 1 de FTP (Recuperación).</td>
<td>1. 3 minutos a la máxima intensidad posible.</td>
<td>10-15 min en Zonas de FTP: 1 (Recuperación) – 2 (Fondo).</td>
</tr>
<tr>
<td></td>
<td>2. 10 min en Zona 2 de FTP (Fondo).</td>
<td>2. 15-20 min en Zona 1 de FTP (Recuperación).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3. 3 repeticiones de 1 minuto a 100 rpm, pausa: 1 min rodando a baja intensidad.</td>
<td>3. 5 minutos a la máxima intensidad posible.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. 5 min en Zona 2 de FTP (Fondo).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Carrera</td>
<td>1. 3 minutos a la máxima intensidad posible.</td>
<td>2. 100 km a 85% de ritmo umbral.</td>
<td>2.00 km a 85% de ritmo umbral.</td>
</tr>
<tr>
<td>Protocolo 2k</td>
<td></td>
<td>2. 100 km a la máxima intensidad posible.</td>
<td></td>
</tr>
<tr>
<td>Natación</td>
<td>1. 400 metros a la máxima intensidad posible.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Fuente:** Elaboración propia.

Entre los deportes de natación y carrera, se opta por implementar éste último; es decir, por el “Test de Running de 2km”.

**Fundamentos.**

En relación a los fundamentos, es importante tener en cuenta los siguientes aspectos:

- Es un protocolo de amplia adopción y globalmente adoptado.
- Presenta gran similitud con otros test que persiguen mismo objetivo, como por ejemplo el test de 5km, ya que con ambos se puede construir la curva de potencia crítica.
En natación por ser un deporte muy técnico se prioriza la percepción del esfuerzo subjetiva de cada atleta, en carrera si bien esto también se considera, los valores objetivos son más relevantes en este deporte.

La carga interna se cuantifica a partir del sensor de frecuencia cardíaca que viene integrado en el propio dispositivo, al igual que la carga externa cuantificada a partir de la medición de distancia y ritmo, ambos medibles en el mismo dispositivo.

Existe la posibilidad de futuros trabajos a partir de integración ANT+ con nuevos dispositivos experimentales con sensores de potencia aplicados al running, para la cuantificación de carga externa, como lo es el Stryd (Stryd, 2017).

**Objetivo.**

Desarrollar una aplicación nativa en Monkey-C que realice todas las lecturas necesarias para automatizar la obtención de resultados del test de valoración de rendimiento de “Running 2km”.

**Alcance.**

Desde que el atleta carga los datos requeridos de su perfil fisiológico, hasta que la aplicación arroja los resultados de sus zonas de potencia posterior a la ejecución asistida del test.

**4. Mockups o prototipos de interfaz**

Para el diseño de los prototipos de interfaz, se siguieron las recomendaciones de experiencia de uso y guías de diseño oficiales de Garmin, fabricante del hardware seleccionado (Garmin, 2017d).
Prototipo principal.
En la Figura 16 se observa que el prototipo está montado sobre un wearable del tipo reloj inteligente (smartwatch por su nombre en inglés), modelo Forerunner 735XT de Garmin, que fue el seleccionado para este desarrollo.

En dicha figura, se observa la distribución de los datos en la pantalla principal de la aplicación. Los mismos son informativos, de acuerdo al progreso del test de rendimiento en cualquier momento que se lo consulte. En detalle.

- Fases del test. Indicador de fase actual, donde la primera fase “W” es la entrada en calor, la segunda y principal “M” es el bloque principal, y la tercera y última “C” es la vuelta a la calma.
Título. Indicador del nombre de la actividad actual que se está realizando, este proviene de que una fase de un test, puede estar compuesto por más de una actividad.

Variables actuales. Indicadores actuales de progreso de la actividad, es la información más importante que se le brinda al atleta, motivo por el cual ocupa el centro del dispositivo, que tal cómo se mencionó en las consideraciones en el desarrollo de wearables, es la zona donde el usuario enfoca mayor atención. Los datos brindados corresponden al tiempo transcurrido desde que se inició la actividad actual, y el valor actual de la métrica que le corresponde a dicha actividad, la cual podría ser frecuencia cardíaca, potencia, distancia, etcétera.

Métricas objetivos. Indicador de la duración de la actividad, y del valor esperado (o los valores si fuera un intervalo) en la métrica correspondiente. Por ejemplo podría ser, una duración de 2 minutos, con una frecuencia cardíaca entre 135 y 160 bpm o pulsaciones por minuto.

Flujo de pantallas.
A partir del prototipo principal, se plantea un flujo entre interfaces que abarcan otras pantallas complementarias, considerando que luego a la aplicación se le puedan incorporar más test como el de 5km, y el de Montreal.

La primera interacción con el usuario es la de selección de test, en la Figura 17 (página siguiente) se considera otros posibles tests.

En la siguiente pantalla, mostrada en la Figura 18 (página siguiente), se le presenta el test al usuario, quién podrá visualizar como está compuesto utilizando los botones de navegación normalmente dedicados a tal fin, y podrá proceder a darle inicio con el botón de play.
Figura 17: Prototipo Selección de Test.  
Fuente: Elaboración propia.

Figura 18: Prototipo presentación de Test.  
Fuente: Elaboración propia.
Asimismo, la Figura 19 es una muestra de la información que se brinda mientras se ejecuta alguna actividad de alguna fase del test. Es concretamente un ejemplo de la distribución de datos que se planteó en el prototipo principal de la Figura 16.

![Figura 19: Prototipo Test en ejecución. Fuente: Elaboración propia.](image)

Por último al final el test, se le brinda al usuario un resumen con el resultado del mismo, que incluirá tiempo, frecuencia cardíaca mínima, máxima y promedio, tal como se diseñó en la Figura 20 de la siguiente página. Este resultado, en conjunto con el de otros test, son muy valiosos para entrenadores especializados, que a partir de los mismos, determinan curvas de velocidad crítica, un recurso muy valioso al momento de programar y dosificar el entrenamiento.

Finalizado el prototipado, se cuenta con toda la especificación técnica, funcional y visual para comenzar con la programación, para lo cual es necesario configurar el entorno que provee las herramientas necesarias.
5. Implementación de UIX en emulador del dispositivo

Se cuenta con dos alternativas de programación de la interfaz y experiencia de usuario, una es la definición estática a través de un archivo xml que define las pantallas con las cual interactua el usuario, y la otra es programación dinámica, que a través de diferentes sentencias del lenguaje nativo Monkey-C, se accede a posibilidades de dibujar y actualizar cada pantalla en el dispositivo, en el momento adecuado.

Definición estática de interfaz de usuario “layout.xml”.

Ejemplo de la cabecera.

Layout.xml

```xml
<layout id="Cronos3SLayout">
  <label x="center" y="5" text="#Strings.title_warmup" color="Gfx.COLOR_BLUE"
        justification="Gfx.TEXT_JUSTIFY_CENTER" />
  <label x="center" y="35" text="#Strings.test_cronos3y5_w1"
        color="Gfx.COLOR_WHITE"
        justification="Gfx.TEXT_JUSTIFY_CENTER" />
  <label x="center" y="65" text="#Strings.test_cronos3y5_w2"
```
Strings.xml

<strings>
  <string id="AppName">iewgTestRendimiento</string>
  <string id="prompt">Menu para iniciar</string>
  <string id="menu_label_cronos3y5">Cronos 3-5</string>
  <string id="menu_label_cronos20">Crono 20</string>
  <string id="menu_label_ais">Test AIS</string>
  <string id="title_warmup">Entrada en Calor</string>
  <string id="menu_label_principal">Bloque Principal</string>
  <string id="menu_label_calma">Vuelta a Calma</string>
  <string id="test_cronos3y5_w1">10' Zona 1 (Recuperación)</string>
  <string id="test_cronos3y5_w2">10' Zona 2 (Fondo)</string>
  <string id="test_cronos3y5_w3">3 rep a 100rpm</string>
  <string id="test_cronos3y5_w4">5' Zona 2 (Fondo)</string>
</strings>

Drawables.xml

<drawables>
  <bitmap id="LauncherIcon" filename="../images/launcher_icon.png"/>
  <bitmap id="buttonStart" filename="../images/button_start.png"/>
</drawables>

Programación dinámica.

// Esta función se ejecuta cada Segundo mientras esta pantalla está activa (visible) en el dispositivo. Tiene la finalidad de imprimir y actualizar en pantalla todos los componentes de la interfaz.
function onUpdate(dc)
{
    View.onUpdate(dc);
    dc.clear();

    var width = dc.getWidth();
    var height = dc.getHeight();
    var marginTop = 5;
    var marginMid = height/2 - 5;
    var marginLeft = 3;
    var fWidth;
    var fHeight;
    var text;
    var font;

    //Dibujar el título
text = title;
font = Gfx.FONT_SYSTEM_MEDIUM;
fWidth = dc.getTextWidthInPixels(text, font);
fHeight = Gfx.getFontHeight(font);
dc.setColor(Gfx.COLOR_YELLOW, Gfx.COLOR_TRANSPARENT);
dc.drawText(width/2 - fWidth/2, marginTop, font, text,
Gfx.TEXT_JUSTIFY_LEFT);
dc.setColor(Gfx.COLOR_WHITE, Gfx.COLOR_TRANSPARENT);
dc.drawRectangle(0, 0, width, fHeight+(marginTop*2));

//Dibujar el tiempo objetivo
font = Gfx.FONT_SMALL;
text = Lang.format("$1$:$2$", [limite / 60, (limite %
60).format("%02d")]);
fWidth = dc.getTextWidthInPixels(text, font);
dc.drawText(width/4 - fWidth/2, marginTop + height/6, font, text,
Gfx.TEXT_JUSTIFY_LEFT);

//Dibujar el intervalo de potencia objetivo
font = Gfx.FONT_SMALL;
text = powerIntervalGoal;
fWidth = dc.getTextWidthInPixels(text, font);
dc.drawText((width * 3 / 4) - fWidth/2 + marginLeft, marginTop + height/6,
font, text, Gfx.TEXT_JUSTIFY_LEFT);

//Dibujar el tiempo actual transcurrido
font = Gfx.FONT_MEDIUM;
if (counter == null){text="0";}
else {text = counter;}
var timeString = Lang.format("$1$:$2$", [counter / 60, (counter %
60).format("%02d")]);
fWidth = dc.getTextWidthInPixels(timeString, font);
dc.drawText(width/4 - fWidth/2, marginMid, font, timeString,
Gfx.TEXT_JUSTIFY_LEFT);

//Draw the current AVG power (W) or current cadence (RPM)
font = Gfx.FONT_MEDIUM;
if (currentStep < 3){ //! Separar por tipo de actividad esto es
Hardcodeado.
    if(info.averagePower == null){text="0";}
    else {text = info.averagePower;}
} else {
    if(info.currentCadence == null){text="00";}
    else {text = info.currentCadence;}
}
fWidth = dc.getTextWidthInPixels("000", font);
dc.drawText((width * 3 / 4) - fWidth/2, marginMid, font, text,
Gfx.TEXT_JUSTIFY_LEFT);
}

Estático versus dinámico.
La idea principal del sistema estático de layout es separar de una forma simple, la lógica de negocio de la interfaz de usuario. Tiene la gran ventaja de la sencillez; pero en contrapartida, por heterogeneidad de hardware, esta sencillez no siempre se puede aprovechar. Un caso de ejemplo sería una app que tiene como requerimiento que se ejecute en relojes de pantalla redonda y cuadrada; en este caso, los archivos XML de layout deben sobrescribirse, atentando contra la simplicidad mencionada.

El modo dinámico requiere un poco más de esfuerzo de programación, lo cual se compensa con mayor flexibilidad y eficiencia, éste último no es un factor menor; el sistema definición estática requiere más memoria proveniente del parseo y manipulación de los archivos XML.

Estará a criterio del programador qué sistema utilizar, lo cual puede influenciar en la decisión los requerimientos de la aplicación. En este caso, por tratarse de una aplicación con fines académicos, se combinará el uso de los dos, priorizando el aprendizaje por sobre las buenas prácticas de programación y optimización de recursos.

**Recomendaciones de UIX del vendedor.**

Podría haberse usado layout.xml si se usaban valores absolutos. Al comenzar con la programación de las interfaces gráficas, fue prioridad respetar el diseño tal cual se especificó en la fase correspondiente de diseño de mockups. Algunos aprendizajes importantes sobre el esfuerzo de realización y las recomendaciones de usabilidad de Garmin (Garmin, 2017f), influenciaron en un leve rediseño de las pantallas principales, esto es propio del aprendizaje inherente de la tecnología y uso de la misma.
6. Generación y persistencia de datos

Remitiéndonos a la documentación en línea de Connect IQ (Garmin, 2017g), observamos que el SDK llamado “Toybox” está organizado jerárquicamente por módulos que son contenedores lógicos de clases, luego al momento de programar, se referencian sólo los que se utilizan. Esta es una estructura típica de cualquier SDK en todo lenguaje de programación.

Por ejemplo, en la fase 5 de implementación de UIX en el dispositivo emulador, se presentó un fragmento de código de programación dinámica de interfaz, que tenía la finalidad de imprimir por pantalla todos los componentes gráficos, más precisamente era el método onUpdate(dc), que pertenece a la clase View, la principal del módulo WatchUI.

```java
// Se importa el módulo a utilizar y se setea un alias para fácil referenciación.
using Toybox.WatchUi as Ui;

// En la declaración de la clase la heredamos de View, clase principal del módulo.
class ActivityView extends Ui.View {

  // View tiene tres subclases: DataField, Picker y Watchface, que se pueden instanciar para mayor especialización, vamos a utilizar DataField.
  class ActivityView extends Ui.DataField {

    // Función que actualiza contenido dinámico cuando la View está activa.
    function onUpdate(dc) {

      // Se cuenta con más de 20 módulos y 50 clases, abarcando un amplio espectro de posibilidades de desarrollo, como ser manejo de operaciones matemáticas, gráficos, sensores, comunicaciones, actividad física, temporizadores, etc. Todas detalladas en la documentación pertinente (Garmin, 2017g). No es intención detallar cada una ellas, si tomar foco en las que están relacionadas con el alcance del proyecto que utiliza las más importantes.

      Generación de datos necesarios.
```
Considerando el origen de los datos en una aplicación para wearables, los podemos agrupar en 3 grandes categorías bien diferenciadas.

a) Desde el dispositivo: Obtención desde recursos propios del reloj, se diferencian entre modelos porque algunos vienen con más sensores y funcionalidades que otros.

b) Desde los sensores: A través de ANT+ el reloj se puede conectar con un gran abanico de otros dispositivos de hardware que implementen este protocolo de comunicación, potenciómetros, termómetros, bandas elásticas de frecuencia cardíaca, etc.

c) Calculables: Son aquellos datos resultados de aplicarle alguna fórmula o algún procesamiento matemático a un conjunto de datos bases.

Acordes a los requerimientos es necesario obtener o calcular 3 datos claves para construir la aplicación: tiempo, distancia y frecuencia cardíaca.

1) Tiempo: Es la variable más básica y esencial, de hecho es lo que mide todo cronómetro tradicional y obviamente cualquier reloj inteligente.

2) Distancia: También es otra de las variables más sencillas de obtener, si bien un cronómetro no la brinda, todo reloj inteligente sí lo hace dado que viene equipados con GPS o acelerómetro. El SDK de Garmin brinda información relacionada a la distancia proveniente directamente desde el dispositivo, para poder manipularse, se cuenta con el módulo “Activity” que contiene datos sobre la actividad física actual, no sólo la distancia, sino también otras variables que puede originarse nativamente en el dispositivo, cómo ser la cadencia, calorías consumidas, etc.

Un ejemplo de utilización de dicho módulo actividad:
// Importar el módulo actividad
using Toybox.Activity as Act;

// declarar un objeto llamado “info”, y asignarle los valores actuales de la actividad.
var info = Act.getActivityInfo();

// declarar variable “distancia” y asignarle la distancia actual de la actividad
var distancia = info.elapsedDistance;

// declarar variable “actualFC” y asignarle la frecuencia cardíaca actual de la actividad.
var actualFC = info.currentHeartRate;

De una manera sencilla y similar a ésta, se obtienen otros parámetros muy útiles como el desnivel acumulado, ya sean ganados (distancia recorrida en ascenso), o perdido (distancia recorrida en descenso). Si bien en estos valores no son necesarios para este proyecto de Test de 2km, son extremadamente útiles en otros escenarios.

3) Frecuencia cardíaca: Las veces que late el corazón en un minuto, es a pesar de sus limitaciones, la forma más utilizada para controlar la intensidad, proviene desde el dispositivo si es que el mismo tiene incorporado un sensor óptico en la muñeca, o bien desde sensores externos como lo son las bandas elásticas que el atleta utiliza en el pecho.

En caso de sensores ópticos incorporados en la muñeca, la obtención de este dato es trivial a través de módulo “Activity” anteriormente mencionado.

// Info es un objeto del tipo “Actividad”, brinda la frecuencia cardiaca actual de la actividad en curso accediendo a una de sus propiedades.
var actualFC = info.currentHeartRate;

Ahora bien, en el caso de que el dispositivo no cuente el sensor en cuestión, la vía para obtener estos datos provenientes de un hardware externo, es utilizando el módulo “Sensor” del SDK, quien permite a la aplicación, para controlar cualquier dispositivo que soporte ANT+ nativamente.
Ejemplo de la utilización de un sensor estándar.

//darle permiso a la aplicación de acceso a sensores “<iq:uses-permission id="Sensor"/>”, esto se realiza en el manifest.xml, archivo que contiene además los dispositivos para los cuales se estará disponible y los idiomas soportados entre otros datos de configuración.

```
<iq:manifest xmlns:iq="http://www.garmin.com/xml/connectiq" version="0">
  <iq:application entry="tesis2kmApp" id="d06b54b18c6b484f9560835c7329bd89" launcherIcon="#Drawables.LauncherIcon" minSdkVersion="2.1.0" name="#Strings.AppName" type="watch-app">
    <iq:products>
      <iq:product id="fenix5x"/>
      <iq:product id="fenix5"/>
      <iq:product id="fenix5s"/>
      <iq:product id="fenixchronos"/>
      <iq:product id="fr735xt"/>
    </iq:products>
    <iq:permissions>
      <iq:uses-permission id="Fit"/>
      <iq:uses-permission id="Sensor"/>
      <iq:uses-permission id="FitContributor"/>
    </iq:permissions>
    <iq:languages>
      <iq:language>eng</iq:language>
    </iq:languages>
  </iq:application>
</iq:manifest>
```

//importar el módulo Sensor
using Toybox.Sensor as Sensor;

// METODO: Constructor
function initialize() {

  // Setear en el módulo los sensores que se utilizarán
  Sensor.setEnabledSensors([Sensor.SENSOR_HEARTRATE]);

  // Habilitar métodos que capturen eventos de los sensores
  Sensor.enableSensorEvents( method(onSensor) );
  // . . .

  // METODO onSensor: Todo método disparado por un sensor, recibe de parámetro un objeto de la clase sensorInfo, única de este módulo, con métodos para obtener el dato principal de cada sensor estándar.
  function onSensor(sensorInfo){

    // chequear que el sensor tenga lectura de frecuencia cardíaca.
    if( sensorInfo.heartRate != null ) {
      // asignar frecuencia cardíaca a una variable
      fcActual = sensorInfo.heartRate;
    }

    Una observación importante sobre el fragmento de código anterior, es la simplicidad con la que se obtienen los datos bases de sensores estándares. Entiéndase por estándares, aquellos que son soportados nativamente en el
SDK, como lo es el utilizado sensor de frecuencia cardíaca, potenciómetros, podómetros, temperatura, cadencia y velocidad.
Nos referimos a dato base a aquellos a partir de los cuales se obtienen otros derivados se suma utilidad que se facilitan automáticamente cuando provienen el mismo dispositivo. Notar que el módulo “Activity” cuando el dispositivo está provisto de sensor de frecuencia cardíaca brinda datos como valor actual de la frecuencia cardíaca, valor máximo durante la sesión, valor promedio, etc. En cambio en el módulo “Sensor” sólo provee valor actual.

Actualización de lecturas de datos.
Los presentados valores de frecuencia cardíaca y distancia, como cualquier otro de su misma naturaleza, son instantáneos al momento que se consultan, y deben actualizarse periódicamente cada una cierta cantidad de tiempo para poder construir información valiosa. Dicha actualización debe darse a lugar respetando las capacidades técnicas de los sensores ya sean externos o incorporados en el smartwatch. Por ejemplo la máxima capacidad de frecuencia de actualización de posición es cada un segundo, esto proviene de la capacidad máxima de los sensores GPS de trabajar a 1Ghz.

La vía para llevar esto a cabo, es utilizando la clase “Timer” del módulo homónimo. Tiene la finalidad de llamar a algún método después de transcurridos una cierta cantidad de milisegundos. Hay dos tipos de utilización, el modo “sólo una vez” o “repetitivo”, en este segundo modo es cómo se van actualizando o refrescando las sucesivas lecturas necesarias. Concretamente con un ejemplo.

```java
//importar el módulo Timer
using Toybox.Timer as Timer;

// METODO: Constructor
function initialize() {

// Instanciar la variable “timer” llamando al constructor “Timer” del módulo “Timer”
timer = new Timer.Timer();

// Iniciar el timer para que cada 3 segundos se ejecute el método “actualizarLecturas”
timer.start(method(:actualizarLecturas), 3000, true);
```
El timer se finaliza cuando la lectura de valores indica que la distancia recorrida es de 2km, o bien capturando el evento disparado cuando la atleta cancela o detiene el test. Para liberar consumo de memoria, se sugiere detener el timer para lo cual se utiliza el método stop.

```
// METODO: detenerTest
function detenerTest() {
    timer.stop()
}
```

Tras obtener, actualizar y calcular todos los datos necesarios por la aplicación, el siguiente paso es grabarlos.

Persistencia de datos necesarios.

Existen dos formas de persistir o grabar los datos obtenidos para un posterior uso, la más simple es a través del almacén de objetos usando propiedades de la aplicación, la más completa y escalable a través de creación de un archivo FIT tal como fueron presentados anteriormente en este trabajo.

a) Propiedades en la aplicación: Cada aplicación tiene acceso a su propio almacén de objetos para persistir datos, siguiendo la estructura de mapa compuesto por clave y valor con la que implementa cualquier propiedad. A continuación se ejemplifica cómo se persiste el tiempo total en que el atleta recorrió la distancia objetivo de 2 km

```
// METODO: actualizarLecturas que es disparado frecuentemente por el timer.
function actualizarLecturas(){
    // actualizar y calcular mediciones (distanciaActual, fcPromedio)
    // chequear si se llega el objetivo de 2km (2000 metros)
    if(distanciaActual >= 2000){
        // si se cumplió el objetivo, grabar ambas mediciones
        // grabar con la clave "tiempoTotal" el valor en la variable "tiempoTotalValor". Similar con frecuencia cardiaca promedio
        App.getApp().setProperty("tiempoTotal", tiempoTotalValor);
        App.getApp().setProperty("fcPromedio", fcPromedioValor);
    }
    // . . .
```
// llamar a mostrarResultados, método al terminar el test.
mostrarResultados();

Recuperación de datos persistidos en el almacén de objetos de la aplicación.

// METODO: mostrarResultados obtiene valores ya persistidos.
function mostrarResultados(){
    var tiempoTotalValor = App.getApp().getProperty("tiempoTotal");
    var fcPromedioValor = App.getApp().getProperty("fcPromedio");
}

Tal como se mencionó, se utiliza el almacén de datos tanto para el grabado y obtención de datos, y dicho almacén pertenece a cada aplicación. El módulo “Application” utilizado en este ejemplo con el alias “App” es el que contiene la clase base para cualquier aplicación, posee sólo un método getApp() que retorna un objeto de la única clase que contiene “AppBase”, ésta clase es la base para cualquier aplicación y es utilizada no sólo para acceder al almacén de objetos, sino para gestionar el ciclo de vida de la aplicación. Veamos con un ejemplo.

//importar el módulo Application (obligatorio a toda aplicación)
using Toybox.Application as App;

// declarar clase principal de la aplicación que hereda de AppBase
class tesis2kmApp extends App.AppBase {
    function initialize() {
        // . . .

        // Devolver la vista inicial de la aplicación (método obligatorio de implementar porque da inicio a la interfaz de usuario).
        function getInitialView() {
            var mainView = new ActivityView();
            var viewDelegate = new ActivityDelegate(mainView);
            return [mainView, viewDelegate];
        }
    }
}

Está explicito que cuando invocamos App.getApp() estás obteniendo la clase base de la aplicación, la que nos permite manipular el almacén de datos a través de varios métodos, como ser clearProperties(), deleteProperty(key), getProperty(key), setProperty(key,value), validateProperty(key, value), entre otros.

b) Archivos FIT: Los dispositivos tienen capacidad de generar archivos .fit, para ello el SDK dispone del módulo “ActivityRecording”, con el cual se
puede iniciar, detener y grabar las actividades de una sesión en un archivo FIT. Otro módulo complementario es el de FitContributor, que permite crear campos personalizados dentro del archivo. En el alcance de nuestro proyecto no está el requerimiento de crear un campo propio, pero a fines académicos se incluye en el mismo, un dato propio llamado “recuperación”, que se mide en “bpm” o pulsaciones por minutos, que representan la cantidad de pulsaciones que un atleta recupera (o disminuye) en periodo de descanso post-actividad. Cabe mencionar que es un ejemplo de las posibilidades de la tecnología, y que puede carecer de validez científica en ámbitos de ciencias del ejercicio.

```csharp
// Importar ambos módulos y asignarles un alias.
using Toybox.ActivityRecording as Record;
using Toybox.FitContributor as Fit;

// METODO: Constructor
function initialize() {

// Instanciar una sesión en la variable “ses” que será la grabada.
    ses = Record.createSession({:name=>"Test2km",:sport=>Record.SPORT_RUNNING});

// Instanciar una variable recuperación como un campo nuevo de la sesión
    recuperacion = ses.createField("Recuperacion", 0, Fit.DATA_TYPE_FLOAT, {
        :msgType=>Fit.MESG_TYPE_RECORD, :units=>"bpm" });

// METODO: StartStopTest que da inicio y fin al test, grabando el archivo.
function StartStopTest(){
    // Chequear si la versión del SDK utilizado soporta esta función.
    if(Toybox has :ActivityRecording ){
        // Iniciar el grabado
        if((ses == null)||(ses.isRecording() == false)) {
            recuperacion.setData(recuperacionValor);
            ses.start();
        }
        // Finalizar el grabado.
        else if((ses != null ) && ses.isRecording()) {
            session.stop(); // Frena la sesión.
            session.save(); // La guarda creando el archivo .fit
        }
    }
    // En cualquier caso actualizar la pantalla.
    Ui.requestUpdate();
}
```

El archivo FIT obtenido es un binario, por lo cual no podría ser abierto con un editor de texto plano con expectativas de obtener alguna información
legible. Para convertirlo a un archivo de fácil manipulación, podríamos construir nuestra propia aplicación específica para esto que implemente el protocolo, lo cual conlleva un elevado esfuerzo, o bien utilizar algunas de las herramientas provistas por el mismo SDK. Entre ellas se encuentra FitCSVTool.jar, una pequeña aplicación Java que recibe de parámetro un binario FIT y retorna un archivo CSV como resultado. Estos archivos son de texto plano y como su nombre lo indica, son valores separados por coma, del inglés “Comma-Separated Values”, por lo cual puede ser abierto por planillas de cálculo como forma predeterminada.

En la Tabla 3 se muestra un fragmento del FIT generado en nuestro proyecto, ya convertido en archivo .csv para una posible interpretación humana.

<table>
<thead>
<tr>
<th>Tipo</th>
<th>Ids</th>
<th>Message</th>
<th>Field 1</th>
<th>Val 1</th>
<th>Field 2</th>
<th>Val 2</th>
<th>Field 3</th>
<th>Val 3</th>
<th>Field 4</th>
<th>Val 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definición</td>
<td>0</td>
<td>fit_file_id</td>
<td>0</td>
<td>1</td>
<td>time_created</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>product_name</td>
<td>20</td>
</tr>
<tr>
<td>Data</td>
<td>0</td>
<td>fit_file_id</td>
<td>0</td>
<td>1</td>
<td>time_created</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>manufacturer</td>
<td>3</td>
</tr>
<tr>
<td>Definición</td>
<td>1</td>
<td>fit_creator</td>
<td>1</td>
<td>1</td>
<td>software_version</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>software_version</td>
<td>1.0</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>fit_creator</td>
<td>1</td>
<td>1</td>
<td>software_version</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>manufacturer</td>
<td>3</td>
</tr>
<tr>
<td>Definición</td>
<td>2</td>
<td>device_info</td>
<td>1</td>
<td>1</td>
<td>serial_number</td>
<td>1</td>
<td>num_operating_time</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>2</td>
<td>device_info</td>
<td>1</td>
<td>1</td>
<td>num_operating_time</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>manufacturer</td>
<td>3</td>
</tr>
<tr>
<td>Definición</td>
<td>3</td>
<td>user_profile</td>
<td>3</td>
<td>1</td>
<td>friendly_name</td>
<td>1</td>
<td>sleep_time</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>3</td>
<td>user_profile</td>
<td>3</td>
<td>1</td>
<td>sleep_time</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>manufacturer</td>
<td>3</td>
</tr>
<tr>
<td>Definición</td>
<td>4</td>
<td>sport</td>
<td>4</td>
<td>1</td>
<td>name</td>
<td>1</td>
<td>sport</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>4</td>
<td>sport</td>
<td>4</td>
<td>1</td>
<td>sport</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
<td>manufacturer</td>
<td>3</td>
</tr>
<tr>
<td>Definición</td>
<td>10</td>
<td>field_desc</td>
<td>10</td>
<td>1</td>
<td>field_name</td>
<td>1</td>
<td>recovery</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>10</td>
<td>field_desc</td>
<td>10</td>
<td>1</td>
<td>field_name</td>
<td>1</td>
<td>recovery</td>
<td>1</td>
<td>manufacturer</td>
<td>3</td>
</tr>
<tr>
<td>Definición</td>
<td>11</td>
<td>record</td>
<td>11</td>
<td>1</td>
<td>timestamp</td>
<td>1</td>
<td>distance</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>11</td>
<td>record</td>
<td>11</td>
<td>1</td>
<td>timestamp</td>
<td>1</td>
<td>distance</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Definición</td>
<td>11</td>
<td>record</td>
<td>11</td>
<td>1</td>
<td>timestamp</td>
<td>1</td>
<td>distance</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>11</td>
<td>record</td>
<td>11</td>
<td>1</td>
<td>timestamp</td>
<td>1</td>
<td>distance</td>
<td>1</td>
<td>unknown</td>
<td>1</td>
</tr>
</tbody>
</table>

**Tabla 3**: Fragmento archivo FIT resultante.

**Fuente**: Elaboración propia.

7. Pruebas Unitarias

Las pruebas unitarias son importantes en cualquier proyecto de software, independientemente de la plataforma donde se ejecute, ni el lenguaje de programación con el que sean escritas. El valor radica en la automatización de casos de prueba, reutilizándolos sin intervención manual en cada versión de la aplicación que se vaya publicando (o pasando a producción), además que de alguna manera también documentan comportamientos esperados.

Por los motivos mencionados, es de suma importancia que el SDK provea de las herramientas necesarias para poder crear proyectos de pruebas, y efectivamente “Toybox” lo hace desde sus versiones iniciales. Las posibilidades que brindan
dichas herramientas, son las básicas necesarias, pero suficientes para construir un proyecto de test unitarios, y automatizar la ejecución de los mismos. Dado que la sección de testing no representa un foco de estudio profundo del presente trabajo, basta con la mencionada valoración de las herramientas para pruebas unitarias, de las cuales concluimos que no tienen ninguna diferenciación con las básicas de cualquier otro lenguaje de programación, pero que cumple con lo mínimo esperado, lo cual es muy valioso en un lenguaje y entorno de desarrollo nuevo como lo es Monkey-C.

**Resultados obtenidos**

Con el transcurso de las iteraciones en el desarrollo, y el consecuente aprendizaje de las posibilidades brindadas por la tecnología, en conjunto al dominio del lenguaje de programación, dió como resultado una aplicación mucho más completa visual e internamente, con varias mejoras a nivel de legibilidad y organización del código desde las primeras versiones a la final. La mayor diferencia a nivel visual se puede apreciar en la Figura 21, que corresponde a la pantalla principal de la aplicación resultante, en donde se brinda mayor cantidad de información y mejor presentada como se preveía en la etapa de mockups, remitirse a la Figura 18 para una comparativa.
A nivel de código fuente, las primeras versiones sólo tenían algunos archivos donde se combinaba lógica de la aplicación, visualización de datos hasta persistencia en la sólo una clase. Progresivamente se fue ordenando el código aplicando diferentes patrones de diseño, la reconstrucción de todo el proyecto sucedió al identificar la posibilidad de implementar al patrón de arquitectura de software “MVC”, proveniente de Modelo – Vista – Controlador.

En este patrón arquitectónico se separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario. Existiendo un módulo que gestiona los eventos y las comunicaciones (Buschmann et al., 1996).

Los tres componentes son:

a) El modelo: Se implementa la lógica de negocio de la aplicación. Opera toda la información necesaria en el sistema, gestionando accesos, actualizaciones, consistencia en los datos, etc. Respondiendo a toda petición proveniente desde los controladores.

Ejemplo de una clase de la capa de modelo de nuestro proyecto.
class Actividad {
    var titulo;
    var tiempoActual, distanciaActual, tiempoUnidad, distanciaUnidad;
    var etapaIndice; // Valores posible: 0=Warm up, 1=Main, 2=Cool down.
    // ...
}

b) La vista: Básicamente presentan la información y lógica de negocio (los modelos), en un formato comprensible, amigable y sencillo de interactuar para el usuario final, conocido como interfaz de usuario, la cual se adapta acorde al dispositivo y sus características.

Ejemplo de una vista en el proyecto.

class ActividadView extends Ui.View {
    var width, height;
    // ...

    // Función principal para actualizar los datos en pantalla
    function onUpdate(dc) {
        View.onUpdate(dc); // Actualizar clase padre
        width = dc.getWidth(); // Ancho de pantalla del dispositivo
        height = dc.getHeight(); // Alto de pantalla del dispositivo

        // lineas separadoras
        dc.setColor(Gfx.COLOR_DK_GRAY, Gfx.COLOR_TRANSPARENT);
        dc.drawLine(0, height/6 , width, height/6);
        dc.drawLine(width/2, height/6 , width/2, height*0.75);
        // ...

        // Titulo
        var text = mActividad.titulo; // mActividad es el modelo
        var font = Gfx.FONT_MEDIUM;
        dc.drawText(width/2,0,font, text,Gfx.TEXT_JUSTIFY_CENTER);
}

c) El controlador: Hace de intermediario entre la vista y el modelo, respondiendo a eventos accionados por el usuario, e invocando al modelo cuando dicho evento impacta de alguna forma contra los datos. También responde en sentido contrario, actualizando las vistas si surge algún cambio o evento desde los datos o controlador.

Ejemplo de un controlador en el proyecto:

class ActividadDelegate extends Ui.BehaviorDelegate {
    var parentView;

    function initialize(view) {
        parentView = view;
    }

    // Cuando el controlador captura el evento onSelect (el usuario
presionó dicho botón), se ejecuta un método que actualiza la interfaz de usuario y que detiene o inicia el test en el modelo.

```javascript
function onSelect() {
    parentView.StartStopTest();
}
```

// ...

**Resumen del capítulo**

Se obtuvo el resultado esperado, la implementación técnica de una aplicación para wearable tal como representa el nombre del capítulo. Se planteó un ciclo de desarrollo ágil, iteractivo e incremental, donde las tres primeras fases fueron estáticas por tratarse de configuración de entorno de programación y definición del alcance del trabajo a desarrollar.

Posteriormente, en las fases de prototipado y de implementación de interfaces de usuario, se experimentaron varias iteraciones a medida que se avanzaba con el aprendizaje, tanto del lenguaje de programación, como de las guías de recomendaciones de experiencia de usuario. Las posibilidades que brinda la tecnología son muy amplias, se plasmaron diferentes alternativas, se ahondaron en las mismas, y se fundamentó la selección de cada una.

Generar, actualizar y persistir los datos necesarios acorde a los objetivos de la aplicación propuesta, fue el centro de las últimas fases de esta implementación. Se mantuvo la metodología de presentar alternativas a partir del aprendizaje, evaluarlas, y seleccionar una de ellas justificándola. Así se terminó construyendo la aplicación, generando conocimiento para plasmarse como buenas prácticas de programación en wearables.
Conclusiones

En cuanto a la realidad y desarrollo actual de la industria y mercado, la adopción de wearables crece exponencialmente, tendencia que se estima se mantendrá en los próximos años, lo que impactará en mayor demanda de cantidad y calidad de las aplicaciones, este fenómeno hace que cada vez hay más empresas interesadas y activas en lanzar propuestas atractivas, no sólo en ámbitos de deportes sino también en salud. La inmensa cantidad de datos que ofrecen los wearables a partir de los sensores en contacto con el cuerpo, satisfacen la tendencia de los consumidores a más y mejores métricas, que brinden información de progreso y evolución. Si bien estos dispositivos no pueden suplantar el consejo profesional de un médico o entrenador, le asisten a ellos brindándoles datos para facilitar la toma de decisiones. Sólo a partir de mediciones se puede verificar el esfuerzo real del individuo, y en ámbitos deportivos, contrastar que lo dosificado por el entrenador, sea efectivamente lo que el atleta realizó, y poder evaluar el resultado obtenido. En estas mediciones es donde los wearables toman un lugar privilegiado.

Ahondando puntualmente en los objetivos planteados en este proyecto en relación a la experiencia desarrollo de aplicaciones, se evaluaron los fabricantes más importantes de la industria, siendo seleccionado aquél que brindaba el entorno propicio para la creación de aplicaciones personalizadas. La curva de aprendizaje del lenguaje en cuestión (Monkey-C) fue muy favorable, no es un lenguaje difícil de aprender, considerando que como programador, el autor tiene la experiencia en varios lenguajes y desarrollando software para dispositivos móviles. Para atenuar el esfuerzo de programación de las interfaces con el usuario, es muy recomendable disponer del prototipado de cada pantalla antes de comenzar la programación, adaptadas a las características de visualización de cada dispositivo (relojes redondos, cuadrados, con y sin touch-screen, etcétera). Dicho esfuerzo de programación es proporcional a la cantidad de dispositivos en los cuales se va a ejecutar la aplicación, esto es por la heterogeneidad de características disímiles entre los dispositivos wearables.
Aspectos críticos a tener en cuenta, es que la experiencia de usuario en un wearable es muy distinta a la de cualquier dispositivo, se lee información pero se interactúa muy poco. En sesiones de alta intensidad, el atleta entabla muy poco contacto visual con el reloj mientras realiza la actividad, entonces la forma de comunicarle algún evento se limita a vibraciones en el reloj con intensidad y duración controladas. Otro aspecto son las limitaciones provenientes del hardware, donde hay que optimizar la utilización de recursos de procesamiento y almacenamiento, por ello es que los relojes inteligentes persisten datos en formato binario (archivos FIT), muchos más eficientes que los formatos en texto plano basados en etiquetas.

Los estándares más importantes son los protocolos FIT y ANT+, ambos ampliamente adoptados por la industria, el primero tiene una especificación muy flexible que facilitan la persistencia de datos, la desventaja de ser binario radica en complejidad para revisar, controlar y manipular los archivos. ANT+ es el punto fuerte en la integración, brinda la posibilidad de conectar y mantener comunicados entre sí varios dispositivos, que en conjunto extienden el campo de aplicación y aprovechamiento de una gran cantidad de sensores, tanto estándares como particulares o hardware específico.

Finalmente, concluimos que es posible construir aplicaciones siguiendo buenas prácticas de patrones arquitectónicos como el utilizado MVC, se logró el objetivo principal del trabajo de crear una aplicación de referencia, que obtenga datos valiosos, y que se logre experiencia plasmada en buenas prácticas, como corolario un nutrido listado de trabajos futuros propuestos.
Trabajos futuros

Durante el desarrollo del presente trabajo se fueron presentando un sinfín de opciones para extenderlo, de crear sub-proyectos o proyectos derivados. Fue tan difícil o quizás aún más, decidir qué no incluir en el trabajo para mantener el foco. La tecnología ha crecido vertiginosamente, las posibilidades de desarrollo son muy tentadoras, y particularmente en los nichos de salud, bienestar, ciencia del ejercicio, entrenamiento profesional con base científica, etc, hay muchísimas oportunidades de implementación de mejoras. Como cierre de proyecto, se presentan algunos de los próximos trabajos que el autor identificó y tiene interés de concretar, clasificados respecto de su implicancia directa o indirecta con el proyecto de esta tesis.

1) Directos a este Proyecto.
   a. Extender el alcance de este proyecto: Implementando otros tests de valoraciones de rendimiento utilizados mundialmente, y realizando, midiendo y documentando pruebas de campo con atletas y entrenadores.

   b. Integrar y conectar con otros dispositivos: Capturando en la aplicación realizada, información de potencia desde el dispositivo Stryd, con lo que se obtendría carga externa, además de la interna ya contemplada con la frecuencia cardíaca.

   c. Presentar resultados con gráficos: Curva de velocidad crítica, con la realización del actual test de 2km, más el adicional de 5km, que establecen dos puntos con los que utilizando un algoritmos científico, se traza la curva de velocidad crítica, útil para proyectar rendimientos del atleta en distintas distancias de competencia. La tecnología actual permite construir gráficos en el reloj inteligente.
2) Indirectos o proyectos relacionados en la industria.

En la actualidad es accesible adquirir dispositivos portátiles como la última generación de teléfonos celulares de Apple, o cámaras filmadoras para deportes como la VIRB de Garmin, ambas líneas de productos con apertura a poder desarrollar aplicaciones nativas que se ejecuten, o bien en su defecto, con capacidad de integración a aplicaciones propias. Estos equipos tienen capacidad de filmar a 240fps, cuando los estudios científicos se basan en lecturas entre 100 y 200fps. Esto conlleva a que una aplicación para iPhone con un costo menor a 10 USD, convierta al celular, en un laboratorio de análisis del movimiento en el bolsillo del entrenador o el atleta.

Los métodos físicos que son utilizados en grandes laboratorios, se pueden programar en una aplicación para los mencionados dispositivos, a partir de la filmación en cámara ultra lenta (240fps) que estos ofrecen. Si bien ya existen una decena de estas aplicaciones para adquirir en las tiendas virtuales, las metodologías, algoritmos, y cálculos son públicos y abiertos, brindando la posibilidad de perfeccionar la programación, o bien integrarlo en una suite de soluciones aplicadas.

A continuación algunos de los posibles trabajos:

a. Medición de fuerza, velocidad y potencia en los saltos.
   Método científico (Samozino et al., 2008).
   Una implementación actual (Apple, 2017).

b. Rendimiento de aceleración y perfil fuerza-velocidad-potencia
   Método científico (Morin et al., 2012).
   Una implementación actual (Apple, 2017b).

c. Fuerza máxima a una repetición
   Método científico (Balsalobre et al., 2017).
   Una implementación actual (Apple, 2017c).
Referencias


(Ávila, 2015). Ávila, José. “¿Qué son los wearables y cuál es su futuro?”. Espidi doctor. Documento en línea, consultado en febrero de 2017 de: http://www.espididoctor.com/que-son-wearables-futuro/


(Ounae, 2015). “Wearables cuantificadores de salud, ventajas y futuro de un arma de doble filo”. Documento en línea, consultado en febrero de 2017 de: http://ouan.e.com/wearables-cuantificadores-de-saludventajas-y-futuro-de-un-arma-de-doble-filo/


Índice

Actualización de datos.............................................................................................................76
Aplicaciones de los wearables a la salud y al deporte..................................................11
Archivos FIT, estructura y perfiles.....................................................................................33
Archivos GPX, esquema y estructura...............................................................................41
Archivos TCX, esquema y estructura...............................................................................37
Bluetooth y ANT+..................................................................................................................29
Cadencia................................................................................................................................21
Ciclo de desarrollo de software para “wearables”.........................................................57
Comandos básicos del SDK de Connect IQ.................................................................59
Comparación Monkey C con otros lenguajes...............................................................51
Comunicación entre “wearables”......................................................................................28
Consumo de oxígeno...........................................................................................................22
Definición dinámica de interfaces visuales.......................................................................69
Definición estática de interfaces visuales.........................................................................68
Deming y ciclo de la mejora continua..............................................................................23
Desarrollo en “wearables”...............................................................................................25
Despliegue Garmin Connect IQ & Push API.................................................................53
Dosificación del entrenamiento.......................................................................................15
Enfoque sistémico en el entrenamiento..........................................................................15
Especificación de la aplicación.........................................................................................61
Estructura de datos generada por “wearables”..............................................................30
Frecuencia cardíaca...........................................................................................................21
Gafas inteligentes o “Smartglasses”................................................................................10
Garmin Connect..................................................................................................................47
Garmin Connect IQ............................................................................................................48
Garmin, listado de API......................................................................................................54
Garmin, Plataforma global...............................................................................................47
Generación de datos..........................................................................................................72
Guía de diseño para “wearables”....................................................................................26
Implementación de interfaces visuales en la aplicación..........................68
Lactato en sangre..............................................................................23
Lo que no se mide no se puede mejorar........................................15
Mediciones del entrenamiento......................................................19
Mejora continua............................................................................23
Modelo de caja negra....................................................................16
Modelo MVC en desarrollo para wearables.................................82
Monkey C.......................................................................................51
Navegabilidad en la aplicación....................................................65
Necesidades del entrenamiento de resistencia..............................19
Persistencia de datos utilizando archivos FIT...............................78
Persistencia de datos utilizando propiedades...............................77
Potencia.........................................................................................20
Principales fabricantes; comparación y selección..........................45
Principales fabricantes; listado......................................................42
Protocolo ANT+ ...........................................................................28
Protocolo FIT................................................................................36
Prototipos de la aplicación...........................................................64
Pruebas unitarias............................................................................80
Pulseras inteligentes o “Smartbands”..........................................10
Relojes inteligentes o “Smartwatches”..........................................10
Ropa inteligente............................................................................11
Temperatura..................................................................................22
Tiempo y distancia.......................................................................20
Tipos de aplicaciones para wearables..........................................50
Trabajos futuros relacionados al proyecto e industria....................87
Wearables o “Ponibles”.................................................................7