UNIVERSIDAD NACIONAL DE SANTIAGO DEL ESTERO
FACULTAD DE CIENCIAS EXACTAS Y TECNOLOGÍAS

LICENCIATURA EN SISTEMAS DE INFORMACIÓN

TRABAJO FINAL DE GRADUACIÓN

Prototipo de sistema móvil para e-turismo

Autor

Pablo Javier Najar Ruiz

Profesor Guía

MSc. Lic. Susana Isabel Herrera

Año

Octubre, 2013

UNIVERSIDAD NACIONAL DE SANTIAGO DEL ESTERO
FACULTAD DE CIENCIAS EXACTAS Y TECNOLOGÍAS

TRABAJO FINAL DE GRADUACIÓN DE LA LICENCIATURA EN SISTEMAS DE INFORMACIÓN

Prototipo de sistema móvil para e-turismo

Autor

Pablo Javier Najar Ruiz

Profesor Guía

MSc. Lic. Susana Isabel Herrera

Asesor

Ing. Eugenio Ledesma

Aprobado el día ..... del mes de ..... del año 20.....
por el Tribunal integrado por:

....

(firma) (firma) (firma)

....

(aclaración) (aclaración) (aclaración)

Santiago del Estero – Argentina

Dedico mi trabajo final a mis amores y en especial a quien con su amor, paciencia, comprensión y consejos, supo sacrificar su tiempo para que yo pudiera cumplir con mis proyectos. Dándome en esta etapa a lo que son la luz de mis ojos.

Romina, Patricio y Máxima

Agradecimientos

A mi Madre la mujer más especial, que con su simpleza y amor me enseño los valores de la vida.

A mi Padre Carlos y Hermanos Verónica, Ana y Carlos que me dieron todo en la vida para que yo pudiera lograr mis sueños.

A mis Abuelos Juan Pablo y Margarita y Tía Leoni que forjaron con ahínco mi carácter para que pudiera terminar no solamente este trabajo, sino, toda mi carrera.

A mis Tías Analia, Pichy, Hetty, Midri, Cira y Marta y a la Familia Buitrago, a todos ellos, por estar siempre brindándome su cariño y amor sin límites.

En especial, a mi guía y apoyo, Susana Herrera, que supo comprender mis tiempos y me motivó a concluir este proyecto ofreciéndome incondicionalmente su tiempo y sabiduría.

A la Dra. Cecilia Challiol, que generosamente me brindo su tiempo y sus invalorables consejos.

A los integrantes del Proyecto de Investigación, en especial a Sergio Rocabado por su excelente predisposición y valiosos consejos.

Al "Master", Eugenio Ledesma, mi maestro, el que me abrió el camino, me enseño y me acompaño en mi desarrollo profesional.

A mis compañeros y amigos de carrera, en especial a Javier, Sergio, Carlos, Sebastián y Mema con los que juntos transitamos esta hermosa etapa de mi vida.

A mis Amigos del Alma quienes siempre están presentes en todo momento, en especial para festejar mis logros.

Y a Dios por iluminarme el camino para salir adelante en los momentos difíciles y recordarme todos los días que tengo mucho por quien vivir y ser feliz.

A todos, Gracias!!

Pablo Javier Najar Ruiz

ÍNDICE

RESUMEN

En el presente trabajo se presentan los resultados de una investigación sobre la eficiencia de los sistemas de información móviles. La motivación surge a partir de la necesidad de contar con sistemas que brinden información sobre los sitios y circuitos turísticos de una ciudad (lugares y eventos culturales, folclóricos y religiosos), en forma ubicua y móvil, a turistas provenientes de otras regiones. Es por ello que los objetos de estudio son aplicaciones móviles turísticas o aplicaciones de m-turismo.

Se trabaja con un tipo particular de aplicaciones móviles, las basadas en posicionamiento, en las cuales la posición del usuario es la principal variable. A partir de ella, las aplicaciones brindan diversos servicios al usuario. Para lograr ubicuidad se apoyan en el uso de dispositivos móviles que poseen recursos limitados en cuanto a capacidad de procesamiento, memoria, tamaño de pantalla. De acuerdo a cómo se diseña la aplicación, es factible optimizar el uso de estos recursos, mejorando la eficiencia. En particular, en este trabajo se estudia la relación existente entre las arquitecturas de diseño y la eficiencia (en cuanto al uso de recursos) de una aplicación móvil. Las arquitecturas consideradas son: arquitectura híbrida (aplicación nativa Android y servicios web) y arquitectura web.

Siendo la eficiencia una característica de la calidad, se la evalúa usando la Norma ISO/IEC 25000 y el proceso de evaluación GOCAME, diseñado en la Universidad Nacional de La Pampa. Para la medición, se desarrollaron dos prototipos de m-turismo usando la metodología XP, uno con arquitectura híbrida y otro con arquitectura web. Las pruebas se llevan a cabo tanto en ambiente Wi-Fi como 3G.

Los resultados muestran que las aplicaciones móviles con arquitectura híbrida son más eficientes que aquellas con arquitectura web. Además, este trabajo constituye un aporte a la medición de la calidad de aplicaciones dado que, al no encontrar antecedentes sobre la medición de la eficiencia, se definieron las propias métricas, indicadores elementales e indicadores globales necesarios en el proceso GOCAME.

PALABRAS CLAVES

Aplicaciones móviles, arquitecturas de diseño, eficiencia de aplicaciones móviles, evaluación de la calidad de aplicaciones, GOCAME.

1. CAPÍTULO 1: PROBLEMA, OBJETIVOS Y METODOLOGÍA.

1.1. INTRODUCCIÓN

La sociedad se encuentra en un momento de grandes cambios, producto del consumismo y de un *bombardeo* tecnológico, los cuales, a diferencia de otros tiempos tienen el distintivo de producirse a gran velocidad y en algunos aspectos en forma desorganizada. Internet y la red de telefonía móvil como así también las grandes innovaciones a nivel tecnológico inciden directamente en las formas de comunicarse, trabajar, producir, vender, comprar e incluso de recrearse de las personas. La llamada *globalización* toma fuerza y aparecen en la escena mundial las nuevas Tecnologías de la Información y Comunicación (TICs). La humanidad está viviendo la *era de las comunicaciones* donde el *estar conectados* está convirtiéndose en un actividad fundamental del ser humano.

El turismo, en su conjunto, no es ajeno a este contexto de grandes y rápidas transformaciones. Dicha actividad es una de las más beneficiadas por este escenario tecnológico, el cual le permite un desarrollo inusitado en la administración y puesta en marcha en muchos lugares del mundo. Los centros turísticos tomaron conciencia que el turista necesita información dinámica, clara y rápida para tomar una decisión y utilizaron a la Informática como la base de este restructuración en la actividad turística. El análisis del sector pone de manifiesto que la provincia de Santiago del Estero (Argentina), a pesar de estar relacionada con un turismo religioso y folclórico, presenta un gran potencial en crecimiento, impulsado por el estado provincial y nacional con obras e infraestructura en los principales centros turísticos y la promoción de eventos folclóricos, deportivos y culturales.

Las aplicaciones móviles son uno de los pilares fundamentales en el fomento del turismo en numerosos lugares del mundo. Particularmente, han tenido éxito las aplicaciones sensibles al contexto y basadas en posicionamiento, en las cuales la posición del usuario es la variable principal. A partir de ésta, las aplicaciones brindan diversos servicios al usuario, teniendo en cuenta su contexto social y preferencial. Estas aplicaciones se ejecutan sobre dispositivos móviles, con diferentes sistemas operativos (SO), que poseen recursos limitados en cuanto a capacidad de procesamiento, memoria, duración de la batería, entre otros aspectos. Por ello, el desarrollo de las aplicaciones móviles involucra cuestiones propias e importantes al momento de utilizarlas, como ser: el diseño de interacción, posicionamiento, representación del espacio, formas de sensado, tratamiento del contexto, arquitecturas y herramientas de desarrollo e implementación. Todas estas cuestiones constituyen las variables que impactan directamente en la calidad de la aplicación que se desarrolla. No es fácil obtener una aplicación usable cuando la pantalla, memoria, velocidad y conexión a internet son limitados.

En este trabajo se investiga sobre la calidad de las aplicaciones móviles basadas en posicionamiento, estudiando para ello la relación que existe entre la arquitectura y la eficiencia de estas aplicaciones. A

tal efecto se desarrollaron prototipos de m-turismo (aplicaciones móviles de turismo) con arquitectura híbrida y arquitectura web. El objetivo principal consistió en determinar cuál arquitectura es más eficiente, considerando un entorno móvil específico en un momento y zona dados.

Por otra parte, con el propósito de evaluar la eficiencia de las aplicaciones móviles con diferentes arquitecturas, se aplicó el método GOCAME[1] -utilizado para la medición de la calidad de aplicaciones en general-. Para hacer esto posible se diseñaron y aplicaron métricas e indicadores elementales para los atributos de la eficiencia; así como también indicadores globales para las subcaracterísticas de la eficiencia.

El trabajo se desarrolló en el marco del Proyecto de Investigación del Instituto de Investigaciones en Informática y Sistemas de Información (IIISI) de la UNSE denominado *Optimización de la calidad de los Sistemas Móviles mediante la implementación de nuevas arquitecturas, realidad aumentada, técnicas de visualización y redes móviles Ad-Hoc*.

1.2. PLANTEAMIENTO DEL PROBLEMA

El problema surge a partir de observaciones del entorno donde se radica esta investigación, advirtiendo la necesidad de desarrollar aplicaciones móviles eficientes para los turistas que visitan la ciudad que mayor turismo recibe en la Provincia de Santiago del Estero, es decir, la ciudad de Termas de Río Hondo. A partir de esto, la atención se concentra en las características que se deberían revisar para lograr el desarrollo e implantación exitosa de soluciones informáticas móviles turísticas en esta ciudad. Se considera el éxito en función de la eficiencia. Se consideran aplicaciones móviles a las que se ejecutan en tabletas o en teléfonos inteligentes o *smartphones*.

Lograr aplicaciones móviles eficientes constituye un verdadero desafío. La limitación se debe principalmente a los escasos recursos del dispositivo donde se ejecuta la aplicación. Es difícil lograr aplicaciones veloces cuando los procesadores no son potentes y la capacidad de la memoria principal es baja. Más difícil aún es lograr aplicaciones interactivas cuando el tamaño de la pantalla y del teclado es pequeño o cuando el uso agota rápidamente la batería del dispositivo. Otra dificultad es lograr aplicaciones sensibles al contexto interactivas (en particular sensibles a la posición) cuando la conexión a Internet no es estable (cae) o es lenta. A esto se le complementa la dificultad de desarrollar aplicaciones compatibles con dispositivos con sistemas operativos diferentes.

[1] GOCAME es un método de evaluación y medición de la calidad propuesto por el grupo de investigación de calidad dirigido por el Dr. Luis Olsina en la Universidad Nacional de La Pampa.

En esta investigación se considera que la principal variable que influye sobre la eficiencia de una aplicación móvil es la arquitectura que utiliza el desarrollador en la etapa de diseño. Si bien esta arquitectura depende del entorno móvil que caracteriza una determinada época y lugar (tipo de conexión del país, sistema operativo, velocidad del procesador, capacidad de la memoria, duración de la batería), esta investigación se limita al estudio de arquitecturas alternativas. A continuación se presentan las mismas:

- **Arquitectura Cliente-Servidor**: Parte de la aplicación en el cliente y parte en el servidor. La aplicación cliente consulta al servidor a través del uso, por ejemplo, de servicios web. Se tiene en cuenta características del Sistema Operativo y características del dispositivo donde se instalará la aplicación cliente. Permite interactuar con los periféricos del dispositivo (cámara, bluetooth, GPS); esto, a su vez, permite determinar los datos de contexto relacionados al dispositivo móvil, por ejemplo, la posición. En este trabajo, se la denomina ***arquitectura híbrida*** para diferenciarla de la arquitectura cliente-servidor web.
- **Arquitectura Web:** si bien también es arquitectura cliente-servidor, la aplicación en este caso reside completamente en el servidor Web y es accedida a través del browser del dispositivo móvil. Puede ser accedida por la mayoría de los mismos. Son idénticas a las páginas Web que uno consulta a diario que son para browser de escritorios, sólo que se las adapta a dispositivos móviles. Necesita de conectividad a Internet/Wan/Lan para su funcionamiento. Desde el browser, se dificulta la interacción con los periféricos del dispositivo, por ello se debe utilizar otras tecnologías como por ejemplo el uso de códigos QR para la determinación de la posición del dispositivo.
- **Arquitectura Cliente:** la aplicación reside completamente en el cliente. En este caso es fundamental determinar las características, la plataforma (sistema operativo) y especificaciones del dispositivo donde se va a implementar la aplicación cliente. En muchos casos no necesita conectividad a Internet/Wan/Lan para su funcionamiento. Permite interactuar con los periféricos del dispositivo (cámara, bluetooth, GPS); esto, a su vez, permite determinar los datos de contexto relacionados al dispositivo móvil.

En 2009 había al menos siete sistemas operativos para dispositivos móviles diferentes en el mundo. Entre los más conocidos se encuentran: Symbian de Nokia, BlacKBerry OS, iPhone OS de Apple, Windows Phone de Microsoft. Sin embargo, el sistema operativo que vertiginosamente se impuso sobre los otros es Android de Google. Para contrarrestar esto, en Febrero del 2011 Microsoft y Nokia realizaron una alianza estratégica a partir de la cual los smartphones de Nokia usan el SO Windows Phone. Sin embargo, actualmente en el mercado de smartphones y tabletas continúa prevaleciendo Android. La adquisición de Motorola Mobile por parte de Google, en Agosto del 2011, fue uno de los hechos que contribuyó al dominio de Android. Y constituye una tendencia que, por el momento, es irreversible.

Considerando que Android es el SO que domina el mercado, este trabajo se limita a esta plataforma en el caso de las arquitecturas híbridas. Además, considerando que las aplicaciones m-turismo necesitan conexión a Internet, se consideran solamente las arquitecturas híbrida y web. Por otra parte, se toma como marco empírico el turismo de la ciudad de Termas de Río Hondo.

Por todo lo mencionado en este apartado, se formula el problema de la siguiente manera:

¿Cómo desarrollar aplicaciones móviles para turismo eficientes?

¿Cuál arquitectura es más conveniente utilizar? ¿Híbrida o Web?

Y además: ¿Cómo se puede evaluar la eficiencia de este tipo de aplicaciones?

1.3. OBJETIVOS

1.3.1. GENERALES

- Contribuir a la optimización de la eficiencia de las aplicaciones móviles mediante el estudio comparativo de arquitecturas alternativas.
- Contribuir a la promoción del turismo mediante el desarrollo de aplicaciones que se ejecuten eficientemente en dispositivos móviles.

1.3.2. ESPECÍFICOS

- Analizar el entorno móvil de una aplicación de m-turismo (dispositivos, SO, conectividad, servicios habilitados sobre la red de telefonía móvil) para la ciudad Termas de Río Hondo, provincia de Santiago del Estero, años 2012-2013.
- Diseñar modelos que permitan identificar los elementos y relaciones de las arquitecturas híbrida y web.
- Desarrollar prototipos de aplicaciones de m-turismo en base a las arquitecturas bajo estudio, utilizando una metodología ágil y herramientas específicas para el desarrollo de aplicaciones móviles (lenguajes de programación, librerías, SDK, emuladores de dispositivos, etc.).
- Diseñar métricas, indicadores elementales e indicadores globales que permitan medir atributos y subcaracterísticas de la eficiencia, que cumplan con la norma ISO/IEC 25000.
- Evaluar la eficiencia de aplicaciones de m-turismo, siguiendo un proceso robusto que involucre los prototipos, métricas e indicadores diseñados.

1.4. METODOLOGÍA

Para el desarrollo del trabajo se siguieron las etapas que se detallan debajo. Estas fueron modificadas mínimamente respecto a la propuesta inicial, con el propósito de lograr efectivamente los objetivos mencionados en el apartado anterior.

Etapa 1 - Investigación bibliográfica.

En esta etapa se estudiaron aspectos generales relacionados al desarrollo de aplicaciones móviles, abarcando las arquitecturas, rendimientos, costos, documentación y el impacto de los mismos en el desarrollo de aplicaciones móviles. Se exploraron protocolos de comunicación, metodologías actuales, SDKs, librerías y lenguajes de programación; así como también los estándares de calidad, principalmente las normas ISO/IEC 25000 (International Standar Organization, 2013).

Etapa 2 - Construcción del marco referencial.

Se organizaron en forma lógica y secuencial los elementos teóricos y tecnológicos derivados de la información recabada en fuentes fidedignas, y que sirvieron de base para proponer la solución al problema planteado. Se estudiaron normas de calidad y las variables relacionadas la eficiencia de una aplicación móvil. Se exploran las aplicaciones que permiten medir dichas variables.

Etapa 3 - Relevamiento del marco empírico.

Se realizaron entrevistas a especialistas en Turismo, visitas a Centros de Turismo de la Provincia de Santiago del Estero. Esto permitió definir posteriormente las historias de usuario de los prototipos.

Etapa 4 – Análisis del Entorno Móvil.

Se realizó un estudio sobre la conectividad (datos y voz) que poseen las redes móviles en la ciudad de Santiago del Estero y Termas de Río Hondo. Se evaluó el equipamiento (dispositivos, smartphones, tabletas, etc.) y la manera en que éstos pueden realizar una conexión estable a la base de datos de puntos de interés.

Etapa 5 – Definición de arquitecturas alternativas.

Se modelizaron las arquitecturas híbrida y web. Para ello, se analizaron sus diferencias y particularidades como así también las funcionalidades que comparten en una solución completa, como lo son el uso aplicaciones de terceros y el procesamiento e identificación de puntos de interés. Las arquitecturas se vinculan directamente a la tecnología utilizada en el desarrollo de la aplicación móvil, la manera en que ésta accede a la base de datos de puntos de interés y la forma en que se desenvuelve en un ambiente móvil cambiante para dar respuesta a las necesidades de los usuarios móviles.

Etapa 6 – Desarrollo de prototipos para las diversas arquitecturas.

Se desarrollaron dos prototipos de aplicaciones de m-turismo que utilizan los modelos diseñados en la etapa anterior, utilizando metodología ágil *Programación Extrema (XP)* y buenas prácticas de diseño. Se llevaron a cabo las siguientes sub-etapas.

Sub-Etapa 6.1 - Planificación. Determinación de las *historias de los usuarios* a partir de la información relevada en la Etapa 3. Se definieron las características y funcionalidades a desarrollar.

Sub-Etapa 6.2 - Diseño. Diseño sencillo del prototipo del sistema a desarrollar basado en las arquitecturas definidas en la Etapa 5.

Sub-Etapa 6.3 – Codificación y prueba. Codificación de las historias de los usuarios, usando las herramientas definidas en la Etapa 2. Diseño y ejecución de pruebas unitarias y de integración, basadas en la funcionalidad del sistema.

Etapa 7 – Diseño de la evaluación de la eficiencia de las arquitecturas.

En base al proceso GOCAME (Lew, Olsina, Becker, & Zhang, 2011) se definieron los pasos a seguir para evaluar la *eficiencia* como una característica de la *calidad*. Se determinaron los escenarios de prueba: ambiente Wi-Fi y ambiente 3G, describiendo los componentes de cada ambiente. Se definieron los casos de prueba a tomar en base a los Puntos de Interés característicos de una aplicación m-turismo que podrían tener comportamiento diferente en cuanto al uso de recursos del dispositivo (impactando sobre la eficiencia). Se definieron las subcaracterísticas de la eficiencia a medir y los atributos de cada subcaracterística. Se diseñaron las métricas para medir cada atributo. Se diseñaron los indicadores elementales que permitieron posteriormente interpretar las mediciones de cada atributo. Se diseñaron los indicadores globales que permitieron posteriormente evaluar la eficiencia.

Etapa 8 – Evaluación de la eficiencia de las arquitecturas.

También siguiendo el proceso GOCAME se llevaron a cabo las mediciones (aplicando las métricas e indicadores elementales definidos) de los atributos. Luego se evaluaron las subcaracterísticas y finalmente la eficiencia de cada prototipo correspondiente a arquitectura híbrida y web. Para ello se utilizaron los software implicados en las métricas (PowerTutor, TrafficMonitor, etc.), se registraron los datos obtenidos en tablas y se capturaron pantallas de los software de medición.

Etapa 9 - Análisis de resultados y elaboración de conclusiones.

Se analizó la información obtenida en la etapa anterior, se compararon las gráficas arrojadas así como también los valores de los indicadores de cada prototipo. Posteriormente, se elaboraron conclusiones y se definieron líneas futuras de investigación relacionadas con esta temática.

1.5. PUBLICACIONES

Los resultados parciales que se fueron obteniendo durante el desarrollo de la propuesta fueron publicados en las memorias de congresos de Informática y de Sistemas. Los artículos son:

- *Marco Sistémico para el Desarrollo de Aplicaciones de m-turismo.* Herrera, S. I., Najar Ruiz, P. J., Contreras, N., Fénnema, C., Lara, C. Anales del 9° Congresso Brasileiro de Sistemas. ISBN: 978-85-89102-43-8. Palmas, Brasil, Octubre 2013.
- *Evaluación de la calidad en aplicaciones móviles.* Herrera, S. I., Najar Ruiz, P. J., Palavecino, R., Goñi, J. IX Jornadas de Ciencia y Tecnología de Facultades de Ingeniería del NOA. Santiago del Estero, Octubre 2013.
- *Sistema de Información Móvil para Turismo Receptivo.* Herrera, S. I., Najar Ruiz, P. J., Ledesma, E., Rocabado, S. Revista Gestao e Conhecimento, Edición Especial, Anales del 8° Congresso Brasileiro de Sistemas. ISSN 1808-6594. Pozo de Caldas, Brasil, Octubre 2012.

1.6. ESTRUCTURA DEL DOCUMENTO

Este trabajo se estructura, en adelante, de la siguiente manera.

En el Capítulo 2 se presentan los marcos referenciales. En el marco teórico se describen las arquitecturas de software tradicionales, los servicios web, normas y procesos de evaluación de la calidad del software y la eficiencia en aplicaciones móviles. En el marco tecnológico se describe el SO Android, el ciclo de vida de las aplicaciones que corren en el mismo, la interacción de las aplicaciones Android con librerías para el consumo de servicios web, códigos QR y el funcionamiento de los programas de lectura (elemental en los sistemas basados en posicionamiento), también se presenta el entorno tecnológico móvil de la región (tecnologías usadas por los proveedores de servicios de telefonía móvil y dispositivos móviles más usados).

En el Capítulo 3 se presentan modelos concretos de implementación de las arquitecturas propuestas (híbrida y web). Además, se realiza una comparación analizando rasgos comunes entre ambas.

En el Capítulo 4 se exponen las primeras actividades realizadas para la evaluación de la eficiencia de aplicaciones con arquitecturas híbrida y web. En general, estas actividades están relacionadas con la preparación de: prototipos, ambientes de prueba, casos de prueba, métricas e indicadores. Con mayor detalle, se presentan:

- Desarrollo de los prototipos de aplicaciones con arquitectura híbrida y web

- Diseño y preparación de los escenarios de prueba y definición de los casos de prueba
- Evaluación de la calidad de los prototipos, siguiendo el proceso GOCAME (las primeras etapas):
    - Definición de los requisitos no funcionales o NFR
    - Diseño de las mediciones (el conjunto de métricas para medir atributos)
    - Diseño de la evaluación (el conjunto de indicadores para evaluar subcaracterísticas y características)

En el Capítulo 5 se sintetizan los resultados del trabajo de campo, es decir, la información obtenida en las mediciones y evaluación de la eficiencia (últimas etapas del proceso GOCAME). Se utiliza el prototipo, los ambientes y casos de prueba, las métricas e indicadores para obtener la información.

En el Capítulo 6 se presenta la discusión sobre los resultados obtenidos en el proceso de medición y evaluación de la eficiencia de aplicaciones móviles con arquitectura híbrida y arquitectura web.

En el Capítulo 7 se presentan las conclusiones y se proponen futuras líneas de investigación relacionadas con la temática.

Además, este trabajo presenta cinco Anexos en los cuales el lector encontrará la información que se detalla en los siguientes párrafos.

En cuanto al marco empírico referido a la actividad turística en general y al turismo en Santiago del Estero en particular, se lo describe en el Anexo 1. Por otra parte, en el Anexo 2 se adjunta la entrevista a prestadores de servicio de comunicaciones móviles en la región, en base a la cual se describió el entorno tecnológico. Ambos Anexos son referenciados desde el Capítulo 2.

En el Anexo 3 se presenta la documentación del desarrollo de los prototipos siguiendo la metodología XP, presentando las historias de usuario y las actividades del desarrollo. Mientras que en el Anexo 4 se presentan ejemplos del código desarrollado en Java, las principales funciones o actividades. Ambos Anexos son referenciados desde el Capítulo 4.

Finalmente, en el Anexo 5 están transcriptos ejemplos de los archivos .log de las mediciones realizadas con el software PowerTutor. Este Anexo está referenciado desde el Capítulo 5.

2. CAPÍTULO 2: MARCOS REFERENCIALES.

En este capítulo se presentan las teorías y tecnologías que sustentan el estudio realizado en el trabajo. Las primeras se presentan en el Marco Teórico mientras que las segundas en el Marco Tecnológico. En cuanto al Marco Empírico, donde se abordan conceptos referidos al Turismo, se presenta en el Anexo 1, dado que se trata de conceptos complementarios que no pertenecen al dominio de la Informática.

2.1. MARCO TEÓRICO

El lector encontrará en este apartado los conceptos sobre Computación Móvil, específicamente lo referido a aplicaciones móviles. Entre éstos se encuentran: arquitecturas, mecanismos de sensado de posición, servicios web y el Patrón Modelo Vista Controlador.

Por otra parte, también se presentan los estándares de calidad del software y los procesos para su evaluación y medición. Luego, en particular, se trata lo referido a la característica *eficiencia* en aplicaciones móviles.

2.1.1. COMPUTACIÓN MÓVIL

Computación Móvil es un término genérico que describe la habilidad para usar tecnología sin ataduras, es decir, no conectada físicamente o que pertenece a entornos remotos o móviles, no estáticos (Lawrence, Pernici, & Krogstie, 2004). Para una mayor comprensión se citan tres definiciones que serán de gran ayuda para poder entender qué es lo que involucra la computación móvil:

- "Se puede definir la Computación Móvil como el entorno de computación sobre la movilidad física. En este entorno, los usuarios pueden acceder a datos, información o cualquier otro objeto lógico desde cualquier dispositivo mediante cualquier red mientras se va moviendo" (Talukder, Ahmed, & Yavagal, 2010)
- "La Computación Móvil se puede definir como un servicio que se mueve con las personas, el cual permite que las mismas se puedan inscribir, recordar, comunicar y razonar independientemente de la posición de los dispositivos" (Lyytinen & Yoo, 2002).
- "La Computación Móvil puede ser definida como la computación que se produce cuando usuario está usando un dispositivo móvil al mismo tiempo que se mueve. La ubicación del usuario es independiente y se puede comunicar a través de una red wireless" (Roy, Scheepers, & Kendall, 2003).

Dicho concepto, como está visto, tiene diversos enfoques y aspectos a tener en cuenta. Este trabajo se enfoca en las aplicaciones móviles y, en particular, se investiga sobre las aplicaciones móviles basadas en posicionamiento. Dichas aplicaciones son las que usan LBS (Servicios Basados en Localización) donde los mismos proveen la posibilidad de encontrar la ubicación geográfica de un dispositivo móvil con el fin de proveerle servicios basados en esa ubicación (Open Geospatial Consortium, 2012). En otras palabras, son servicios que integran una localización o ubicación de un dispositivo móvil con otra información para proveer un valor agregado a un usuario (Schiller, 2004).

Para la creación de una aplicación móvil se deben considerar distintos factores como se mencionó en (Challiol, 2012). Entre éstos se encuentran:

- Modelo de Contexto
- Modelo de Usuario
- Representación del Espacio
- Dinámica de la Aplicación
- Representación de los Puntos de Interés (PI)
- Modelo de Dominio
- ***Mecanismos de Sensado***
- ***Arquitectura de la Aplicación.***

En este trabajo se consideraron cuestiones referidas a las diferentes arquitecturas de las aplicaciones móviles y a un mecanismo de sensado particular de posición (para determinar la posición del dispositivo móvil, por ende, del usuario). La arquitectura determina el esquema de la aplicación y como se distribuyen los demás conceptos.

2.1.2. ARQUITECTURA DE SOFTWARE CLIENTE-SERVIDOR Y MONOLÍTICA (CLIENTE)

En la década de los 60's a los 80's la computación estaba acostumbrada a formar estructuras computacionales de procesamiento centralizado donde se encontraban Mainframes y Terminales *Boba*s; estas últimas sólo se utilizaban para ingreso y visualización de datos y despliegue de la interfaz. Todo el procesamiento lo llevaba a cabo las Mainframes, lo cual generaba en muchas casos una congestión del canal de comunicación provocando el colapso del mismo y, por ende, el mal funcionamiento de los sistemas informáticos. Con la aparición de la computadora personal (PC), de las redes y de los Sistemas de Gestión de Base de Datos, se comenzó a transformar esos ambientes centralizados en distribuidos, donde los programas y las base de datos están ubicados en uno o más computadoras o servidores. Esto permitió la descentralización del procesamiento y de los recursos de cada uno de los servicios. De esta manera, se introdujo el concepto de arquitectura *Cliente-servidor,*

una forma de Informática distribuida que separa las funciones en dos partes distintas bien definidas. Una de ellas el *Cliente* (Front-End), que hace peticiones. Y el *servidor* (Back-End), que separa pedidos de sus clientes y los satisface. Ambos programas pueden residir en la misma máquina. Todo esto es propiciado por el *Middleware*, encargado de enlazar el cliente con el servidor y permitir que un cliente obtenga un servicio de un servidor a través, en la mayoría de los casos, de una red de comunicación. Dicha definición se puede apreciar de manera gráfica en la figura 2.1.

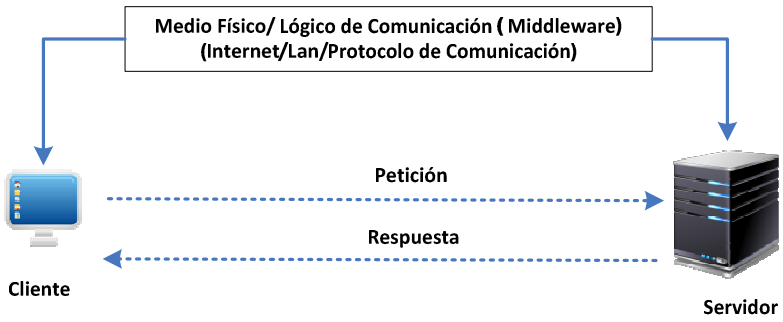


Figura 2.1 - Arquitectura *Cliente-servidor* (General).

En una Arquitectura *Cliente-servidor,* están involucrados básicamente tres componentes: la lógica de negocios, los datos y la interfaz de usuario de la aplicación. Dependiendo dónde se concentran estos componentes, se tienen *Clientes Gordos* (mayor concentración de la lógica en la aplicación cliente) o *Servidores Gordos* (mayor concentración de la lógica en el servidor). Un caso particular de *servidores gordos* se da en las Arquitecturas Web donde los tres componentes mencionados radican en el servidor. En la figura 2.2 se puede apreciar los tres componentes mencionados y cómo éstos están relacionados entre sí; asimismo se puede apreciar que la lógica de negocios es la que permite la comunicación entre los datos y la interfaz del usuario.

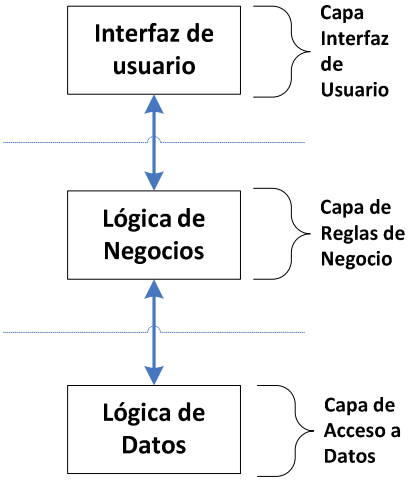


Figura 2.2 - Niveles o etapas de una Arquitectura Cliente-servidor (Users, 2009).

También se pueden tener una arquitectura cliente-servidor nivelada, donde no se tiene un *cliente gordo* ni un *servidor gordor*. Este caso se puede apreciar en la figura 2.3, donde la interfaz de usuario se encuentra del lado del cliente, la lógica de datos del lado del servidor, y la lógica de negocios está repartida entre el cliente y el servidor.

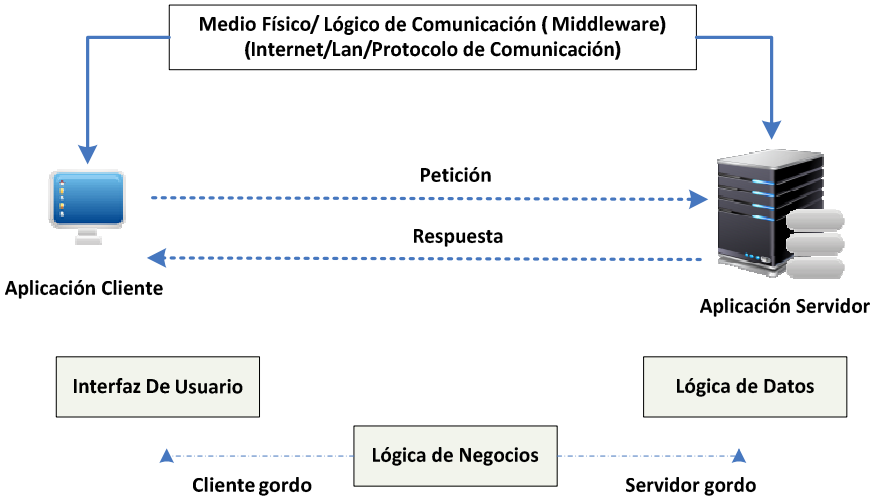


Figura 2.3 - Arquitectura de software cliente-servidor (Users, 2009).

En el caso de las arquitecturas cliente, los tres componentes (la lógica de negocios y datos de la aplicación como así también la interfaz de usuario) están en el cliente, y hay ausencia de una aplicación servidor. Es decir, es una arquitectura de software en un sólo bloque, *monolítica* (cliente). El esquema de esta arquitectura se aprecia en la figura 2.4.

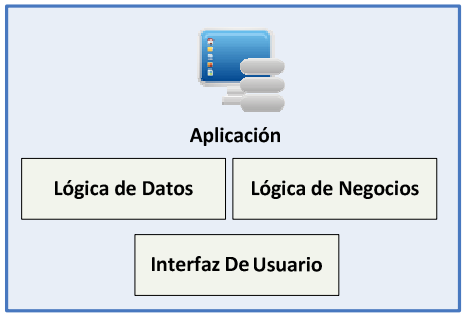


Figura 2.4 - Arquitectura de software Monolítica (Cliente).

Cada una de estas arquitecturas podría interactuar con servicios externos; por ejemplo, mediante el uso de *servicios web*, aprovechando de esta manera información o comportamientos ya existentes.

Por último, el objetivo de las diferentes arquitecturas consiste en favorecer la mayor cantidad de características del software, desde el alto rendimiento hasta la facilidad de mantenimiento, la extensibilidad (suma de elementos al software) y la escalabilidad (crecimiento del software).

2.1.2.1. CLASIFICACIÓN DE ARQUITECTURAS MÓVILES

Las arquitectura cliente-servidor y cliente mostradas en las figuras 2.3 y 2.4, pueden ser aplicadas a aplicaciones móviles, donde en particular, la aplicación cliente va a residir en el dispositivo y debe ser compatible con el mismo.

A continuación se presentan tres arquitecturas que se pueden implementar en aplicaciones móviles. Dos de ellas son arquitecturas cliente-servidor, una denominada *Arquitectura Híbrida* (donde hay un balance entre lo que reside en el cliente y en el servidor) y otra *Arquitectura Web* (donde se tiene un *servidor gordo*). También se presenta una arquitectura cliente.

- Arquitectura Híbrida (cliente-servidor): Parte de la aplicación en el cliente y parte en el servidor. El aplicativo cliente consulta el aplicativo servidor (servicios web) a través del uso de librerías. Se tiene en cuenta características del Sistema Operativo y características del dispositivo donde se instalará la aplicación Cliente, la cual permite interactuar con los periféricos del dispositivo (cámaras, Bluetooth, GPS) que posibilitan el sensado de los datos de contexto en el que se encuentra el móvil. Es necesario conectividad (Internet, Wan, Lan, etc.) entre las dos aplicaciones para su correcto funcionamiento. La figura 2.5 contiene la modelización de esta arquitectura.

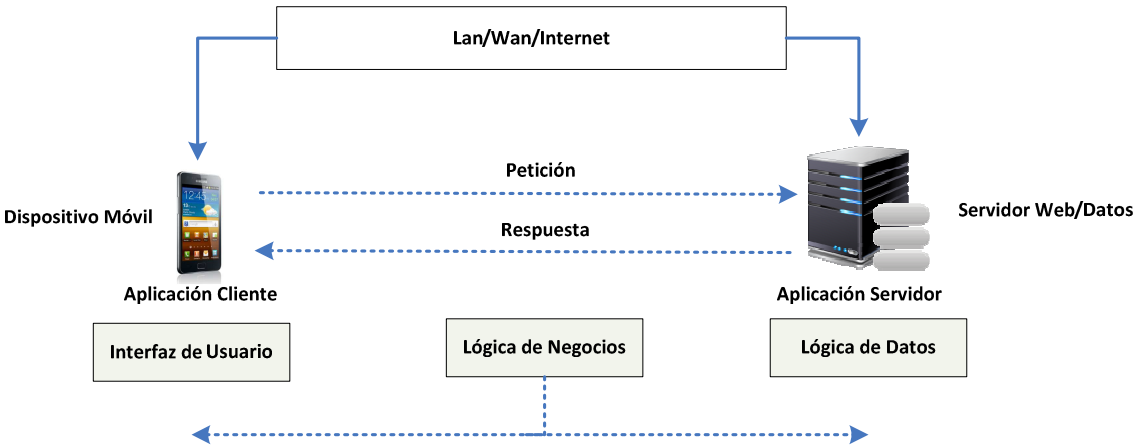


Figura 2.5 - Modelo de la Arquitectura Híbrida (Cliente-servidor).

- Arquitectura Web: La aplicación reside completamente en el servidor Web y es accedido a través del browser del dispositivo. Puede ser accedida por la mayoría de los dispositivos móviles. Son idénticas a las páginas web que se consultan a diario mediante browsers de escritorio, sólo que se las adapta para dispositivos móviles. Se dificulta la interacción con los periféricos del dispositivo, por ello se debe utilizar otras tecnologías como por ejemplo el uso de lectores de etiquetas (Ej.: Códigos QR) para la determinación del contexto de localización. Es necesario conectividad (Internet, Wan, Lan, etc.) entre el browser y la aplicación web para

su correcto funcionamiento. El modelo de esta arquitectura se puede visualizar en la figura 2.6.

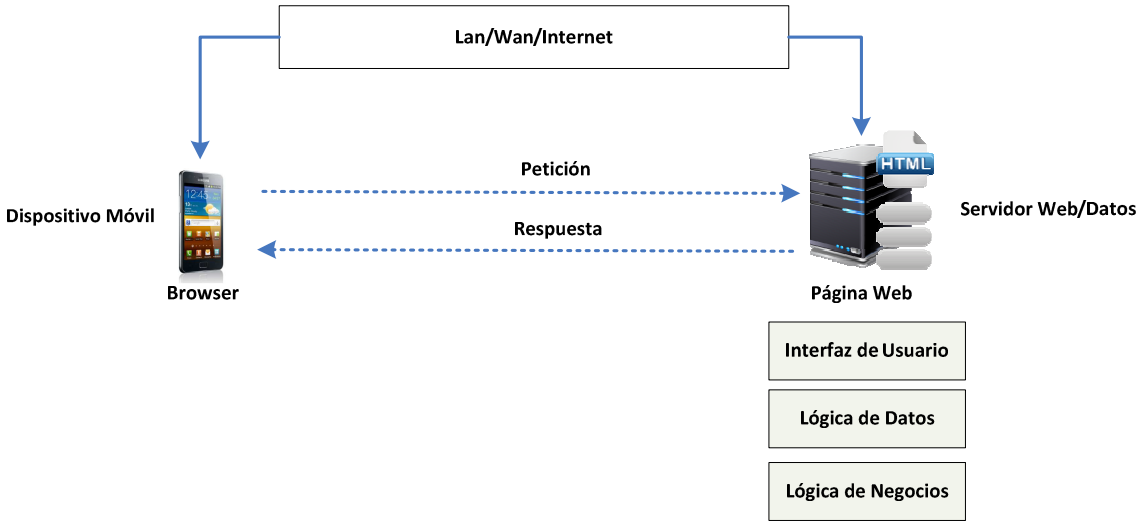


Figura 2.6 - Modelo de la Arquitectura Web.

- Arquitectura Cliente: en el caso particular de aplicaciones móviles, el cliente debe ser una aplicación que se pueda instalar en el dispositivo. La cual debe considerar las características del Sistema Operativo y características propias del dispositivo. La aplicación Cliente puede interactuar con los periféricos del dispositivo (cámaras, bluetooth, GPS) que permitirán determinar los datos de contexto en el que se encuentra el dispositivo móvil. El modelo de esta arquitectura se puede visualizar en la figura 2.7.

Figura 2.7 - Modelo de Arquitectura Cliente.

Como se mencionó para las arquitecturas en general, las arquitecturas relacionadas con aplicaciones móviles (Híbrida, Web y Cliente) pueden interactuar con servicios permitiendo aprovechar información o comportamientos ya existentes.

2.1.3. MECANISMOS DE SENSADO DE POSICIÓN

En las aplicaciones móviles que son de interés en este trabajo es fundamental la posición del usuario para poder brindarle la información contextual de donde se encuentra el mismo.

A continuación se describen algunos mecanismos que se pueden usar para determinar la posición del usuario.

- **Sistemas de Posicionamiento Global (A-GPS).** Este sistema permite determinar en todo el mundo la posición, en relación a la tierra, de un objeto, una persona o un vehículo, con una precisión de pocos centímetros (si se utiliza GPS diferencial), aunque lo habitual es una precisión de 5 a 10 metros. Es conveniente su uso para posicionamiento *outdoor* (ambientes abiertos), ya que en áreas *indoor* (ambientes cerrados como un edificio) se tiene problemas de señal o errores de posicionamiento. Este sistema hace triangulación de satélites para determinar la posición del usuario, lo mínimo necesario son tres satélites, pero cuantos más satélites estén involucrados mayor será la precisión de la posición. En la actualidad, la mayoría de los Smartphone posee dicha funcionalidad. No requiere la intervención del usuario para su funcionamiento.
- **Códigos de Barra 2D (tags o etiquetas).** Los códigos de barra 2D son un tipo particular de etiquetas que están representados en dos dimensiones. Permiten determinar la posición de un dispositivo móvil a través de la lectura e interpretación de un código. Para realizar dicha lectura e interpretación se debe contar con un programa especial, denominado generalmente, lector de código de barra. Este programa usa la cámara para capturar el código. El mismo puede ser utilizado tanto en ambientes *indoor* como *outdoor*, pero requiere de la intervención del usuario para su funcionamiento. Es decir, el usuario debe ejecutar el programa de lectura de código para poder recibir la información. El código por sí sólo no tiene un posicionamiento asignado, sino que es el desarrollador de la aplicación móvil el que da la interpretación de posición acorde a dónde se halla ubicado el mismo.
- **Wi-Fi.** Permite determinar la posición del dispositivo móvil a través del análisis de señales recibidas de los puntos de acceso (APs) o bien mediante WPS (software que contiene en sus bases de datos la posición de redes Wi-Fi). Sirve para ambientes *indoor* y *outdoor*, pero la interferencia como así también la precisión lo convierten en un método poco eficaz y no muy utilizado.
- **Antenas de un Proveedor de Telefonía Móvil (Cell ID and TimingAdvance|CGI-TA).** Este sistema permite obtener la posición del dispositivo móvil en relación a la tierra, basándose en redes de telefonía. Se usa para determinar la posición y el radio de alcance de la terminal a la cual está conectada el dispositivo. Para lograr más precisión se pueden usar varias terminales y triangular los radios de las mismas, pero todo depende de las distancias entre las distintas terminales.

2.1.4. SERVICIOS WEB

Existen múltiples definiciones sobre los servicios web, lo que muestra su complejidad a la hora de dar una adecuada definición que englobe todo lo que son e implican. La World Wide Web Consortium (W3C Consortium) lo define como *"…un sistema de software diseñado para soportar interacción interoperable máquina a máquina sobre una red. Este tiene una interfaz descrita en un formato procesable por una máquina (específicamente WSDL). Otros sistemas interactúan con los servicios web en una manera prescrita por su descripción usando mensajes SOAP, típicamente enviados usando HTTP con una serialización XML en relación con otros estándares relacionados con la Web"* (W3C Consortium, 2004). En este trabajo se usan los servicios web acorde a la definición de la W3C.

Los servicios web proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar. La figura 2.8 muestra un esquema general de servicio web, donde se tiene un proveedor del servicio (servidor web) por un lado y por otro, las aplicaciones clientes que consumen ese servicio. Las aplicaciones que consumen esos servicios, podría ser aplicaciones desktop o móvil, como así también aplicaciones ejecutándose en un servidor como se puede visualizar en la figura.

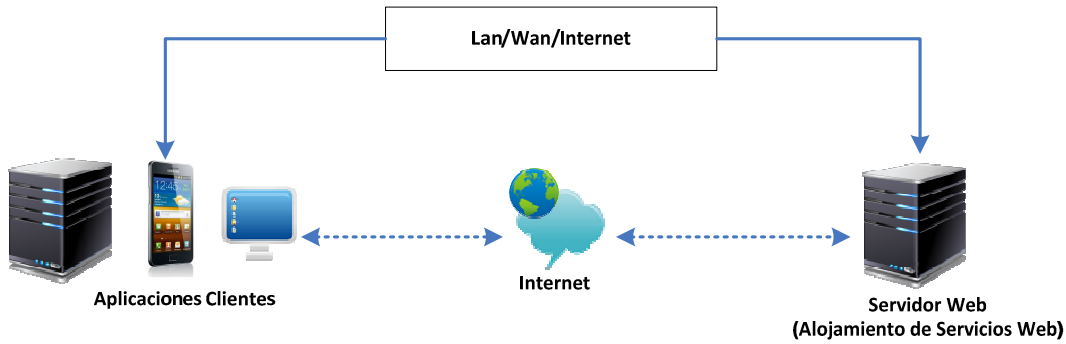


Figura 2.8 –Esquema general de servicios web.

El formato estándar para intercambiar datos es el lenguaje XML (Extensible Markup Language). Mediante este lenguaje se representan los datos que provee el servicio web. Estos datos deben de ser enviados usando algún protocolo de comunicación (entre las aplicaciones y el servicio web).

A continuación se detallan dos protocolos de comunicación que pueden ser usados para implementar servicios web:

- SOAP (*Simple Object Access Protocol*). Es un protocolo de la capa de aplicación para el intercambio de mensajes basados en XML sobre redes de computadoras. Básicamente, es una vía de transmisión entre un SOAP Sender y un SOAP Receiver. Pero los mensajes SOAP deben

interactuar con un conjunto de aplicaciones para que se pueda generar un *dialogo* a través de mensajes SOAP. Un mensaje SOAP es la unidad fundamental de una comunicación entre nodos SOAP (W3C Consortium, 2007).

- RESTful (*Representational State Transfer*). Fielding (Fielding Roy, 2008) da la siguiente definición: *"estilo de arquitectura de software para sistemas hipermedias distribuidos tales como la World Wide Web".* En resumen, es un conjunto de principios para el diseño de redes, que es utilizado comúnmente para definir una interfaz de transmisión sobre HTTP de manera análoga a como lo hace SOAP. Aunque REST como tal no es un estándar, posee un conjunto de estándares tales como HTML, URL, XML, GIF, JPG y tipos MIME. Los principios de REST son:
    - Escalabilidad de la interoperabilidad con los componentes
    - Generalidad de interfaces
    - Puesta en funcionamiento independiente
    - Compatibilidad con componentes intermedios

Otros estándares relacionados con servicios web son listados a continuación.

- Web Services Protocol Stack: así se denomina al conjunto de servicios y protocolos de los servicios web.
- XML (*Extensible Markup Language*): es el formato estándar para los datos que se vayan a intercambiar.
- WSDL (*Web Services Description Language*): es el lenguaje de la interfaz pública para los servicios web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios web.
- UDDI (*Universal Description, Discovery and Integration*): protocolo para publicar la información de los servicios web; permite comprobar qué servicios web están disponibles.
- WS-Security (*Web Service Security*): Protocolo de seguridad aceptado como estándar por OASIS (*Organization for the Advancement of Structured Information Standards*); garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados.

Es muy importante la arquitectura de la aplicación al momento de utilizar servicios web para la comunicación entre aplicaciones (cliente-servidor). Es por ello que a continuación se hace referencia a un concepto que está tomando fuerza en los nuevos desarrollos de aplicaciones: Arquitectura Orientada a Servicios (SOA). No es un fabricante, ni un modo de programación. Simplemente es una arquitectura para conectar sistemas entre sí permitiendo aplicar lógicas de control, negocio o procesos (Alba, 2008). Se denomina SOA a un marco conceptual de arquitecturas informáticas de negocios que se caracteriza por ofrecer las funcionalidades básicas de los Sistemas de Información de una empresa a través de servicios reutilizables (Mezo, Chaparro, & Heras, 2008).

Desde el punto de vista de negocio, se define SOA como un conjunto de componentes informáticos que se integran de forma flexible para configurar distintos procesos de negocio; desde una perspectiva técnica, estas arquitecturas constan de servicios que se pueden invocar para realizar operaciones (Mezo, Chaparro, & Heras, 2008).

El principal objetivo de las SOAs es construir los distintos sistemas de información de una empresa sobre un conjunto de estándares informáticos con el objetivo de que todos ellos, incluso los realizados con distintas tecnologías, puedan operar de forma integrada y sin que existan dependencias entre los mismos. Gracias a esta configuración, SOA se considera una tecnología especialmente adecuada para obtener la integración de las distintas actividades de negocio con un costo viable (Mezo, Chaparro, & Heras, 2008).

Dentro del marco conceptual de SOA, encajan distintas tecnologías y protocolos, como los servicios web. Estas tecnologías son las más empleadas en la actualidad y son la puerta de entrada de SOA en empresas que no están preparadas para asumir el cambio global que supone la completa implantación de este paradigma informático (Mezo, Chaparro, & Heras, 2008).

SOA presenta las siguientes ventajas:

- Escalabilidad
- Robustez
- Homogeneidad
- Facilidad en la adaptación de nuevos servicios
- Facilidad en la restructuración de sistemas
- Aplicar lógica en el middleware pudiendo implementar procesos de negocio
- Recoger información y procesarla para obtener resultados más útiles
- Ahorro de tiempos de implantación
- Ahorro en tiempos de mantenimiento y operación
- Independencia de la plataforma y del lenguaje de programación, homogéneo para implementaciones (utilizando servicios web).

2.1.5. BUENAS PRÁCTICAS DE DISEÑO: MODELO VISTA CONTROLADOR

Durante toda la década del setenta, SmallTalk y algunos otros lenguajes como Simula I, fueron construyendo gradualmente el paradigma de programación orientada a objetos y estableciendo conceptos tales como objetos, clases, encapsulamiento, herencia y polimorfismo (Burbeck, 1997). Si bien dichos lenguajes no son usados actualmente para implementar aplicaciones comerciales, los conceptos que dejaron en el mundo del desarrollo de software están vigentes en la actualidad y son la base de lenguajes modernos como C++, Java o C#.

SmallTalk también fue el primer lenguaje de programación que permitió diseñar interfaces de usuario con múltiples *ventanas* desplegadas en una misma pantalla, concepto que después fue aplicado por GEMS, Macintosh, X11, Windows y otras modernas interfaces gráficas de usuario. El concepto central detrás de las librerías de interfaz de usuario provistas por SmallTalk está basado en el patrón de diseño Modelo Vista Controlador (MVC), creado en el año 1979 por el profesor Trygve Reenskaug (Deacon, 2013).

MVC es un patrón de diseño que considera dividir una aplicación en tres módulos claramente identificables y con funcionalidad bien definida: el modelo, las vistas y el controlador.

Modelo (Model): Esta es la representación específica de la información con la cual el sistema opera y se compone por el Sistema de Gestión de Base de Datos y la lógica de negocio. La lógica de negocio asegura la integridad de estos y permite derivar nuevos datos. El Sistema de Gestión de Base de Datos (SGBD) será el encargado de almacenar los cambios en los datos (agregar datos, editarlos o borrarlos) producidos por la lógica de negocio; ejemplos de SGBD son MySQL, Oracle y MSSQL. Es recomendable una capa de abstracción extra denominada Data Access Object (DAO), que es un componente de software que suministra una interfaz común entre la lógica de negocio y el SGBD. La lógica de negocios podría estar implementada usando, por ejemplo, un enfoque orientado a objetos.

Vista (View): Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario. Por lo tanto, la vista es la encargada de presentar los datos al usuario y la interfaz necesaria para modificarlos. Un ejemplo de tecnología podría ser las JSP (*Java Server Page*) que, mediante el servidor, genera HTML que interpreta el navegador del usuario mostrándole los datos y los formularios que constituyen la vista para que pueda interactuar con la aplicación.

Controlador (Controller): Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista. Por lo general, el controlador sería la unidad central que comunica la vista con el modelo y viceversa, asociando los eventos del usuario con los cambios que se producirán en el modelo y devolviendo los datos resultantes que genere el modelo a la vista que corresponda.

En la figura 2.9 se puede apreciar el esquema que tiene el patrón MVC, donde se destaca la interacción que existe entre cada uno de los tres componentes.

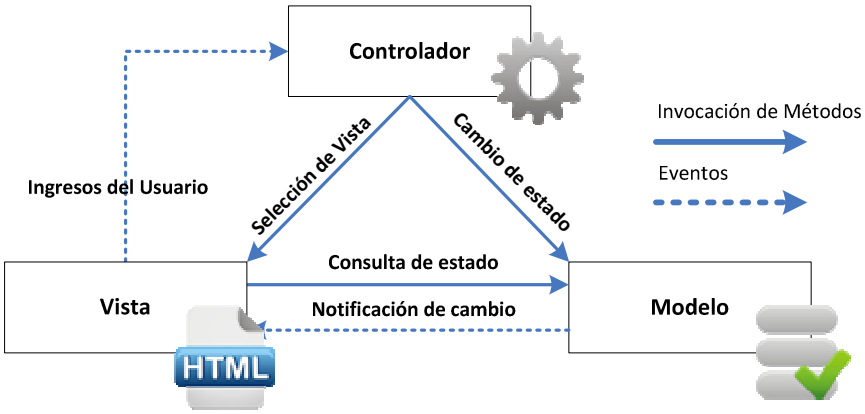


Figura 2.9 –Esquema patrón Modelo-Vista-Controlador (MVC).

El usuario siempre ingresa los datos desde la vista, el control recibe esta nueva información y reacciona, por ejemplo cambiándole el estado al modelo, y devolviéndole una nueva vista al usuario. En el esquema presentado en la figura 2.9, la vista tiene acceso a los componentes del modelo. También se puede usar este patrón restringiendo a que la vista sólo interactué con el controlador, y el modelo sólo es accedido desde el controlador.

El Patrón MVC se usa frecuentemente en aplicaciones Web, donde la vista está constituida, por ejemplo, por las páginas HTML. Algunos frameworks Web que usan el patrón MVC son:

- ASP.NET MVC (.NET)
- Symfony (PHP)
- Ruby onRails (Ruby)
- Struts (Java)

2.1.5.1. ASP.NET MVC

El ASP.NET MVC Framework es un framework de aplicaciones Web que implementa el patrón MVC. Provee una alternativa a ASP.NET Web. Este framework MVC está definido en el espacio de nombre *System.Web.Mvc* y es una parte fundamental de soporte del espacio de nombres *System.Web.*

En (Micorosoft ASP.NET Team, 2007) se mencionan las siguientes características de ASP.NET MVC:

- Separación de tareas de aplicación (lógica de entrada, lógica comercial y lógica de la interfaz de usuario), facilidad para pruebas y desarrollo basado en pruebas (TDD);
- Un marco extensible y conectable. Los componentes del marco de ASP.NET MVC están diseñados para que se puedan reemplazar o personalizar con facilidad;

- Amplia compatibilidad para el enrutamiento de ASP.NET, un eficaz componente de asignación de direcciones URL que le permite compilar aplicaciones que tienen direcciones URL comprensibles y que admiten búsquedas;

Además, se menciona en dicho documento las ventajas de usar MVC como parte de ASP.NET:

- Divide la complejidad de las aplicaciones en tres componentes bien diferenciados;
- Los desarrolladores tienen control absoluto de los estados de la aplicación, ya que el modelo sólo puede ser modificado a través de la interacción con la vista;
- Usa la idea de un controlador frontal permitiendo recibir requerimientos complejos;
- Es compatible con el desarrollo de pruebas basadas en test;
- Produce una interfaz limpia sin compartimiento propio de la lógica de negocios.

2.1.6. CALIDAD Y EFICIENCIA EN APLICACIONES MÓVILES

La calidad es una característica del software o de los sistemas que no es factible de ser medida. La calidad se evalúa o se calcula, a partir de la medición de ciertos atributos relacionados con ella. Un atributo es una propiedad física o abstracta que se puede medir usando una métrica. Esos atributos definen el valor de uno o varios conceptos calculables que permiten evaluar la calidad.

La calidad puede ser vista desde diferentes puntos de vista: desde el punto de vista del desarrollador, del usuario, etc. Estos puntos de vista están relacionados con los estudios de calidad interna, externa o en uso. Es muy importante considerar que siempre se desea comprender y evaluar la calidad de un producto software para poder tomar decisiones respecto al mismo. Primero se debe medir atributos, luego evaluar la calidad, así se logra comprender la calidad para poder hacer recomendaciones sobre un producto software. A raíz de las recomendaciones se producen cambios y, por consiguientes, mejoras en el sistema.

Si bien existe una gran cantidad de estudios sobre la calidad de las aplicaciones en general, no ocurre lo mismo con la calidad de las aplicaciones móviles. Es por ello que este trabajo pretende contribuir al análisis de la calidad de las aplicaciones móviles, abordando una de las principales características de la calidad: la eficiencia.

Se toma como referencia de calidad la Norma ISO 25000 (International Standar Organization, 2013) y el trabajo académico y de investigación desarrollado por el grupo de investigación GIDIS WEB (Becker, Lew, & Olsina, 2012; Olsina, Lew, Dieser, & Rivera, 2012; Olsina, Lew, Dieser, & Rivera, 2011; Olsina, Pappa, & Molina, 2008; Olsina & Rossi, 2002; Lew, Olsina, Becker, & Zhang, 2011) en Argentina.

A continuación se presentan los principales aspectos de la calidad y luego se particulariza la característica eficiencia en aplicaciones móviles.

2.1.6.1. ISO 25000

Las ISO 25000 (International Standar Organization, 2013) constituyen una familia de normas que sustituyen y amplían las normas ISO 9126 (Calidad de un Producto Software) e ISO 14598 (Evaluación del Software) desarrollada por el subcomité SC 7 (Ingeniería de software y sistemas) del Comité Técnico Conjunto ISO/IEC JTC 1. Es conocida con el nombre de SQuaRE (Requisitos y Evaluación de Calidad de Productos de Software) y la misma se organiza en cinco apartados detallados en la figura 2.10. Esos capítulos son:

Figura 2.10 – Organización de la familia de normas ISO 25000 [ISO, 25000].

- **ISO/IEC 2500n. División de gestión de calidad**. Los estándares y normas que forman esta división definen todos los modelos comunes, términos y referencias a los que se alude en las demás divisiones de SQuaRE.
- **ISO/IEC 2501n. División del modelo de calidad.** La norma que conforma esta división presenta un modelo de calidad detallado, incluyendo características para la calidad del producto y en uso.
- **ISO/IEC 2502n. División de mediciones de calidad**. Las normas pertenecientes a esta división incluyen un modelo de referencia de medición de la calidad del producto software, definiciones matemáticas de las métricas de calidad y una guía práctica para su aplicación. Presenta aplicaciones de métricas para la calidad del producto y en uso.
- **ISO/IEC 2503n. División de requisitos de calidad.** Los estándares que forman parte de esta división ayudan a especificar los requisitos de calidad. Estos requisitos pueden ser usados en el proceso de especificación de requisitos de calidad para un producto software que va a ser desarrollado ó como entrada para un proceso de evaluación. El proceso de definición de requisitos se guía por el establecido en la norma ISO/IEC 15288 (ISO, 2003).
- **ISO/IEC 2504n. División de evaluación de la calidad**. Estos estándares proporcionan requisitos, recomendaciones y guías para la evaluación de un producto software, tanto si la llevan a cabo evaluadores, como clientes o desarrolladores.
- **ISO/IEC 25050–25099. Estándares de extensión SQuaRE**. Incluyen requisitos para la calidad de productos de software "Off-The-Self" y para el formato común de la industria (CIF) para informes de usabilidad.

Al igual que la norma ISO/IEC 9126, este estándar define tres vistas diferenciadas en el estudio de la calidad de un producto detalladas en la tabla 2.1:

Tabla 2.1 - Aspectos o vistas diferenciadas para el estudio de la calidad de un producto.

	Vista Interna	**Vista Externa**	**Vista en Uso**
Definición	Se ocupa de las propiedades del software como: el tamaño, la complejidad o la conformidad con las normas de orientación a objetos. Es medible a partir de las características intrínsecas como lo es el código fuente	Analiza el comportamiento del software en producción y estudia sus atributos, por ejemplo: Una prueba del software o el rendimiento de un software en una máquina determinada, el uso de memoria de un programa o el tiempo de funcionamiento entre fallos.	Mide la productividad y efectividad del usuario final al utilizar el software en un contexto determinado.
Características	Puede utilizarse desde las primeras fases del desarrollo, permitiendo detectar deficiencias en el software en edades muy tempranas del ciclo de vida del software	Necesita que el producto software este completo y se utilizará por tanto en el pase a producción del producto, siendo muy dependiente de la máquina donde se ejecute	Estudia el producto software finalizado. Es dependiente del usuario y estará condicionada a los factores personales del mismo.
Modelo	**Calidad del Producto**		**Calidad en Uso**

En concordancia con la filosofía de los modelos clásicos, el modelo de referencia para la medición de la calidad del producto software de la norma ISO/IEC 25000 establece que la calidad del producto software está determinada por *características* que, a su vez, están determinadas por *subcaracterísticas*. Estas subcaracterísticas pueden ser valoradas a través de ciertos atributos o aspectos que son medidos con métricas y herramientas.

La norma ISO/IEC 25010 contempla dos modelos de calidad:

- Modelo de Calidad de Producto (Vista Interna/Externa): categoriza las propiedades de la calidad del producto o sistema en ocho características: *funcionalidad, eficiencia, compatibilidad, usabilidad, confiabilidad, seguridad, capacidad de mantenimiento y portabilidad*. Cada característica está compuesta de un conjunto de subcaracterísticas. El modelo de calidad de producto puede ser aplicado a un producto software o a un sistema computacional que incluye software, dado que la mayoría de las subcaracterísticas son relevantes tanto para el software como para los sistemas.
- Modelo de Calidad en Uso (Vista en Uso): define cinco características referidas a las salidas de la interacción con el sistema: *efectividad, eficiencia, satisfacción, libre de riesgos, contexto*. Cada característica puede ser asignada a diferentes actividades de los stakeholders, por ejemplo, la interacción de un operador o el mantenimiento de un desarrollador. La calidad en uso de un sistema caracteriza el impacto que un producto (sistema o software) tiene en los stakeholders. Está determinado por la calidad del ambiente del software, hardware y operativo, y por el ambiente de los usuarios, tareas y grupo social. Todos estos factores contribuyen a la calidad en uso del sistema.

Por otra parte, la norma ISO/IEC 25012 contempla el Modelo de Calidad de Datos, que no se considera en el presente trabajo.

La calidad del producto y la calidad en uso están muy relacionadas, es por ello que muchos factores que inciden en una de ellas, repercuten en la otra. Por ejemplo: un software con alta funcionalidad y complejidad, ejecutado sobre una máquina con bajas prestaciones, provocará una baja eficiencia en el usuario final, independientemente del factor humano. Esto es muy común en entornos móviles donde los recursos principales del dispositivo son escasos.

En las tablas 2.2 y 2.3 se presentan las características y subcaracterísticas de los modelos de calidad del producto y de calidad en uso.

Tabla 2.2 - Modelo de Calidad en Uso, características y subcaracterísticas de la calidad.

Modelo	Características	Sub-Características
Calidad en Uso	Efectividad	• Efectividad
	Eficiencia	• Eficiencia
	Satisfacción	• Utilidad • Confianza • Placer • Confort
	Libre de Riesgo	• Mitigación del riesgo económico • Mitigación del riesgo en cuanto seguridad y salud • Mitigación de riesgos ambientales
	Contexto	• Completitud contextual • Flexibilidad

Tabla 2.3 - Modelo Calidad del Producto, características y subcaracterísticas de la calidad.

Modelo	Características	Sub-Características
Calidad del Producto	Funcionalidad	• Completitud • Correctitud • Oportunidad
	Fiabilidad	• Madurez • Tolerancia a fallos • Disponibilidad • Recuperabilidad (Capacidad de Recuperación)
	Usabilidad	• Capacidad de ser reconocido apropiadamente • Facilidad de aprendizaje • Operabilidad • Interfaz de usuario (estética) • Protección contra errores del usuario • Accesibilidad
	Eficiencia	• Tiempo de respuesta (Comportamiento en el tiempo) • Utilización de los Recursos • Capacidad

Tabla 2.3 - Modelo Calidad del Producto, características y subcaracterísticas de la calidad (continuación).

	Mantenibilidad	• Modularidad • Reusabilidad • Capacidad de análisis • Capacidad de ser modificado • Capacidad de ser probado
	Portabilidad	• Adaptabilidad • Facilidad de instalación • Facilidad de ser reemplazado
	Compatibilidad	• Coexistencia • Interoperabilidad
	Seguridad	• Confidencialidad • Integridad • No-Repudio • Responsabilidad • Autenticidad

En este proyecto se trabajará en el marco del Modelo de Calidad del Producto. Desde este punto de vista, se define a la eficiencia *como la capacidad del producto software para proporcionar prestaciones apropiadas, en relación a la cantidad de recursos usados, bajo condiciones determinadas* (Piattini, Garcia, & Caballero, 2007).

Como fue explicado anteriormente, la Calidad del Producto influye en la Calidad en Uso; y la eficiencia no está exenta de esto. Es por ello que se considera que poniendo énfasis en la etapa de desarrollo del producto software, se podrá garantizar un buen desempeño del mismo y así mejorar su Calidad en Uso.

Los recursos considerados para definir la eficiencia del producto pueden ser: otros productos software, la configuración del software y hardware del sistema, impresoras, cámaras, GPS, Wi-Fi, o unidades de almacenamiento.

Revisando el estándar, la eficiencia se determina en función de tres subcaracterísticas:

- **Comportamiento respecto al tiempo:** Grado en el cual un sistema en ejecución cumple sus requisitos de tiempos de procesamiento, de respuesta y tasas de througput (rendimiento, caudal de datos).

- **Utilización de los Recursos:** Grado en el cual la cantidad y tipos de recursos usados por un sistema en ejecución cumplen sus requerimientos. Se incluyen los recursos humanos.
- **Capacidad:** Grado en el cual los límites máximos de los parámetros del producto software o sistema cumplen con los requerimientos. Los parámetros pueden incluir el número de artículos que pueden ser almacenados, el número de usuarios simultáneos, el ancho de banda de la comunicación, el rendimiento de las transacciones y el tamaño de la base de datos.

2.1.6.2. MEDICIÓN DE LAS CARACTERÍSTICAS DE LA CALIDAD

Las medidas de calidad del software indican las características y subcaracterísticas de calidad del producto software. El valor de estas medidas de calidad del software se obtiene por la aplicación de una función de medición a los elementos de medición de la calidad, ver figura 2.11. Los elementos de medición de la calidad son atributos o medidas base o medidas derivadas obtenidas según describe el método de medición correspondiente, de acuerdo a la ISO/IEC 15939. Es importante resaltar que las normas ISO/IEC 9126 y 25000 no indican qué medidas de calidad se deben usar para una determinada característica o subcaracterística.

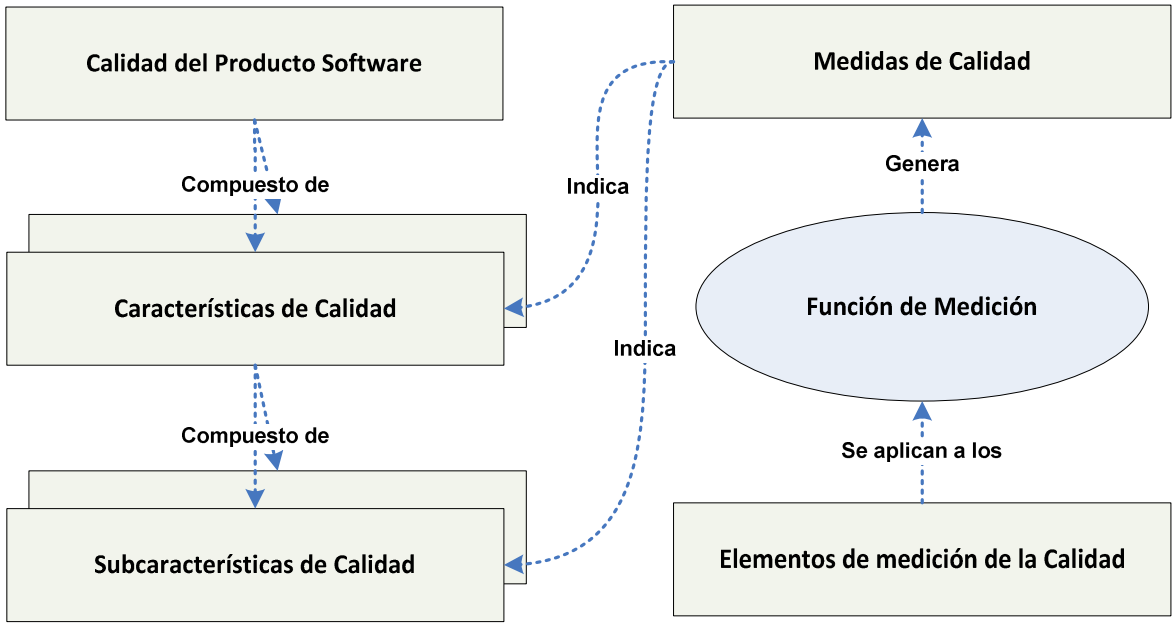


Figura 2.11 - Modelo de Referencia de Medición de la Calidad del Producto Software, según la ISO/IEC 25000.

2.1.6.3. OTROS APORTES IMPORTANTES SOBRE CALIDAD

En Argentina, el grupo de investigación GIDIS WEB[2], dirigido por el Dr. Luis Olsina, ha venido trabajando en métodos y procesos de medición y evaluación de la calidad de los sistemas desde hace varios años atrás. En los últimos tiempos se inició el estudio de la calidad de las nuevas generaciones de aplicaciones Web y de aplicaciones móviles.

Entre los resultados más difundidos de este grupo se encuentran:

- Estrategia integrada de medición y evaluación de la calidad, *GOCAME: Goal-Oriented Context-Aware Measurement and Evaluation* (Lew, Olsina, Becker, & Zhang, 2011). Basada en tres pilares: un marco conceptual de medición y evaluación, un proceso de medición y evaluación y un método.
- Marco y Modelos de Calidad *2Q2U: Quality, Quality in use, actual Usability and User experience* (Olsina, Lew, Dieser, & Rivera, 2012);
- Estrategia para la comprensión y mejoramiento de la calidad en uso *SIQinU: Strategy for understanding and Improving Quality in Use* (Becker, Lew, & Olsina, 2012);
- Marco de modelación y estrategia de evaluación de la calidad para la comprensión y optimización de aplicaciones Web (Olsina, Lew, Dieser, & Rivera, 2011);
- Marco para la medición y evaluación de la calidad basado en una ontología de métricas e indicadores *INCAMI: Information Need, Concept model, Attribute, Metric and Indicator* (Olsina, Pappa, & Molina, 2008);
- Estrategia cuantitativa para la evaluación de la calidad de un sitio y aplicación Web, *WebQEM* (Olsina & Rossi, 2002).

Uno de los primeros pasos para realizar la evaluación consiste en definir los requisitos no funcionales, lo cual se realiza generalmente a través de modelos de calidad y también a través de un marco de calidad que define relaciones entre los modelos. Los modelos de calidad pueden estar enfocados a diferentes categorías de entidades: producto, sistema, sistema en uso. Los productos son entidades que se encuentran en las primeras etapas del ciclo de vida del software; los sistemas de información son productos ejecutables en un contexto específico, que pueden incluir tanto hardware como software, y el sistema de información en uso son los sistemas anteriores pero operados por usuarios reales en contextos reales de uso. Estos modelos se relacionan directamente con la propuesta de los estándares de la tabla 2.1. En la figura 2.12 se presentan estos modelos propuestos y sus ejemplos.

[2] Grupo de Investigación de Ingeniería de Software y Web, Departamento de Informática, Facultad de Ingeniería, Universidad Nacional de La Pampa.

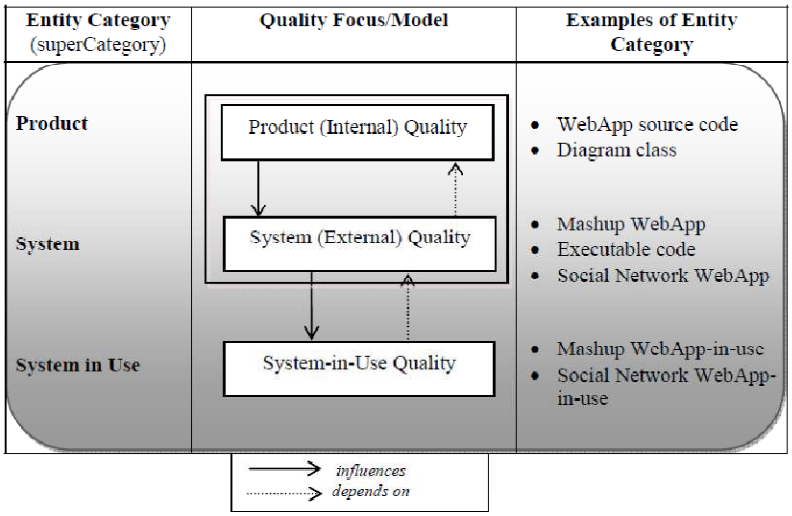


Figura 2.12 – Marco de modelación de calidad que considera la entidad objetivo, el modelo de calidad y las relaciones entre estos (Olsina, Lew, Dieser, & Rivera, 2012).

La **estrategia GOCAME** propone un método integral que involucra tres pilares importantes para la medición y evaluación de la calidad de un software: una base conceptual, un proceso y un método. En cuanto a la base conceptual, se propone una ontología que incorpora conceptos y sus relaciones referidos a: requisitos no funcionales (a los cuales hace referencia la calidad), contexto, medición, evaluación, proyecto. El proceso hace referencia al *qué* se debe medir y el método hace referencia a *cómo* medir.

El **marco C-INCAMI** define la base conceptual para el dominio de medición y evaluación (M&E). Diseñado para satisfacer una necesidad de información específica en un contexto dado definiendo todos los conceptos y relaciones que se usan en todas las actividades de M&E. El marco tiene seis componentes: definición del proyecto, especificación de requisitos no funcionales, especificación de contexto, diseño e implementación de la medición, diseño e implementación de la evaluación, especificaciones de análisis y recomendación.

En la figura 2.13 se presentan los modelos de calidad de los requisitos no funcionales y de contexto, el resto se puede ver en (Olsina, Pappa, & Molina, 2008). Los componentes NFRS especifican la *necesidad de información* de cualquier proyecto M&E; es decir, el *propósito* (comprender, predecir, mejorar, etc.) y el *punto de vista del usuario* (desarrollador, usuario, etc.). Se *enfoca* en un *concepto calculable* y *especifica* la *categoría de entidad* a evaluar (por ejemplo, un sistema, un sistema en uso), por medio de una *entidad* concreta (por ejemplo, cúspide.com). Por otro lado, el concepto calculable seleccionado y sus *subconceptos* pueden ser *representados* por un *modelo de concepto* donde las hojas de un modelo instanciado son *atributos asociados con* una categoría de entidad. De hecho, una entidad concreta *pertenece* a una categoría de entidad.

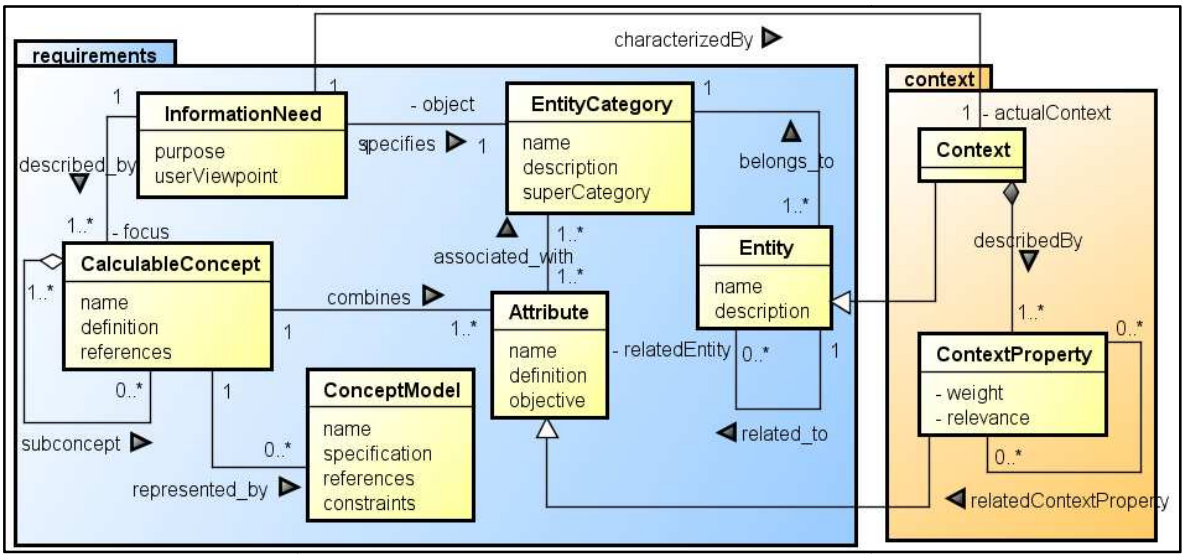


Figura 2.13 – Componentes Requisitos no Funcionales y de Contexto del C-INCAMI (Olsina, Lew, Dieser, & Rivera, 2012).

En cuanto al **proceso de M&E**, el modelado puede hacerse desde diferentes vistas:

- Vista Informacional, se enfoca en los artefactos que se obtienen en el proceso;
- Vista funcional, se enfoca en las actividades que se realizan durante el proceso;
- Vista dinámica o de comportamiento, considera el orden de las actividades (secuencialidad, paralelismo, iteraciones);
- Vista organizacional, se enfoca en los roles de los agentes y las comunicaciones entre ellos;
- Vista metodológica, se concentra en las métricas que se utilizan.

Teniendo en cuenta el marco conceptual presentado y el estándar ISO sobre los procesos de medición y evaluación, **el proceso de la estrategia GOCAME** propone las siguientes actividades fundamentales (ver figura 2.14):

1. Definición de los requisitos no funcionales o NFR
2. Diseño de las mediciones
3. Diseño de la evaluación
4. Implementación de las mediciones
5. Implementación de la evaluación
6. Análisis y recomendación

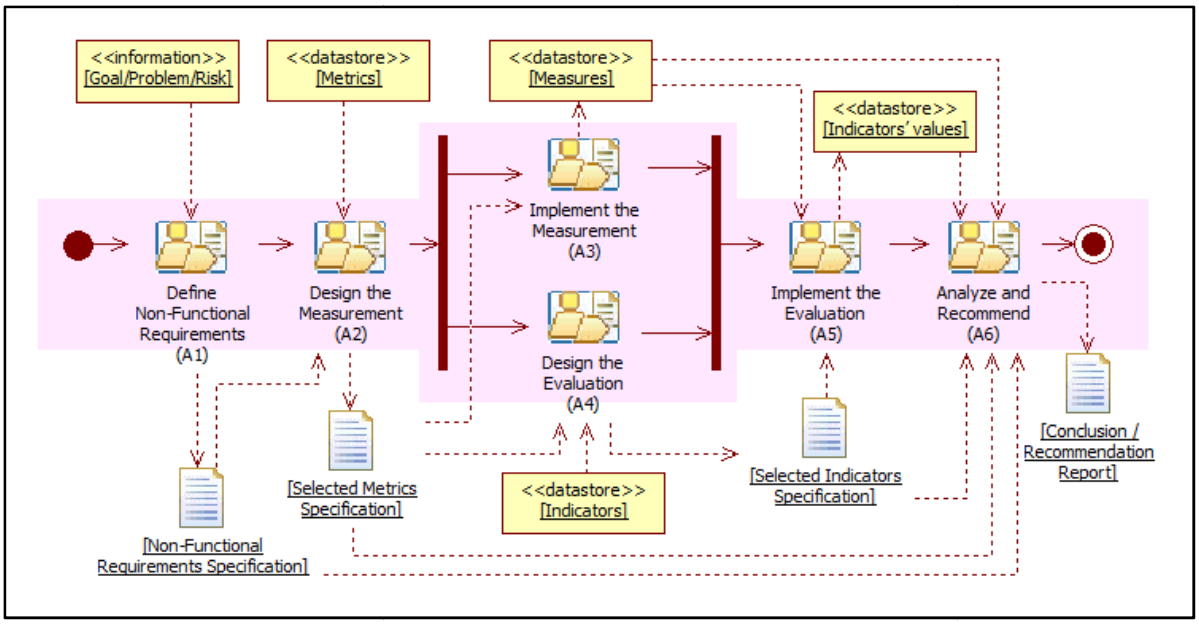


Figura 2.14 – Vista global del proceso de medición y evaluación de GOCAME (Lew, Olsina, Becker, & Zhang, 2011).

El proceso sigue un enfoque orientado a objetivos. La actividad 1, definición de requisitos no funcionales, tiene un objetivo principal como entrada y un documento de especificación no funcional como salida (ver figura 2.14). Luego, la actividad de diseño de mediciones permite identificar las métricas que se usarán para cuantificar los atributos -las métricas se buscan y seleccionan de un repositorio o catálogo de métricas-; la salida es un documento de especificación de métricas. Una vez que esto está realizado, se pueden llevar a cabo las actividades de diseño de la evaluación e implementación de las mediciones. La actividad de diseño de la evaluación permite identificar los indicadores elementales y globales que se van a usar. La actividad de implementación de mediciones usa las métricas especificadas para obtener las mediciones, las cuales se guardan en un repositorio de mediciones. Luego, se puede realizar la actividad de implementación de la evaluación. Finalmente, la actividad de análisis y recomendación tiene como entrada los valores de medición e indicadores, el documento de especificación de requerimientos y las especificaciones de las métricas e indicadores.

La figura 2.14 muestra en un diagrama SPEM sencillo las actividades, orden y artefactos que involucran la medición y evaluación de la calidad. Es decir, este diagrama presenta las vistas funcional y de comportamiento del proceso.

En cuanto a los **métodos y herramientas**, la estrategia GOCAME propone la metodología Web-QEM, Web Quality Evaluation Method (Olsina & Rossi, 2002). Mientras las actividades determinan "qué" es lo que se debe hacer, los métodos describen "cómo" realizar esas actividades, las cuales pueden estar automatizadas mediante herramientas. Una metodología es un conjunto de métodos relacionados.

El grupo GIDIS WEB propone una metodología compuesta por un conjunto de métodos bien definidos y cooperativos, modelos, técnicas y herramientas que, aplicadas consistentemente a las actividades del

proceso, producen las diferentes salidas esperadas. Particularmente, la metodología Web-QEM y la herramienta asociada C-INCAMI_Tool fueron instanciadas del marco conceptual y del proceso.

2.1.6.4. EFICIENCIA EN APLICACIONES MÓVILES

Como está visto, la eficiencia en aplicaciones en general está ligada directamente a la cantidad y forma en que la aplicación consume o utiliza los recursos para poder cumplir con exactitud los requerimientos impuestos.

En la actualidad, las aplicaciones convencionales se ejecutan en un entorno controlado con equipos de computación con grandes prestaciones y bajo una red fija que transfieren datos a gran velocidad.

Sin embargo, en un entorno móvil, donde el usuario necesita estar siempre "conectado", no ocurre lo mismo. Este entorno impone limitaciones de recursos a las aplicaciones móviles. Por ejemplo: El consumo de energía de los recursos principales tiene un impacto directo sobre la duración de la batería del dispositivo. La frecuencia de la comunicación del dispositivo móvil con un servidor de datos debe encontrar un equilibrio entre la obtención de datos y una buena experiencia de usuario sin que se agote la batería. Un uso desmedido del ancho de banda provoca retrasos que afectan el normal funcionamiento de la aplicación y la experiencia del usuario.

Tal es la importancia de esta situación, que los sistemas operativos Android y Windows Phone prevén la suspensión repentina de las aplicaciones en el ciclo de vida de las mismas para una mejor redistribución de los recursos.

Los recursos críticos que hacen uso la mayoría de las aplicaciones móviles y deben contemplarse son: CPU, pantalla, memoria principal, WI-FI, 3G, GPS y Cámara.

Considerando lo expuesto en el apartado 2.1.2.1, el diseño de la arquitectura de una aplicación móvil influye en la utilización de los recursos del dispositivo. Por lo tanto, para optimizar la eficiencia de una aplicación móvil, es importante definir con cautela qué arquitectura se adapta al propósito del aplicativo móvil a desarrollar utilizando de manera eficaz los recursos del dispositivo donde se ejecutará con un tiempo de respuesta adecuado para el usuario final.

2.2. MARCO TECNOLÓGICO

El lector encontrará en este apartado el resultado del estudio sobre las tecnologías que soportan el desarrollo e implementación de aplicaciones móviles. Estas tecnologías están relacionadas con lo abordado en el Marco Teórico.

Entre las tecnologías que se describen se encuentran: sistema operativo Android, aplicaciones Android, códigos QR y su lectura (se los usa para obtener información de contexto), librerías para el uso de servicios web en Android (KSOAP y RESTful).

Por otra parte, también se presentan en este apartado las tecnologías de comunicación móvil que se utilizan mundialmente, en el país y en la región donde se desarrolló y utilizó el prototipo de m-turismo que responde a cada arquitectura. Se incluye un relevamiento basado en entrevista a prestadores de servicio de telefonía celular en la zona.

Finalmente, se analizan las herramientas (software) que permiten la medición de los atributos de la eficiencia en aplicaciones móviles, describiendo PowerTutor y Traffic Monitor.

2.2.1. ANDROID

Cuando se enuncia el término Android, no se sabe si se está hablando de la plataforma o si el mismo es un lenguaje de programación. Es muy común decir “la aplicación está desarrollada en Android”. Para una mayor comprensión del significado de Android es muy importante conceptualizar la plataforma (el SO), las aplicaciones que se ejecutan sobre Android y el lenguaje de programación que se utiliza para el desarrollo de las mismas. Es por ello que, a continuación, se definen las características principales de cada uno de estos elementos.

2.2.1.1. LA PLATAFORMA O SISTEMA OPERATIVO

Android es un Sistema Operativo basado en GNU/Linux diseñado originalmente para dispositivos móviles, tales como teléfonos inteligentes, pero que posteriormente se expandió su desarrollo para soportar otros dispositivos como tabletas, reproductores MP3, netbooks, PCs, televisores, lectores de e-books. Es una pila de software para dispositivos móviles que incluye un sistema operativo, aplicaciones esenciales y middleware. El *kit de desarrollo de software* (SDK) de Android proporciona las herramientas y las *interfaces de programación de aplicaciones* (APIs) necesarias para empezar a desarrollar aplicaciones que se puedan ejecutar en dispositivos con la tecnología Android.

Las características más destacadas de este SO son:

- Soporte para tecnologías de conectividad como los son GSM/EDGE, UMTS, Bluetooth, entre otros;
- Navegador Web incluido en su plataforma;
- Base de Datos liviana propia, SQL Lite;
- El lenguaje de programación es Java, uno de lo más utilizados en la actualidad;
- Importante soporte de las empresas que lo patrocinan;
- Gran documentación de la plataforma en páginas oficiales y foros de consulta.

En cuanto a su arquitectura, en la figura 2.15 se puede apreciar que se divide en cuatro capas bien definidas. Por un lado la capa del *Kernel* que posee un Kernel Linux versión 2.6, similar al que se incluye en las distribuciones de Linux más conocidas pero adaptada para el hardware de los dispositivos móviles. Es la capa que contiene los drivers del hardware con el que interactúan las aplicaciones. No es necesario que la aplicación conozca la marca o el modelo del hardware a utilizar, solo requiere el uso del mismo y el sistema operativo lo abstraerá de las características particulares del hardware solicitado. Es la capa encargada de gestionar los periféricos y hardware del dispositivo.

La segunda capa, comenzando desde abajo, es la de *Biblioteca* o *Librerías*. Dichas librerías son nativas y desarrolladas en C o C++, en ella se encuentran las librerías del motor de base de datos SQLite y el Navegador Web WebKit, entre otras.

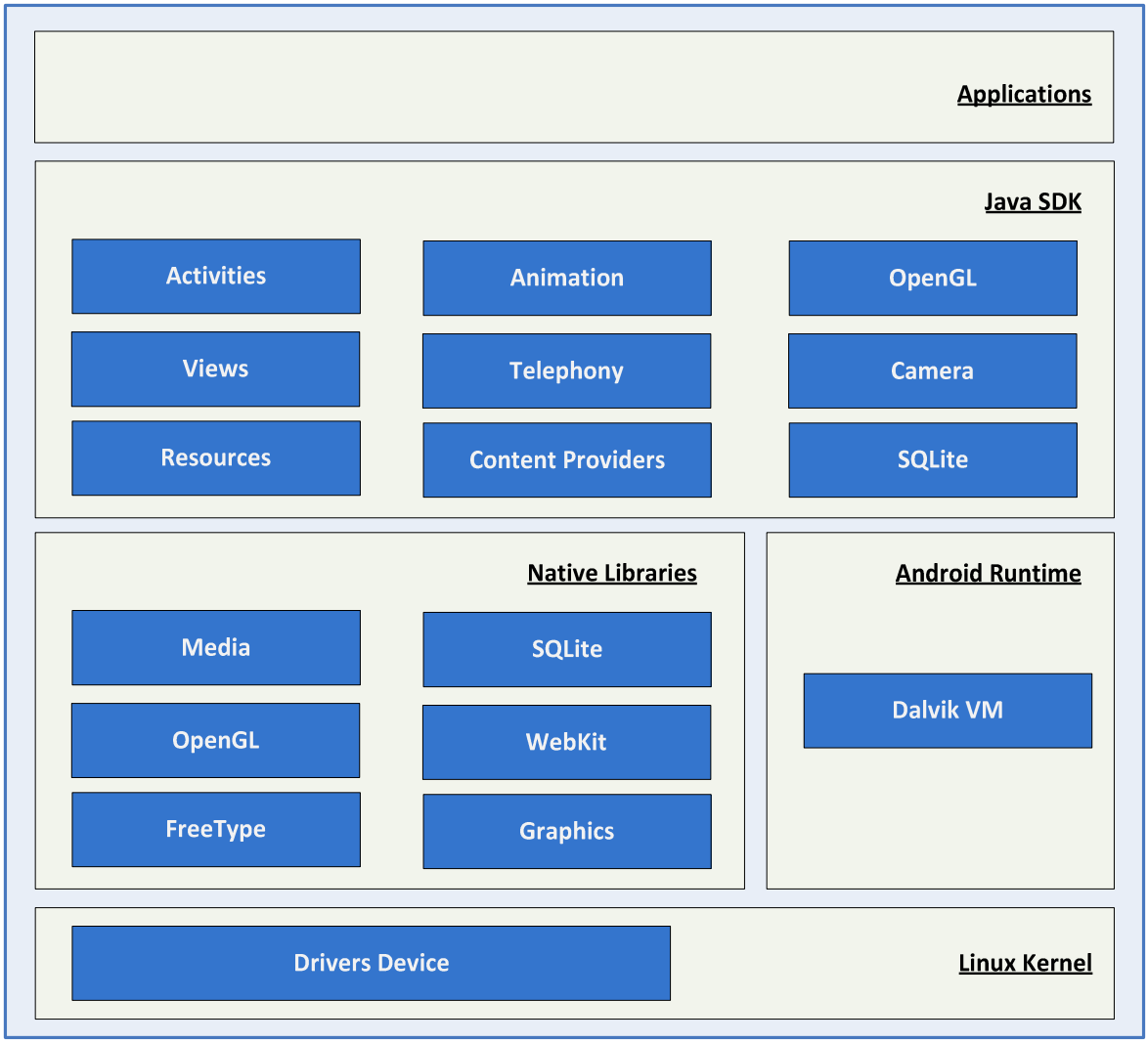


Figura 2.15 – Arquitectura detallada de la Plataforma Android (Komatineni & MacLean, 2012).

Dentro de esta capa, encontramos la sub-capa denominada *Entorno de Ejecución*. Está formada por librerías fundamentales de Android y por la máquina virtual Dalvik, que es el componente encargado de ejecutar todas las aplicaciones no nativas de Android. Estas normalmente están programadas en Java y se compilan en un formato específico "Intermedio" que es ejecutado por la máquina virtual Dalvik. Se lo denomina así ya que no se compila la aplicación para generar un ejecutable compatible con el hardware del dispositivo Android, si no es la máquina virtual la que genera dicha compilación y ejecución de la aplicación. Esto permite que se distribuyan las aplicaciones Android con la certeza que podrán ser ejecutadas en cualquier dispositivo Android que disponga de la versión mínima del SO que requiera cada aplicación. Una de las grandes discusiones que surgieron al conocerse el SO Android es si Dalvik es una máquina virtual Java. La respuesta es negativa, Dalvik no es compatible con el bytecode Java que son ejecutadas por las máquinas virtuales Java. Solo se usa a Java como lenguaje de programación. Cada aplicación Android se ejecuta en un hilo independiente, mejorando así la performance en dispositivos con pocos recursos y la gestión de ellas por parte del SO.

La siguiente capa es la denominada *Marco de la Aplicación o Java SDK*, la forman todas las clases y servicios que son utilizadas directamente por las aplicaciones para realizar sus funciones y que se apoyan en las bibliotecas y en el entorno de ejecución detallado.

Por último la capa llamada *Aplicaciones* está formada por las aplicaciones nativas (programadas en C o C++), administradas (programadas en Java) o las instaladas por el Usuario.

2.2.1.2. LAS APLICACIONES Y EL LENGUAJE DE PROGRAMACIÓN

Las aplicaciones Android están desarrolladas, en su mayoría, en el lenguaje Java utilizando SDKs nativos o extensiones C o C++. Entre ellas se encuentran: Android SDK, Google App Inventor, Mono para Android. Para el desarrollo de las aplicaciones se requiere conocimientos del lenguaje Java y disponer el SDK, que se puede descargar gratuitamente desde el sitio para Android en Google[3].

Algunas características de estas aplicaciones es que solo tienen un primer plano que ocupa toda la pantalla, están formadas por actividades donde, en un momento dado, una actividad pasa al primer plano y se coloca por encima de otra, formando una pila de actividades. El ciclo de vida de la aplicación es controlada por el SO, basándose en las necesidades del usuario, recursos disponibles, etc. En los dispositivos móviles, generalmente, los recursos son limitados, es por ello que el SO Android tiene una mayor incidencia en la gestión de los recursos que insumen las aplicaciones que otros SO, como los desktop. Como se dijo, esto contrasta con el ciclo de vida de aplicaciones desktop, Web o basadas en J2EE, estas últimas son libremente controladas por el contenedor que las ejecuta. Es por ello que las aplicaciones deben ser codificadas teniendo en cuenta que pueden culminar o pausarse en cualquier momento. Android ejecuta cada aplicación en un proceso separado, cada uno de los cuales cuenta con su propia máquina virtual Davlik. Esto proporciona un entorno protegido de memoria. Mediante el aislamiento de aplicaciones a un proceso individual, el SO puede controlar la aplicación que merece mayor prioridad. Este proceso es creado para la aplicación cuando se la inicia y seguirá corriendo mientras sea necesario y el sistema reclame recursos para otras aplicaciones y se los dé a ésta.

Los procesos en Android están definidos de acuerdo a su importancia, una clasificación de los mismos son (Komatineni & MacLean, 2012):

- Foregound process. Es la aplicación que contiene la actividad que ahora mismo se está mostrando en pantalla, se llama método onResume(). Habrá muy pocos procesos Foreground corriendo a la vez en el sistema y estos procesos solamente se destruirán si la memoria es tan baja que ni matando al resto de procesos se logran los recursos necesarios.

[3] Android Developer, http://developer.android.com/index.html.

- Visible process. Es el que contiene una actividad que es visible pero no en primera fila, se llama método onPause(). Estos procesos son considerados importantes por el SO y normalmente no se cierran.
- Service process. Es un servicio como los de cualquier Unix. Estos procesos hacen cosas en segundo plano que normalmente son importantes, el sistema nunca va a terminar un servicio a menos que sea necesario para mantener vivos todos los Visible y Foreground.
- Background process. Es un proceso que contiene una actividad que actualmente no es visible por el usuario, estos procesos no tienen demasiada importancia, puede ser un programa que arrancó hace tiempo y no se ha vuelto a usar, pasa a estar en background. Por eso es importante que cuando una aplicación pase a Background, se libere en la medida de lo posible todos los recursos posibles.
- Empty process. Es un proceso que ya no alberga nada, lo usa Android como cache para cuando se crea un proceso nuevo.

La arquitectura de las aplicaciones Android es *component-and-integration-oriented*. Esto permite una rica experiencia del usuario sin fisuras, con reutilización y fácil integración de aplicaciones. Como se dijo en el párrafo anterior, el ciclo de vida de una aplicación, no depende principalmente de ella, sino que está estrictamente controlada por el SO. Es por ello que es importante para el desarrollo de una aplicación Android consistente, robusta y estable, tener en cuenta el ciclo de vida de la misma cómo así también los eventos que se desencadenan en éste. Los eventos que gobiernan el ciclo de vida de una actividad de una aplicación Android se detallan a continuación y la transición de estos puede ser apreciada en la figura 2.16:

- onCreate() Llamado cuando la actividad es llamada por primera vez. Es donde se debe crear la inicialización normal de la aplicación, crear vistas, hacer los enlaces de los datos, etc. Este método da acceso al estado de la aplicación cuando se cerró. Después de esta llamada, siempre se llama al onStart().
- onStart() Llamada cuando la actividad no está visible al usuario, sino que está por serlo. Después de ésta, se puede ir al onResume() si la actividad va a ser visible o se puede ir al onStop() si se esconde.
- onRestart() Llamada cuando una actividad ha sido detenida, antes de volver a ser comenzada. Siempre viene después de un onStart().
- onResume() Llamada cuando la actividad va a empezar a interactuar con el usuario. Es el último punto antes de que el usuario ya vea la actividad y pueda empezar a interactuar con ella. Siempre después de un onResume() viene un onPause().
- onPause() Llamada cuando el sistema va a empezar una nueva actividad. Esta necesita parar animaciones y parar todo lo que esté haciendo. Hay que intentar que esta llamada dure poco tiempo, porque hasta que no se ejecute este método no arranca la siguiente actividad.

Después de esta llamada puede venir un onResume() si la actividad vuelve a primer plano o un onStop() si se hace invisible para el usuario.

- onStop() Llamada cuando la actividad ya no es visible al usuario, porque otra actividad ha pasado a primer plano. Desde aquí se puede ir al onRestart() si vuelve a primer plano o al onDestroy() si se destruye del todo.
- onDestroy() Esta es la llamada final a la actividad, después de ésta, es totalmente destruida.

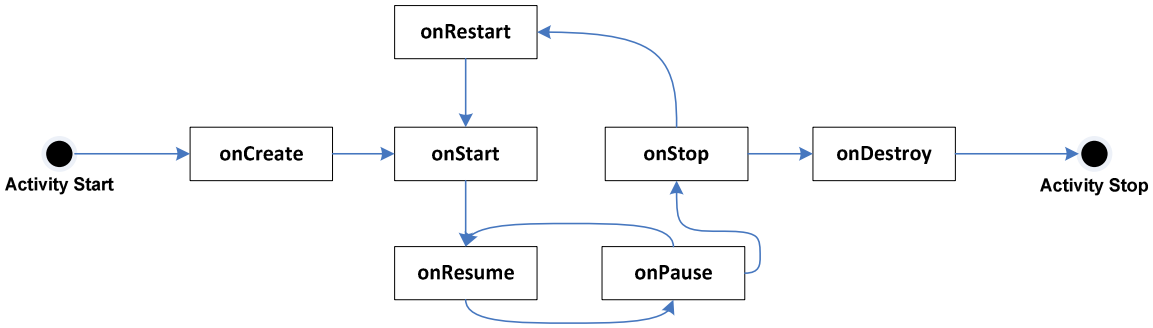


Figura 2.16 - Transición de estados de una actividad.

Toda aplicación consta de una o más actividades (pantallas o ventanas en aplicaciones desktop) las cuales están declaradas en el archivo AndroidManifest.xml. En caso de querer pasar mensajes o interconectar componentes de diferentes aplicaciones se utilizan las Intenciones (Intent). Estas pueden ser utilizadas para:

- Lanzar una actividad, startActivity()
- Lanzar un servicio, startService()
- Lanzar un anuncio de tipo broadcast, sendBroadcast()
- Comunicarnos con un servicio, bindService()

Pueden ser invocadas de dos maneras:

- Invocación explícita: se especifica explícitamente en el código qué componente es el encargado de manejar la intención;
- Invocación implícita: es la plataforma la que determina, a través de un proceso de resolución de intenciones, qué componente es el más apropiado para manejar la intención.

2.2.2. CÓDIGOS DE BARRA 2D

Los códigos de barra 2D son etiquetas de dos dimensiones las cuales pueden codificar en su contenido diferentes tipos de información: texto, códigos de productos, puntos de interés, URLs, etc. y ser leídas o decodificadas de manera rápida y confiable a través de dispositivos específicos (por ejemplo, lectores

de códigos de barra) o cámaras de video. Los datos se codifican por intermedio de la altura y la longitud del símbolo. Los mismos pueden alojar gran cantidad de información. Algunos ejemplos de códigos de barra 1D y 2D se pueden apreciar en la figura 2.17.

Figura 2.17 – Ejemplos de Códigos de Barras.

2.2.2.1. CÓDIGO QR

Un tipo particular de Código de Barra 2D son los Códigos QR. Este código es un módulo para almacenar información en una matriz de puntos o un código de barras bidimensional. Fue creado por la compañía japonesa Denso Wave, subsidiaria de Toyota, en 1994. Se caracteriza por los tres cuadrados que se encuentran en las esquinas que permiten detectar la posición del código al lector. La sigla «QR» se deriva de la frase inglesa *Quick Response*, pues los creadores «Juan Alesandro y EugeDamm» aspiran a que el código permita que su contenido se lea a alta velocidad. Los códigos QR están definidos en el estándar ISO/IEC18004 (QR Code Group, 2012). En la figura 2.18 se pueden apreciar ejemplos de Código QR.

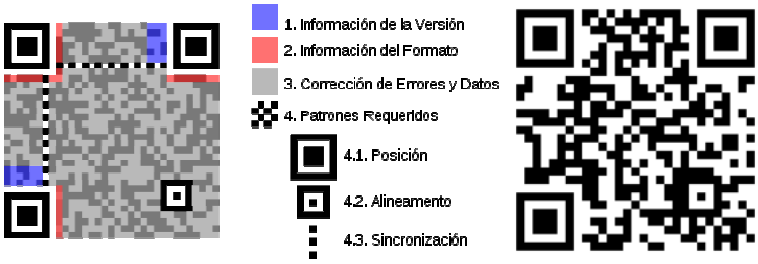


Figura 2.18 - Ejemplos de Códigos QR.

La capacidad de representar datos del Código QR es la siguiente:

- Solo numérico Máx. 7.089 caracteres
- Alfanumérico Máx. 4.296 caracteres

- Binario Máx. 2.953 bytes
- Kanji/Kana Máx. 1.817 caracteres

Existe la posibilidad de generar códigos QR (Url, SMS, Texto, Teléfono) usando diferentes sitios Web, entre ellos:

- Código QR [http://www.codigos-qr.com]
- Kaywa [http://qrcode.kaywa.com]

Para realizar la lectura de un código QR desde un dispositivo se pueden utilizar programas lectores de códigos QR. Entre los más conocidos se encuentran: Barcode Scanner, Inigma Reader y QuickMark Reader. Estos se deben instalar en el dispositivo.

2.2.2.2. LECTURA DE CÓDIGO QR EMBEBIDO EN UNA APLICACIÓN ANDROID

Como se mencionó anteriormente, un Código QR se puede leer mediante el uso de un programa, por ejemplo, QuickMark Reader. Estos programas pueden ser instalados en Android, pero dicha funcionalidad no puede ser resuelta nativamente por una aplicación Android Cliente, es por eso que se deben utilizar librerías adicionales que prevean esa funcionalidad, una de ellas es ZXing. Esta es una librería open-source para el procesamiento (codificación y decodificación) de imágenes de códigos de barra multi-formato 1D/2D; está implementada en Java (Proyecto ZXing, 2012). La misma utiliza la cámara del dispositivo móvil para la captura del código para su posterior procesamiento. Los formatos de códigos de barras soportados por la librería ZXing son:

- UPC-A and UPC-E
- EAN-8 and EAN-13
- Code 39
- Code 93
- Code 128
- ITF
- Codabar
- RSS-14 (allvariants)
- QR Code
- Data Matrix
- Aztec ('beta' quality)
- PDF 417 ('alpha' quality)

La librería ZXing se divide en varios componentes principales:

- Core: Núcleo de decodificación de imágenes biblioteca, y el código de prueba
- Javase: Código de cliente J2SE-Especifico
- Android: Cliente Android (Barcode Scanner)
- Android test: Aplicación Android de Prueba
- Android-integration: Integración de aplicaciones Android con Barcode Scanner, mediante Intent.

2.2.3. ACCESO A WEB SERVICES DESDE ANDROID: LIBRERIAS KSOAP Y HTTP-CLIENT

KSOAP es una librería (J2ME CLDC/CDC/MIDP) para clientes de servicios web SOAP orientada a entornos móviles con limitaciones. Permite, entre otras cosas, consumir servicios web SOAP desde una aplicación instalada en un dispositivo móvil Android.

En la figura 2.19 se puede apreciar un ejemplo de código donde se consume un servicio web SOAP desde una aplicación Android.

```
public String ejecutar_ksoap_a_soap(String IP) throws Exception{
        try {
          String Titulo;
          String accion = "http://srv-puntos-interes.com.ar/svcPuntos_Interes";
          String metodo = "Obtener_Puntos_Interes";
          String namespace = "http://srv-puntos-interes.com.ar/";
          String url = "http://srv-puntos-interes.com.ar/svcPuntos_Interes.asmx";

          SoapObject request = new SoapObject(namespace, metodo);
          SoapSerializationEnvelope sobre = new SoapSerializationEnvelope(SoapEnvelope.VER11);
          sobre.dotNet = true;
          request.addProperty("IPAddress", IP);
          sobre.setOutputSoapObject(request);

          HttpTransportSE transporte = new HttpTransportSE(url);
          transporte.call(accion, sobre);
          SoapObject resultado = (SoapObject) sobre.getResponse();

          Titulo = resultado.getProperty("Titulo_Punto_Interes").toString();

          return Titulo;

        }catch (Exception e){
                return "";
        }
}
```

Figura 2.19 - Código ejemplo de consulta a un servicio web SOAP a través de la librería Ksoap.

Por otra parte, **HTTP-client** es una interfaz que deriva de la clase *org.apache.http.client.HttpClient* para un cliente HTTP. Encapsula una mezcla heterogénea de objetos necesarios para ejecutar peticiones HTTP durante la manipulación de cookies, autenticación, gestión de la conexión, y otras características.

La seguridad de los clientes HTTP depende de la implementación y configuración del cliente. Con la misma se pueden consultar servicios web RESTful desde una aplicación Android a través del envío de mensajes HTTP, ejecutando operaciones de tipo HTTP (POST, GET, PUT, y DELETE).

En la figura 2.20 se puede apreciar un ejemplo de código donde se consume un servicio web RESTful desde una aplicación Android.

```
public String ejecutar_http_cliente_a_RESTful(String strid)
  {
                                HttpClient httpClient = new DefaultHttpClient();
                                HttpGet del = new HttpGet("http://srv-puntos-interes.com.ar/svcPuntos_Interes.svc/Puntos_Interes/" + strid);

                                del.setHeader("content-type", "application/json");
                                try
    {
                HttpResponse resp = httpClient.execute(del);
                String respStr = EntityUtils.toString(resp.getEntity());
                JSONObject respJSON = new JSONObject(respStr);
                String Titulo = respJSON.getString("Titulo_Punto_Interes");
                return Titulo;

    }
    catch(Exception ex)
    {
                return "";
    }

  }
```

Figura 2.20 - Código de Ejemplo de consulta a un Servicio Web RESTful a través de la interfaz HTTP-CLIENT.

2.2.4. COBERTURA DE LA RED DE TELEFONIA MÓVIL[4]

La reglamentación Argentina no se encuentra ligada a ningún tipo de tecnología, por lo cual los prestadores AMX ARGENTINA S.A. (CLARO), TELEFÓNICA MOVILES ARGENTINA S.A. (MOVISTAR), TELECOM PERSONAL S.A. (PERSONAL) han variado las tecnologías usadas a través de los años, según la evolución.

En los primeros tiempos, se desplegaron tecnologías 1G como fueron AMPS. A fines de los años 90, se introdujeron las tecnologías 2G, como fueron D-AMPS, TDMA y CDMA. A principios de siglo se introdujo GSM, actualmente utilizada por todos los prestadores.

[4] Obtenido del sitio web de la Comisión Nacional de Comunicaciones < http://www.cnc.gov.ar>

Por otro lado, las necesidades de comunicaciones de datos móviles han permitido que las redes celulares, que originalmente fueron diseñadas para el transporte de comunicaciones de voz, se adapten a los nuevos requerimientos. En la actualidad, los prestadores han implementado tecnologías de última generación 3G, como UMTS-HSDPA, las cuales brindan una mayor tasa de transferencia de datos que las tecnología de 2,5 G como GPRS y EDGE.

A continuación se enuncian conceptos importantes en el entorno móvil: GSM, tarjeta SIM, bandas de frecuencia y 3G.

GSM son las siglas de Global System for Mobile communications (Sistema Global para las comunicaciones Móviles). Es el sistema de teléfono móvil digital más utilizado y el estándar de facto para teléfonos móviles. Definido originalmente como estándar europeo abierto para que una red digital de teléfono móvil soporte voz, datos, mensajes de texto y roaming en varios países. El GSM es ahora uno de los estándares digitales inalámbricos más importantes del mundo, estando presente en más de 160 países y abarcando el 80 por ciento del total del mercado móvil digital. El sistema GSM ha desplazado en la Argentina y en muchos otros países a otros sistemas, tales como el CDMA. CDMA son las siglas del inglés Code Division Multiple Access (Sistema de Acceso Múltiple por División de Códigos), es también un sistema celular basado en tecnología digital, que se sigue utilizando en algunos países, por ejemplo en EE.UU. 3G significa tercera generación. En la evolución de la telefonía celular se reconocen tres generaciones: la primera comprende a los celulares analógicos, la segunda a los celulares digitales y la tercera a los celulares multimedia.

La principal característica de la tecnología GSM es que permite utilizar el servicio de telefonía móvil con el mismo número de teléfono en todos los países que la emplean (roaming internacional). Se caracteriza por la utilización de una tarjeta denominada SIM Card (o chip), que contiene la información de la cuenta, y en la cual pueden almacenarse todos los contactos, información de agenda, etc. Los equipos utilizados con GSM son de alta tecnología y calidad, y pueden ser reemplazados sin depender de ningún trámite administrativo (simplemente trasladando la tarjeta SIM de un equipo al otro).

Otra característica importante es la aplicación sobre las redes GSM de las tecnologías GPRS y EDGE, que posibilitan la transmisión de datos de alta velocidad.

La **tarjeta SIM** (acrónimo de Subscriber Identify Module, 'Módulo de Identificación del Suscriptor' MIS) es una tarjeta inteligente desmontable usada en los teléfonos móviles GSM que almacena de forma segura la clave de servicio del suscriptor usada para identificarse ante la red, de forma que sea posible cambiar la línea de un terminal a otro simplemente cambiando la tarjeta. El uso de la tarjeta SIM es obligatorio en las redes GSM, ya que es la tarjeta SIM la que representa la línea del abonado, y no el terminal o teléfono donde se la instala.

Las **bandas de frecuencias** utilizadas por el sistema GSM en América son diferentes a las usadas en Europa. En efecto, en América se utilizan las 850 y 1900 MHz, en cambio en Europa se usan las bandas de 900 y 1800 MHz. Muy pocos teléfonos móviles a nivel mundial son capaces de funcionar con las cuatro bandas disponibles, dichos aparatos son conocidos como tetra-banda. En cambio, son más comunes los equipos tri-banda, que además de soportar las dos bandas correspondientes a la región para la cual fueron pensados, incluyen una tercera, correspondiente a la otra región. Se los conoce como tri-banda Europeo (900, 1800, 1900) o tri-banda Americano (850, 1800, 1900) según el caso. Los equipos de baja gama suelen ser por su parte dual-banda, es decir que sólo están pensados para ser utilizados en una determinada región.

Los sistemas móviles **3G** brindan servicios multimedia, mejorando los servicios ya existentes de voz, textos, datos, música y video. La tecnología **3G-WCDMA** está basada en los estándares actuales de GSM, pero ha evolucionado para incluir una nueva interfaz de radio de mayor velocidad binaria, que permite el funcionamiento pleno de los servicios mencionados y su uso simultáneo. La 3ra. Generación coexiste con la anterior, es decir los teléfonos GSM continuarán funcionando. Sólo se deberá adquirir un equipo 3G si se desea tener acceso a más servicios, teniendo en cuenta que los teléfonos 3G son dispositivos de alta gama, que incluyen poderosas capacidades de procesamiento que los asemejan a pequeñas computadoras portátiles.

2.2.4.1. ANÁLISIS DEL ENTORNO MÓVIL EN LA PROVINCIA DE SANTIAGO DEL ESTERO

Se llevó a cabo un relevamiento basado en entrevistas a prestadores de telefonía celular de la Provincia de Santiago del Estero, con el propósito de obtener información acerca de las tecnologías de comunicación que se utilizan en la región. A continuación se describe la información obtenida.

En la provincia de Santiago del estero la telefonía celular posee cobertura 3G y 2G. Dichas redes son proporcionados por las empresas Movistar, Personal y Claro. Como ocurre casi en todas las provincias, esta cobertura depende de la localidad y su cantidad poblacional. En particular, la provincia posee conectividad 3G en el microcentro de las ciudades de Santiago del Estero (Dpto. Capital), La Banda (Dpto. Banda) y Termas de Río Hondo (Dpto. Río Hondo), como puede verse en las figuras 2.21 a 2.24. Esto conlleva a que existen inconvenientes de conectividad en los puntos de interés turísticos que se encuentran fuera de esta cobertura, que en su mayoría lo están. Pero la implementación de equipos específicos (Booster, Antenas Yagi, etc.) en dichos lugares, proveerán una mejoría en la conectividad de los dispositivos que deban utilizar la red 3G para la transmisión de datos a través de la misma.

Es conocido que la cobertura 3G crecerá con el transcurso del tiempo, como lo manifiesta la entrevista realizada al gerente de la empresa Movistar (ver Anexo 2), esto es beneficioso para el entorno móvil de

la provincia ya que permitirá el desarrollo de sistemas no solo en el plano turístico sino también comercial y de la salud entre otros.

Para una mejor comprensión de la cobertura de la red móvil en la provincia de Santiago del Estero, se presenta a continuación la ubicación de algunas torres en las localidades de Santiago del Estero y Termas de Río Hondo.

Localidad: Santiago del Estero

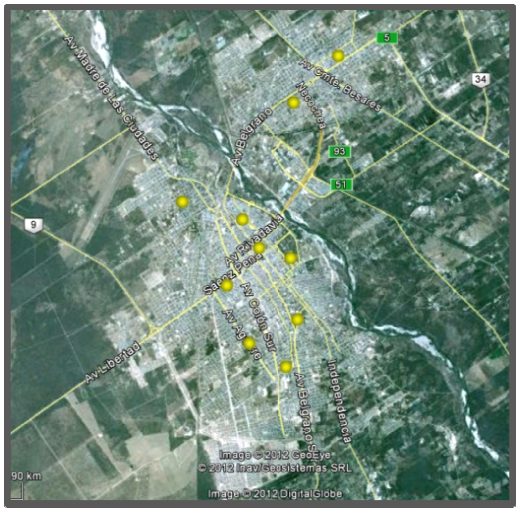


Figura 2.21 – Cobertura 2G

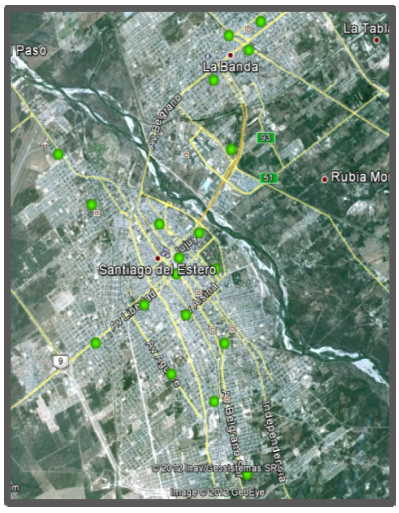


Figura 2.22 – Cobertura 3G

Localidad: Termas de Río Hondo

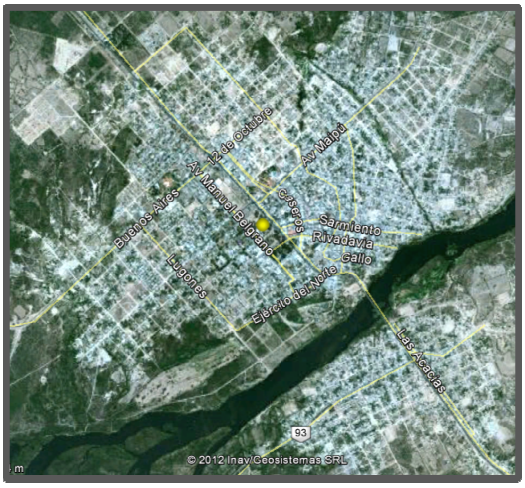


Figura 2.23 – Cobertura 2G

Figura 2.24 – Cobertura 3G

En cuanto a la tecnología de los dispositivos móviles que son utilizados en la provincia, se observa que el SO más utilizado es Android, seguido por RIM (de BlacKBerry). La gente adquiere dispositivos que posibilitan el acceso a redes sociales como primer requisito al momento de su compra. El empuje de las empresas de telefonía hacia el uso de paquetes de datos en las cuentas de sus usuarios y el avance en la tecnología de los dispositivos hará que en breve tiempo se tengan más Smartphone en el entorno móvil de la provincia de Santiago del Estero. Esto se transformará en un problema para las empresas

de telefonías de la provincia ya que habrá más usuarios que deberán conectarse a la red 3G y no solo en el micro centro de sus principales localidades.

2.2.5. HERRAMIENTAS PARA MEDIR LA EFICIENCIA EN APLICACIONES MÓVILES

A continuación se detallarán las herramientas que serán útiles para el proceso de medición de los atributos de las subcaracterísticas que permitirán evaluar la eficiencia de las aplicaciones que usan las arquitecturas alternativas propuestas. Estas herramientas determinan, entre otras cosas, el consumo de batería de los recursos principales en un periodo de tiempo determinado, la tasa de transferencia de datos a través de las diferentes redes de comunicación (3G, WI-FI) por aplicación y/o sistema operativo.

2.2.5.1. POWERTUTOR

PowerTutor es una aplicación desarrollada por la Universidad de Michigan, con apoyo de Google entre otros. Esta aplicación permite la medición de la energía consumida por los principales componentes del dispositivo: CPU, Pantalla LCD/OLED, GPS, placa de red Wi-Fi y 3G, como así también el audio.

Para poder determinar el consumo, en un rango del 5% de los valores reales, utiliza un modelo de mediciones directas sobre los cambios de estados en la administración de energía del dispositivo. PowerTutor fue construido para los dispositivos HTC G1, G2 HTC y NexusOne, pero se ejecuta también en otros teléfonos móviles. Existen algunas dificultades, por ejemplo en el dispositivo Android SII no permite la monitorización del WI-FI ni 3G. Para su instalación requiere Android 1.5 o más. Su última actualización en la play.google.com es de Abril de 2013. Es una herramienta muy útil ya que a los desarrolladores de software le permite determinar o estimar el impacto que tiene una aplicación en la duración de la batería del dispositivo móvil.

Posee diversas funciones, entre las más destacadas:

- Emisión de informes detallados en archivos electrónicos (texto)
- Medición directa del consumo de batería de los componentes principales del teléfono móvil por aplicación
- Vista detallada del consumo de energía de una aplicación en un lapso de tiempo
- Información detallada del estado de la Batería del dispositivo móvil
- Gráficos estadísticos: Torta y Trazo

En las figuras 2.25 y 2.26 se presentan algunas capturas de pantallas de la aplicación.

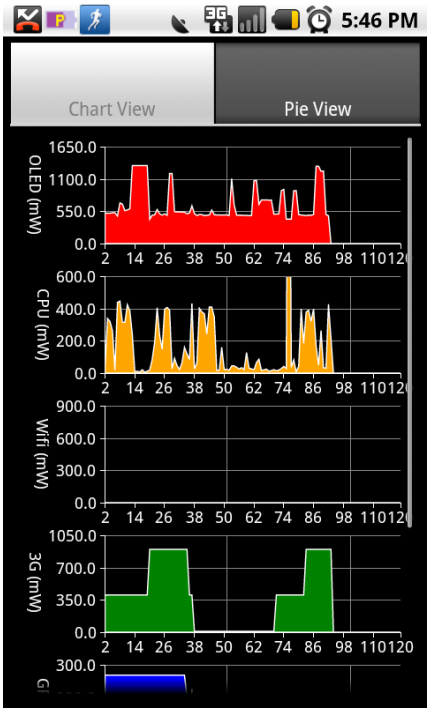


Figura 2.25 - Vista de la gráfica de estadísticas de PowerTutor

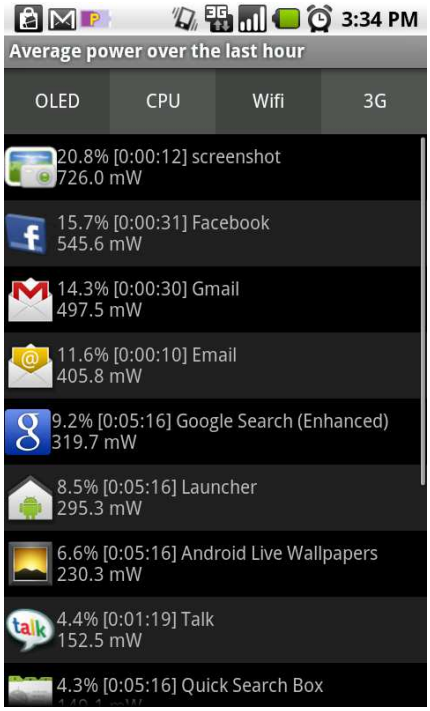


Figura 2.26 - Vista de la pantalla principal de PowerTutor

2.2.5.2. TRAFFIC MONITOR

Traffic Monitor es una aplicación desarrollada para la plataforma Android que informa sobre el consumo de tráfico de datos en las interfaces móvil y WI-FI. Verifica el rendimiento de su conexión a través la prueba de velocidad integrada. Su última actualización en la play.google.com, teniendo en cuenta la fecha de redacción de este informe) es el 15 de octubre de 2013 y requiere Android 2.2 o más para su instalación.

Las funciones más destacadas de Traffic Monitor Plus son:

- Contador de tráfico de datos para redes WI-FI y móvil, separado por subida y descarga.
- Estadísticas de tráfico por aplicación y por día/semana/mes (figura 2.27).
- Mide la velocidad y el retraso de la conexión inalámbrica.
- Evaluación del rendimiento de su conexión (figura 2.29).
- Comprobar aplicaciones en ejecución incluyendo el uso de la memoria principal (figura 2.28).
- Información sobre la red móvil y la celda.
- Test de velocidad integrado (figura 2.26)

En las figuras 2.26 a 2.29 se presentan algunas capturas de pantallas de la aplicación.

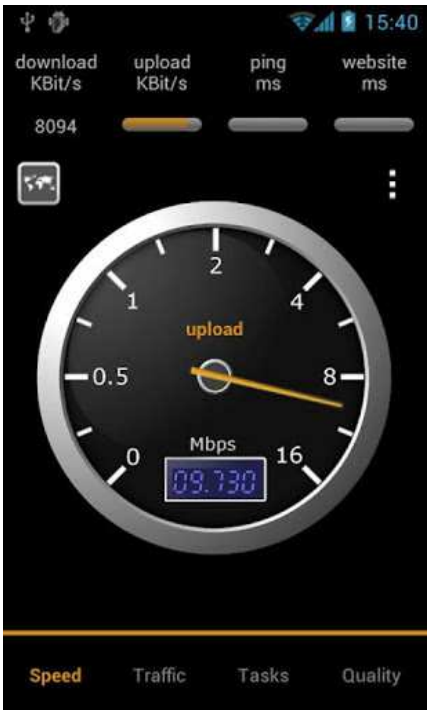


Figura 2.26 - Testeo de velocidad

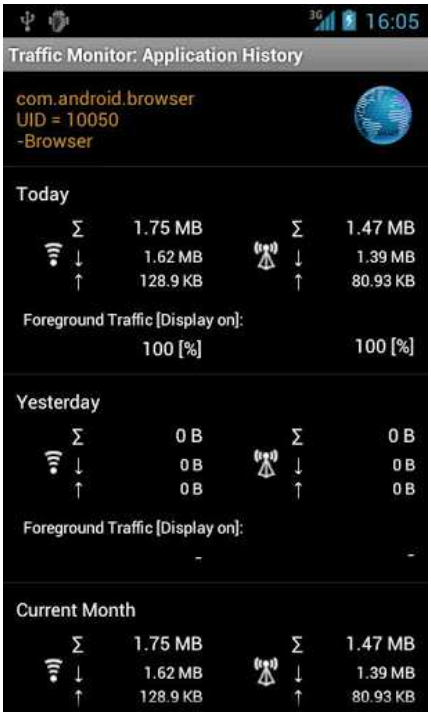


Figura 2.27 - Estadísticas de tráfico por aplicación y por día/semana/mes

Figura 2.28 - Consumo de memoria principal por aplicación

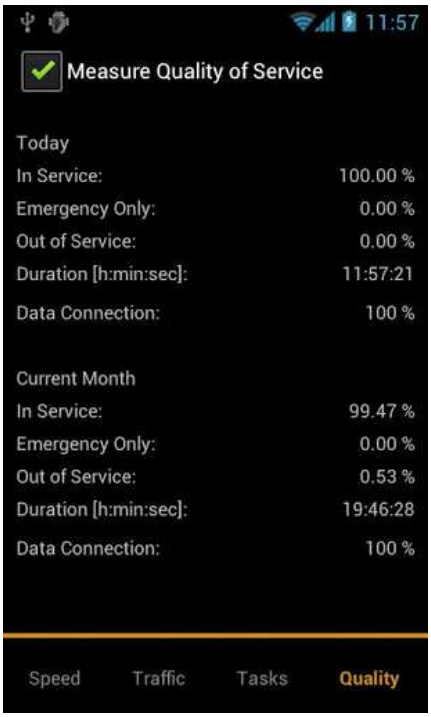


Figura 2.29 - Evaluación del rendimiento de su conexión

3. CAPÍTULO 3: ANÁLISIS DE ARQUITECTURAS MÓVILES.

En la Sección 2.1.2.1 se presentaron arquitecturas móviles de manera conceptual y esquemática independientemente de la plataforma utilizada. En este capítulo se describirá cómo las arquitecturas cliente-servidor, híbrida o web, pueden ser usadas como plataformas particulares para dar solución a las aplicaciones de m-turismo. Se describirán las características más importantes que poseen estas, poniendo énfasis en las tecnologías utilizadas para el posicionamiento del dispositivo móvil y para la comunicación con la base de datos de Puntos de Interés (PI) para atender las consultas del usuario.

3.1. ARQUITECTURA HÍBRIDA

Se propone en este apartado una arquitectura híbrida para el soporte de aplicaciones de m-turismo, definiéndola tanto desde el lado del cliente como del servidor. En las figuras 3.1 y 3.2 se pueden apreciar esquemas, a distinto nivel de detalle, de la arquitectura creada.

Las principales características son las siguientes:

- Una Aplicación Android funciona como cliente. Esta contiene la lógica e interfaz de la aplicación. Además, está integrada con la librería ZXIng la cual permite embeber un lector de código de barra 2D (en particular, Barcode Scanner). Esta integración se realiza mediante la creación de un Intent en la aplicación Android que ejecuta el Barcode Scanner. La comunicación con el servidor se realiza utilizando la librería Http-Client.
- Un servidor de servicios web provee la información de los PI. Este servidor se implementa usando WCF (Windows Communication Framework). Este servidor puede acceder a los datos de los PI.
- La comunicación entre el Cliente y el servidor se puede realizar a través de una red de comunicaciones, ya sea Lan, Wan, Internet, etc.
- Para determinar el posicionamiento del usuario se utilizan código QR, los cuales son leídos por la aplicación Android, mediante el uso del Barcode Scanner. Cada uno de los Códigos QR tiene asociado un PI.
- Las consultas sobre los PI se realizan mediante el uso del protocolo de comunicación RESTful, y los PI son enviados en formato JSON.

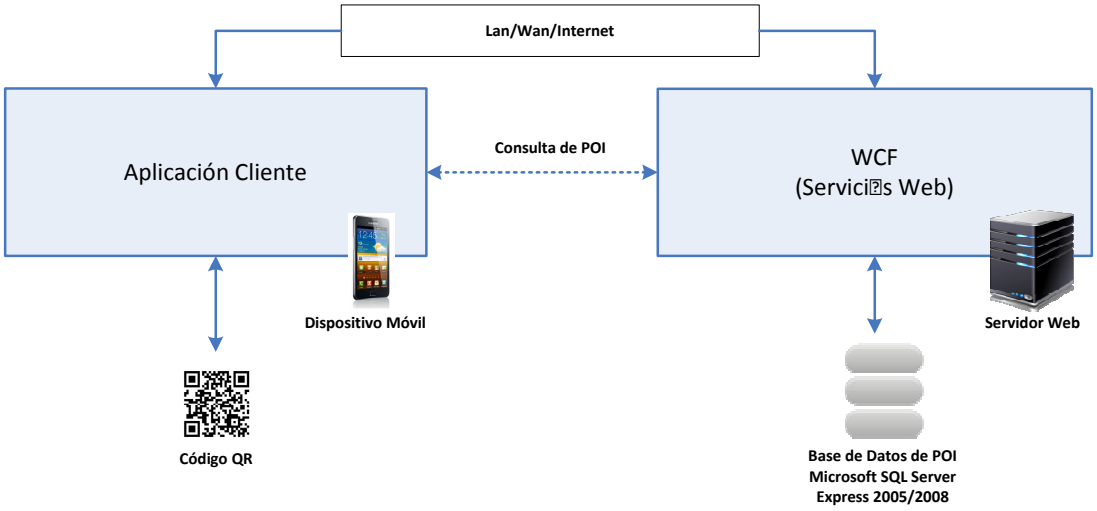


Figura 3.1 - Arquitectura híbrida específica para m-turism? usand? un cliente Andr?id.

A c?ntinuación se describe cóm? es el fluj? de c?ntr?l entre que el usuari? inicia la aplicación hasta que l?gra visualizar la inf?rmación de un PI.

1. El usuari? ejecuta la aplicación Andr?id.
2. El usuari? elije la ?pción de leer/escanear un códig? QR; internamente la aplicación c?nv?ca y ejecuta el Barc?de Scanner a través de Intent y la librería ZXing; el usuari? visualiza en pantalla el scanner de la aplicación Barc?de Scanner.
3. El usuari? escanea un códig? QR.
4. La aplicación Andr?id recibe el códig? QR capturad? (Códig?-URL) en la aplicación Barc?de Scanner.
5. La aplicación Andr?id, a través de la interfaz http-client, se c?necta a la aplicación servid?r, p?r medi? de mensajes http (RESTful) para ?btener la inf?rmación del PI que se c?rresp?nde c?n el códig? QR leíd?.
6. El servid?r de servici?s web busca en la base de dat?s el PI adecuad? y genera un archiv? JSON que es enviad? c?m? respuesta a la aplicación Andr?id.
7. La aplicación Andr?id muestra la inf?rmación del PI al usuari?.

De esta manera, se puede apreciar cóm? funci?na una Arquitectura Híbrida para una aplicación de m-turism?. T?da esta descripción puede ser seguida en la figura 3.2.

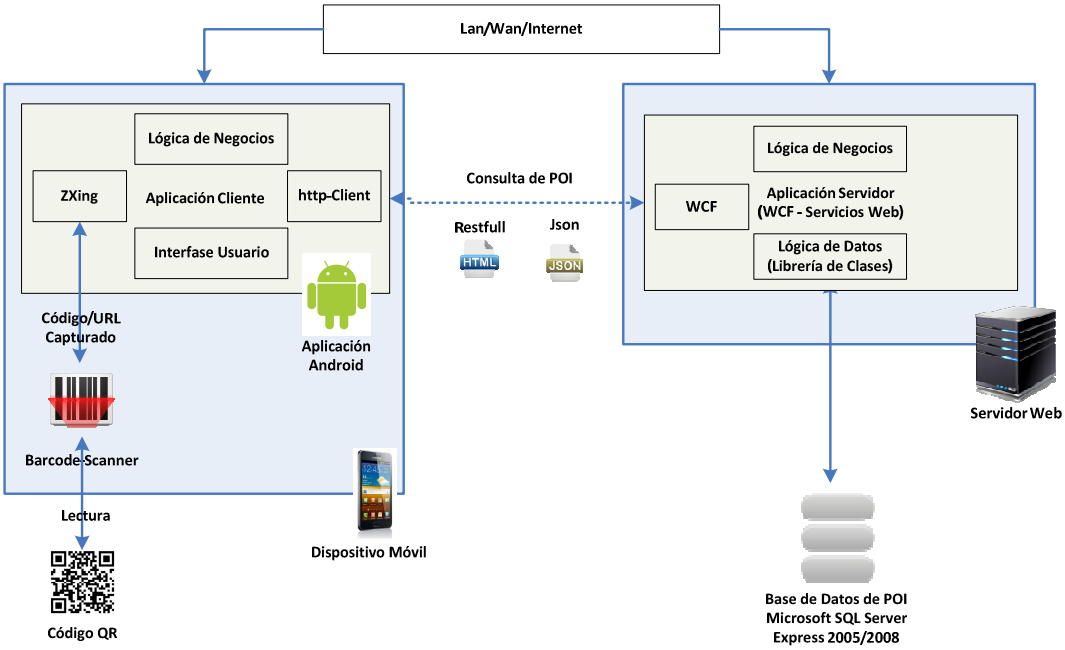


Figura 3.2 - Vista detallada de la arquitectura híbrida específica para m-turismo usando un cliente Android.

3.2. ARQUITECTURA WEB

En esta sección se presenta una arquitectura web para el soporte de aplicaciones de m-turismo donde se definirán para el servidor una arquitectura particular, mientras que en el cliente se contará con un browser web móvil. Esta arquitectura permite independencia de plataforma del lado del cliente como así también no desarrollar la solución para cada una de los Sistemas Operativos móviles del Mercado.

En las figuras 3.3 y 3.4 se pueden apreciar esquemas, a distinto nivel de detalle, de la arquitectura creada.

Las principales características son las siguientes:

- Aplicación Web que implementa el patrón de diseño MVC (en particular, ASP.Net MVC). Esta aplicación se conecta con el servidor de servicios web para obtener los datos de los PI.
- Servidor de servicios web que provee la información de los PI. Este servidor se implementa usando WCF (Windows Communication Framework). Este servidor puede acceder a los datos de los POI.
- La comunicación entre el browser y los servidores (ya sea el de la Aplicación Web o servicios web) se puede realizar a través de una red de comunicaciones, ya sea Lan, Wan, Internet, etc.

- Para determinar el posicionamiento del usuario se utilizan códigos QR, los cuales son leídos por el Barcode Scanner. Como cada Código QR representa una URL, se envía el requerimiento a la Aplicación Web, y la respuesta es visualizada en el browser del dispositivo.
- Las consultas, que hace la Aplicación Web al servidor de servicios web sobre los PI, se realizan mediante el uso del protocolo de comunicación RESTful; y los PI son enviados en formato JSON.

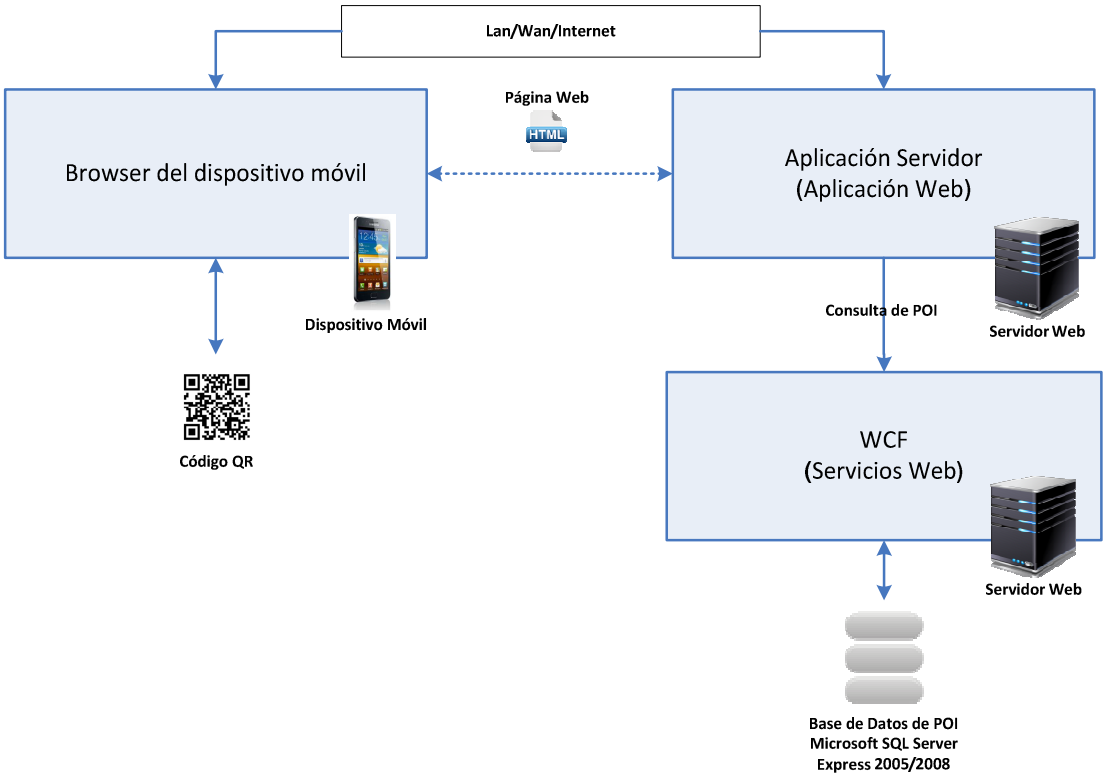


Figura 3.3 - Arquitectura web especificada para m-turismo.

El flujo de control desde que el usuario lee un código QR hasta que logra visualizar la información de un PI en el browser se detalla a continuación:

1. El usuario ejecuta el Barcode Scanner.
2. El usuario lee un código QR; al ser éste una URL, el Barcode Scanner abre el Browser del dispositivo y este último accede a la URL especificada en el código QR (Aplicación Web).
3. La Aplicación Web se comunica por medio de http (RESTful) con el servidor de servicios web para obtener la información del PI que se corresponde con el código QR leído.
4. El servidor de servicios web busca en su base el PI adecuado y genera un archivo JSON que es enviado como respuesta a la Aplicación Web.
5. La Aplicación Web envía la respuesta al usuario, el cual recibe en el browser la información del PI al usuario.

De esta manera, se puede apreciar cómo funciona una arquitectura web para una aplicación de m-turismo. Toda esta descripción puede ser seguida en la figura 3.4.

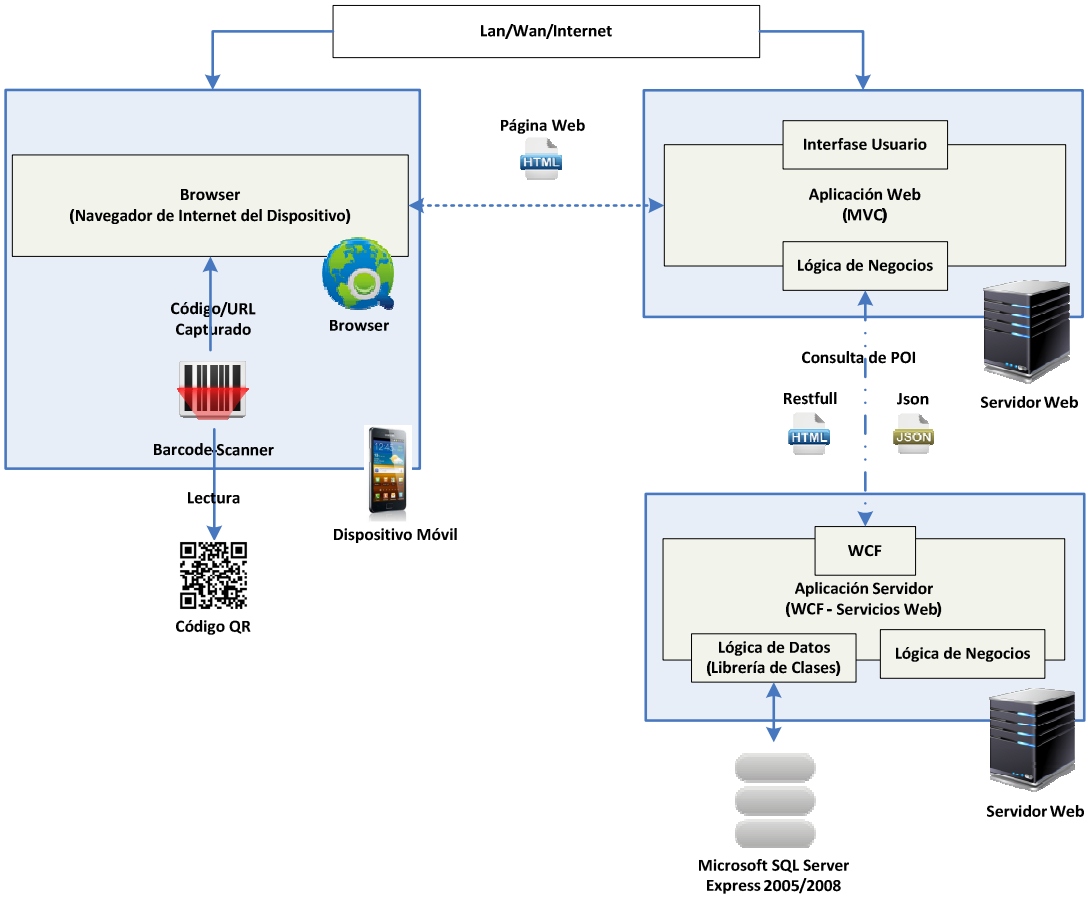


Figura 3.4 - Vista detallada de la arquitectura web especificada para m-turismo.

3.3. ASPECTOS COMUNES A AMBAS ARQUITECTURAS

Como se puede apreciar en los apartados 3.1 y 3.2, ambas arquitecturas poseen aspectos comunes. Uno de ellos es que ambas usan la misma forma de posicionamiento a través de códigos QR. Además, ambas usan el mismo programa de lectura de códigos QR, el Barcode Scanner. En la figura 3.5 se muestra el software mencionado y un ejemplo de código QR. Cabe aclarar que en la arquitectura híbrida este lector está embebido dentro de la aplicación Android, pero el funcionamiento de la misma, a través del software Barcode Scanner, es el mismo en ambas arquitecturas.

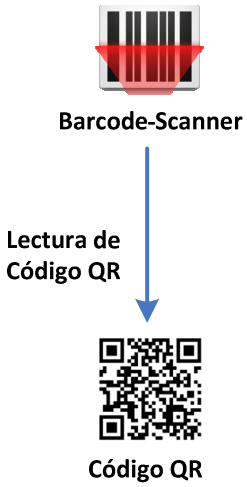


Figura 3.5 - Barcode Scanner usado en ambas arquitecturas

Otro aspecto en común en ambas arquitecturas, es la forma de obtener un PI, para lo cual se utiliza un servidor de servicios web. Este servidor funciona a través del protocolo de comunicación RESTful, y el formato en el que devuelve los PI es JSON. En la figura 3.6 se puede apreciar el servidor de servicios web, el cual puede ser accedido desde cualquier aplicación. En el caso de la arquitectura híbrida es accedido desde la aplicación Android, mientras que en la arquitectura web es accedido desde la Aplicación Web.

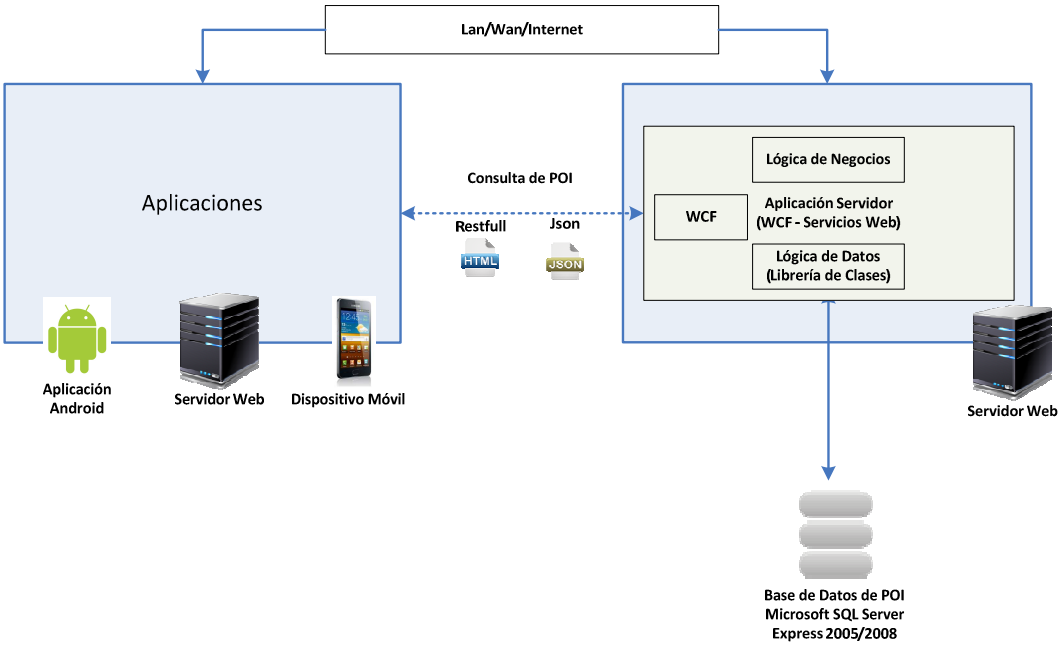


Figura 3.6 - Servidor de servicios web de PI.

Estas características comunes, como son la lectura de códigos QR y el servidor de servicios web, permiten que la evaluación de la eficiencia se lleve a cabo enfocándose en los aspectos diferentes que tienen ambas arquitecturas. En la arquitectura híbrida se cuenta con una aplicación cliente nativa, en particular, Android. Mientras que en la arquitectura web se tiene un browser del lado del cliente y una Aplicación Web ejecutándose en el servidor.

4. CAPÍTULO 4: EVALUACIÓN DE EFICIENCIA DE APLICACIONES MÓVILES CON DIFERENTES ARQUITECTURAS.

Como se mencionó en el apartado 2.1.6, del Marco Teórico, la característica eficiencia se mide en función de subcaracterísticas y éstas, a su vez, se miden a partir de sus atributos. Estos últimos son los que se miden utilizando métricas directas o indirectas.

En el marco de este proyecto se mide la calidad externa de dos prototipos de aplicaciones de m-turismo, desarrolladas con diferentes arquitecturas: híbrida y web. Dichas arquitecturas se evalúan en dos ambientes (Móvil y Wi-Fi) a partir de los mismos atributos de calidad. Para evaluar la calidad, se sigue básicamente el proceso de la estrategia GOCAME, descripto en el apartado 2.1.6.3. Los pasos seguidos son:

1. Desarrollo de los prototipos de aplicaciones con arquitectura híbrida y web
2. Diseño y preparación de los escenarios de prueba y definición de los casos de prueba
3. Evaluación de la calidad de los prototipos
    a. Definición de los requisitos no funcionales o NFR
    b. Diseño de las mediciones
    c. Diseño de la evaluación
    d. Implementación de las mediciones
    e. Implementación de la evaluación
4. Análisis y recomendación

En este capítulo se presentan los resultados de las etapas 1 a 3.c, las cuales responden a las tareas de desarrollo del prototipo, diseño de los ambientes y casos de pruebas y de las mediciones; todas realizadas con el objetivo de evaluar la eficiencia de aplicaciones con arquitecturas híbrida y web.

4.1. DESARROLLO DE PROTOTIPOS DE M-TURISMO

Dada la necesidad de contar en la región con aplicaciones de turismo basadas en posicionamiento, se desarrollaron prototipos en este dominio de aplicación.

Si bien son dos prototipos, uno para cada arquitectura, ambos responden al mismo conjunto de requisitos. Fueron desarrollados siguiendo el modelo de desarrollo ágil denominado Programación Extrema o XP.

En cuanto a la funcionalidad, ambos prototipos responden al siguiente requerimiento:

Un turista visita por primera vez la ciudad de Termas de Río Hondo. Cuenta con un dispositivo móvil con conexión a Internet y lector de código QR. Mientras camina por la pequeña ciudad, desea conocer dónde se encuentran los puntos turísticos más relevantes, con el propósito de ir a visitarlos. Una vez que llega a cada lugar, accede a información detallada del mismo.

El alcance de la prototipación, a efectos de permitir la medición, abarca la última parte del párrafo anterior que define la siguiente historia de usuario:

H1: El turista puede visualizar la información de un PI en su dispositivo móvil a través de la lectura de su etiqueta QR.

A continuación se describe el funcionamiento de ambos prototipos desarrollados y las herramientas que se usaron en la construcción de los mismos. En el Anexo 3 se presenta la documentación del desarrollo con XP. En el Anexo 4 se presentan ejemplos del código de cada prototipo.

4.1.1. PROTOTIPO CON ARQUITECTURA HÍBRIDA

FUNCIONAMIENTO

El usuario accede a la aplicación Android presionando el icono de la misma en el menú principal del dispositivo. Al abrirse la aplicación se presenta la pantalla principal de la aplicación (ver figura 4.1). El usuario tiene la posibilidad de escanear una etiqueta con un código QR o bien ingresar el Id (Identificador o código) del PI. El usuario presiona el botón Scanner y la aplicación ejecuta el Barcode Scanner mostrando el lector del mismo (ver figura 4.3). Una vez enfocada la etiqueta QR, el Barcode Scanner decodifica el código QR leído y envía el dato a la aplicación Android, la cual consulta la aplicación WCF, a través de la interfaz http-client. De esta manera obtiene la información detallada del PI identificado con la etiqueta QR leída. (Ver figura 4.2).

Figura 4.1 - Pantalla principal de aplicación Android Cliente.

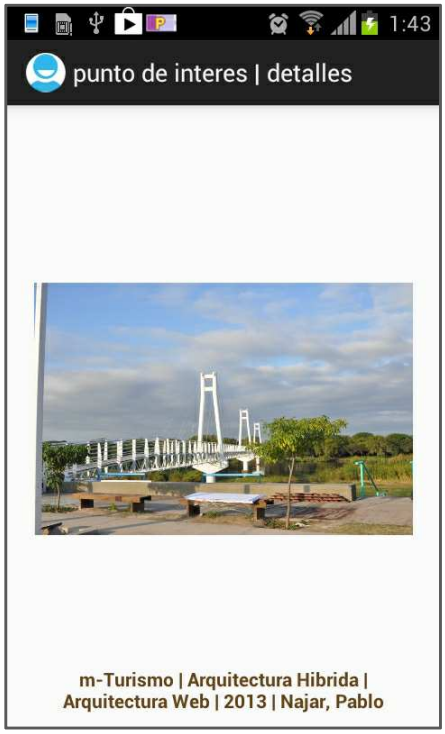


Figura 4.2 - Pantalla detalles de la aplicación Android Cliente.

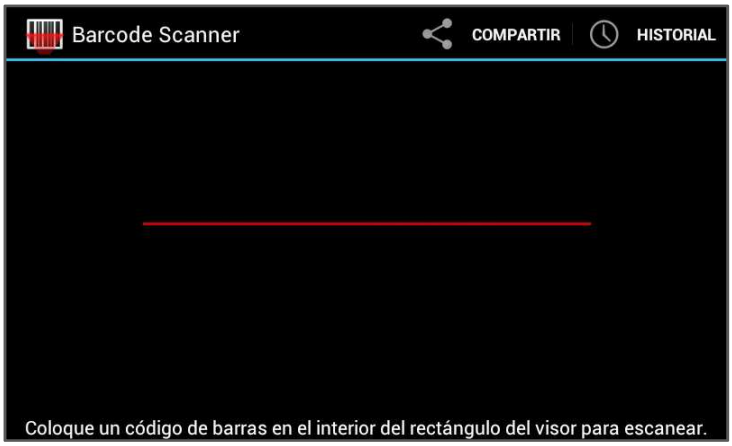


Figura 4.3 - Visualización del lector (Barcode Scanner).

HERRAMIENTAS UTILIZADAS

- Plataforma / S.O.: Andorid v3/4.
- IDE de Desarrollo: Eclipse for Java Developers.
- Lenguaje de Programación: Java.
- Librerías/Api/SDK:
    - Android SDK (Kit de desarrollo de software).
    - Http-Client(Para la conexión a servicios web de tipo RESTful)
    - ZXing (Procesamiento Multi-formato de imagen de código de barras 1D/2D para clientes Android)

TECNOLOGÍA DEL SERVIDOR

- S.O.: Windows 2008 Server.
- Servidor WEB: IIS 7.0.
- IDE: Visual Studio .NET.
- Lenguaje de Programación: Visual Studio .NET WCF (Visual Basic .NET).
- Base de Datos de Puntos de Interés: SQL SERVER EXPRESS 2008.

Esta sección se relaciona con el modelo de las figuras 3.1 y 3.2.

4.1.2. PROTOTIPO CON ARQUITECTURA WEB

FUNCIONAMIENTO

El usuario ejecuta el Barcode Scanner presionando el icono de la misma en el menú principal del dispositivo. Al abrirse la aplicación se presenta la pantalla principal de la aplicación, la misma varía de acuerdo al dispositivo donde se esté ejecutando. El usuario ejecuta la función Scanner y a continuación se visualizará el lector del mismo. Una vez enfocada la etiqueta QR, el Barcode Scanner decodifica el código QR leído y al ser éste un URL, abre el browser del dispositivo y envía el requerimiento de apertura de URL este último. Un vez que el browser ingresa a la Aplicación Web, él mostrará la página Web devuelta por esta última con los datos del PI (ver figuras 4.4 y 4.5). De esta manera se obtiene la información detallada del PI identificado con la etiqueta QR leída.

Figura 4.4 - Visualización de información detallada del PI en el browser del dispositivo móvil.

Figura 4.5 - Visualización de imágenes del PI en el browser del dispositivo móvil.

HERRAMIENTAS UTILIZADAS

- Plataforma / S.O.: Windows 7.
- IDE de Desarrollo: Visual Studio .NET.
- Lenguaje de Programación: ASP.NET MVC 3.5 (Visual Basic .NET).

TECNOLOGÍA DEL SERVIDOR

- S.O.: Windows 2008 Server.
- Servidor WEB: IIS 7.0.
- IDE: Visual Studio .NET.
- Lenguaje de Programación: Visual Studio .NET WCF (Visual Basic .NET).
- Base de Datos de Puntos de Interés: SQL SERVER EXPRESS 2008.

Esta sección se relaciona con el modelo de las figuras 3.3 y 3.4.

4.2. DEFINICIÓN DE LOS ESCENARIOS Y CASOS DE PRUEBA

Los casos de prueba se definen teniendo en cuenta los siguientes aspectos: escenarios de prueba, arquitectura y punto de interés.

4.2.1. ESCENARIOS DE PRUEBA

Los prototipos se ejecutan en dos ambientes de prueba, los cuales se diferencian de acuerdo a la conectividad que utiliza el dispositivo móvil para acceder a la información del POI.

En ambos ambientes participan componentes comunes como lo son el dispositivo móvil, el servidor y el access point (AP). Las características de estos dispositivos se describen en la tabla 4.1:

Tabla 4.1 – Dispositivos/Componentes comunes en los escenarios de pruebas.

Id	Dispositivo Componentes	Características	Ilustración
1	**Servidor Web (PC)** *Computadora que posee alojado el software del servidor Web y el sistema de gestión de base de datos.*	▪ Características de Hardware ▪ Procesador: Intel Core i5-2300 CPU. ▪ Motherboard: Intel. ▪ Memoria RAM: 4 GB. ▪ Adaptador USB Inalámbrico N a 300Mbps TL-WN821N ▪ Características de S.O ▪ Windows 2008 Server. ▪ Aplicaciones. ▪ IIS v7.0. ▪ SQL Server Professional 2008.	
2	**Access Point (AP)** *TL-MR3420 3G / 3.75G Router inalámbrico*	▪ Estándares Inalámbricos: IEEE 802.11n, IEEE 802.11g, IEEE 802.11b. ▪ Frecuencia: 2.4-2.4835GHz ▪ Tipo de Antena: Omnidireccional, desmontable, SMA Reversa. ▪ Rendimiento de la Antena: 2x3dBi. ▪ Velocidad inalámbrica N de hasta 300Mbps. ▪ Compatible con módems UMTS / HSPA / EVDO USB 3G/3.75G. ▪ Compatible con PPPoE, IP dinámica, IP estática, PPTP, L2TP Acceso a Internet.	

Tabla 4.1 – Dispositivos/Componentes comunes en los escenarios de prueba (continuación).

Id	Dispositivo Componentes	Características	Ilustración
3	**Dispositivo Móvil** *Samsung GALAXY S II*	▪ Red: HSPA+ 21Mbps/ HSUPA 5.76Mbps - EDGE/ GPRS Class 12 - Quad band GSM 850/900/1800/1900 -Quad band UMTS 850/900/1900/2100 ▪ CPU: Dual Core Application Processor ▪ Batería: 1650mAh ▪ OS: AndroidPlatform2.3/4.0.3 ▪ Dimensiones: 125.3X66.1X8.49mm. ▪ Pantalla: 4.3" WVGA SUPER AMOLED Plus * ▪ Memoria: 16GB/32GB - MicroSD (up to 32GB) ▪ Cámara: 8MP AF with LED Flash + 2MP Front ▪ Conectividad: Wi-Fi a/b/g/n - BT v3.0+HS - USB v2.0 ▪ GPS: A-GPS	
4	**Dispositivo Móvil** *HTC Explorer*	▪ Red: GSM 850 / 900 / 1800 / 1900 - HSDPA 900 / 2100 ▪ CPU: Qualcomm MSM7227 600MHz, GPU Adreno 200. ▪ Batería: Standard, Li-Ion 1230 mAh. ▪ OS: Android OS, v2.3 Gingerbread. ▪ Dimensiones: 102.8 x 57.2 x 12.9 mm. ▪ Pantalla: 320 x 480 pixels, 3.2 pulgadas. ▪ Memoria: 512MB ROM, 384MB RAM ▪ Cámara: 3.15 MP, 2048x1536 pixels. ▪ Conectividad: EDGE - 3G HSDPA 14.4Mbps / HSUPA 5.76Mbps - Wi-Fi 802.11 b/g/n - Bluetooth v3.0 A2DP, EDRGPS: A-GPS	

Tabla 4.1 – Dispositivos/Componentes comunes en los escenarios de prueba (continuación).

Id	Dispositivo Componentes	Características	Ilustración
5	**Antena de Telefonía Móvil 3G/GPRS:** *Empresa Movistar*	▪ Cobertura 3G y GPRS en la ubicación donde se realizará las pruebas. Ubicación Av. Rivadavia 476. Para ello se utilizara un chip de la empresa Movistar con plan de datos ilimitado.	

A partir de los elementos descriptos, se diseñaron dos ambientes de prueba que poseen diferentes tipos de conectividad: Wi-Fi y 3G/GPRS. Se consideran estos dos principalmente debido a que influyen en forma diferenciada en el consumo de energía y en el tiempo de respuesta.

4.2.1.1. AMBIENTE 1 "WI-FI"

El *ambiente 1* es un escenario de prueba "local-controlado" donde se conecta a través de la red Wi-Fi Clase N el dispositivo móvil y el servidor (ver figura 4.6). El dispositivo lee el código QR y se conecta al servidor a través de la red LAN (Wi-Fi), el servidor procesa la petición y responde al dispositivo móvil a través del mismo medio de comunicación. Como se puede visualizar los elementos principales que se identifican en este ambiente son tres: dispositivo móvil, punto de acceso y el servidor web.

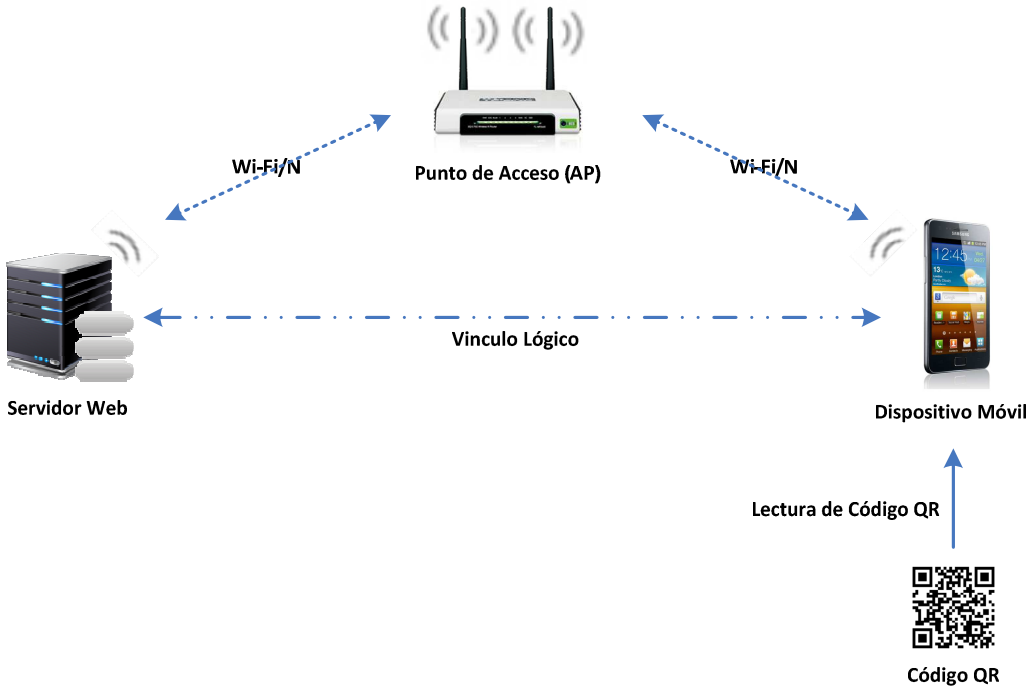


Figura 4.6 - Ambiente 1 "Wi-Fi".

De esta manera, las pruebas, se abstraerán de los inconvenientes de congestionamiento de la red por problemas de "cuellos de botellas" en horarios picos de consumo de la red de telefonía móvil y la falta de cobertura 3G/GPRS (Móvil).

4.2.1.2. AMBIENTE 2 "MÓVIL"

El *ambiente 2* es un escenario de prueba más complejo donde no sólo se compara la performance de las arquitecturas entre ellas sino que, además, se puede obtener datos para poder estimar el rendimiento de las mismas en un escenario real. El vínculo lógico del ambiente 2 es similar a la del ambiente 1, pero a los elementos notables de esta última se le añaden las empresas que proveen la telefonía móvil e Internet, como se puede apreciar en la figura 4.7.

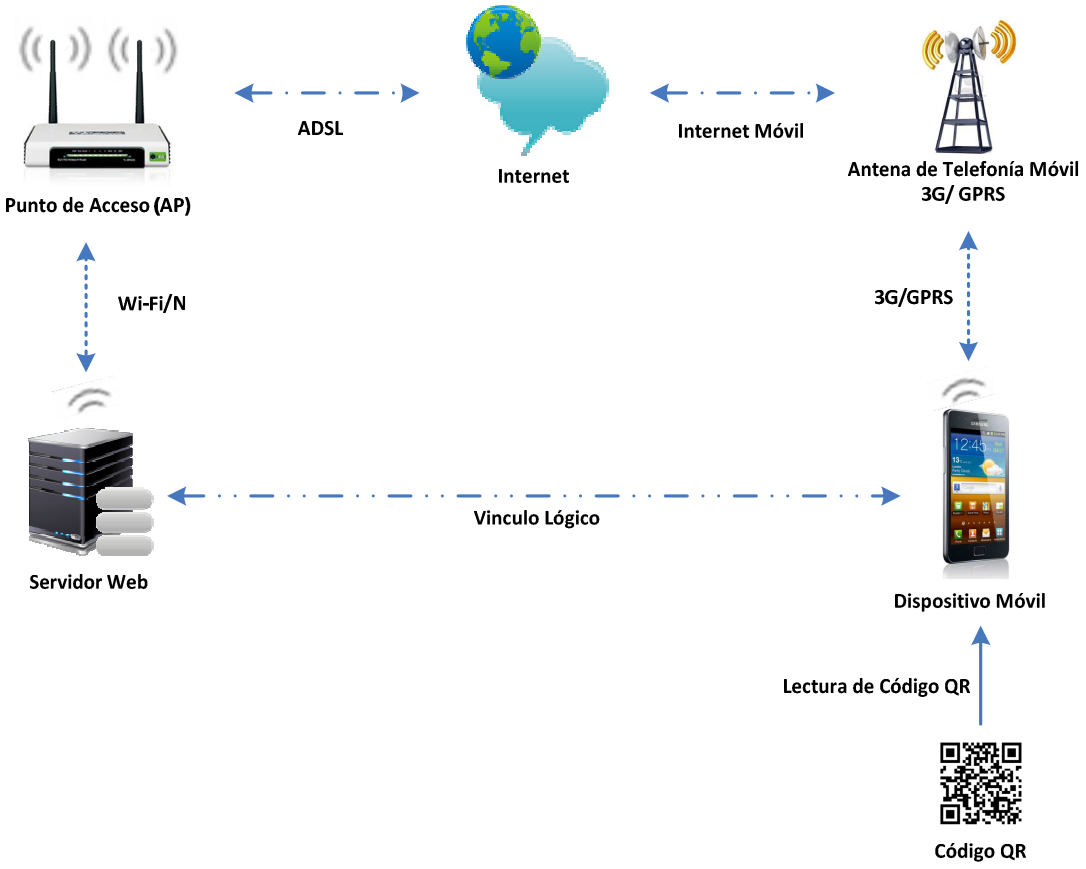


Figura 4.7 - Ambiente 2 "Móvil".

Se espera encontrar inconvenientes de congestionamiento de la red por problemas de "cuellos de botellas" en horarios picos de consumo de la red de telefonía móvil y la falta de cobertura 3G/GPRS (Móvil).

4.2.2. EN RELACIÓN A LA ARQUITECTURA

En relación a la **arquitectura**, se ejecutan los dos prototipos, con arquitectura híbrida y web, en forma alternada. Con ello se puede comparar la performance de ambas arquitecturas a pasar de no realizar las pruebas en paralelo, procedimiento innecesario en el marco de este proyecto.

4.2.3. EN RELACIÓN A LOS PUNTOS DE INTERÉS

En relación a los **puntos de interés**, se define un PI que corresponde a un lugar de Termas de Río Hondo que tiene relevancia turística: Isla del Sol o *Tara Inti*. En general, los PI tienen la siguiente estructura:

- Título
- Descripción
- Organizador por
- Galería de imágenes (0-3-5 imágenes)

Respecto al PI *Tara Inti*, se diseñaron tres casos de prueba, los cuales varían en el tipo y cantidad de información:

- PI-Caso.1: Corresponde a la descripción de *Tara Inti* usando solamente texto.
- PI-Caso.2: Corresponde a la descripción de *Tara Inti* usando texto y 3 imágenes. Las imágenes son archivos en formato Joint Photographic Experts Group (.jpg). El tamaño total de las 3 imágenes es: 76,7KB.
- PI-Caso.3: Corresponde a la descripción de *Tara Inti* usando texto y 5 imágenes. Las imágenes son archivos en formato Joint Photographic Experts Group (.jpg). El tamaño total de las 5 imágenes es: 479 KB.

Las etiquetas QR que identifican a cada caso de prueba se pueden apreciar a continuación en la figura 4.8.

PI-Caso.1

PI-Caso.2

PI-Caso.3

Figura 4.8 - Etiquetas QR de cada PI.

4.3. EVALUACIÓN DE LA CALIDAD DE LOS PROTOTIPOS

Como se mencionó al inicio del presente capítulo, la evaluación de la calidad se realiza siguiendo las actividades propuestas en el método GOCAME (Lew, Olsina, Becker, & Zhang, 2011):

a. Definición de los requisitos no funcionales o NFR
b. Diseño de las mediciones
c. Diseño de la evaluación
d. Implementación de las mediciones
e. Implementación de la evaluación

A continuación se detallan las actividades *a*, *b* y *c*, presentándose los productos obtenidos. Luego en el siguiente capítulo, se presentan los resultados de las mediciones y de la evaluación (actividades *d* y *e*).

4.3.1. DEFINICIÓN DE REQUISITOS NO FUNCIONALES

Para la definición de los requerimientos no-funcionales se tuvo en cuenta el dominio de la aplicación, es decir, el Turismo y las necesidades de los usuarios que consumen este tipo de aplicación.

El propósito fundamental consiste en medir la *eficiencia* de las *arquitecturas* en base a dos variables principales: el tiempo y los recursos (variables críticas en el dominio seleccionado). En este proyecto, no se medirá *throughput* directamente, ya que no es necesario definir el tráfico de información en función del tiempo. Es necesario comparar la eficiencia en virtud del tráfico generado por las arquitecturas para una misma función; no se analiza el comportamiento de las mismas en relación con el canal de comunicación.

Se prevé que una arquitectura que use una mayor cantidad de datos en una transacción utilizará más recursos del dispositivo impactando negativamente en la eficiencia de la aplicación. Sin embargo, es necesario determinar el grado de consumo de energía de cada arquitectura en una tarea común en las aplicaciones de m-turismo (por ejemplo, la consulta a un PI).

Siguiendo lo establecido en (Lew, Olsina, Becker, & Zhang, 2011), en base a los requisitos se define el Árbol de Requerimientos. Se parte de la *característica* de calidad que se desea medir o evaluar, en este caso, la *eficiencia*. A partir de la eficiencia se definen *sub-características*, en este caso *Comportamiento en el tiempo* y *Utilización de recursos*. Tanto las características como las subcaracterísticas no pueden medirse en forma directa, es necesario definir para ellas *atributos* que sean medibles. Es así como se definen tres atributos para *Comportamiento en el tiempo* y cuatro atributos para *Utilización de los recursos*. En la tabla 4.2 se muestra el *Árbol de Requerimientos* elaborado.

Tabla 4.2. Árbol de Requerimientos de *Eficiencia*.

1. Eficiencia	
1.1 Comportamiento en el tiempo	
1.1.1	Tiempo de respuesta en la consulta a un PI
1.1.2	Bytes Recibidos por la Aplicación Móvil en la consulta a un PI
1.1.3	Bytes Enviados por la Aplicación Móvil en la consulta a un PI
1.2 Utilización de los recursos	
1.2.1	Consumo de energía del CPU
1.2.2	Consumo de energía de la Pantalla
1.2.3	Consumo de energía de la interfaz 3G
1.2.4	Consumo de energía de la interfaz Wi-Fi

En las tablas 4.3 a 4.9 se presentan los atributos definidos en el árbol de requerimientos (hojas del árbol), utilizando la Ficha de Especificación de Atributos propuesta por el grupo de investigación del Dr. Olsina (Lew, Olsina, Becker, & Zhang, 2011).

Tabla 4.3. Ficha de Especificación del atributo *Tiempo de respuesta en la consulta a un PI*.

Ficha de Especificación de Atributos	**Cód.: 1.1.1**
Característica	Eficiencia
Subcaracteristica	Comportamiento en el tiempo
Categoría	Aplicaciones Móviles
Entidad	Aplicación de m-Turismo
Nombre	Tiempo de respuesta en la consulta a un PI
Definición	Cantidad total del tiempo en la consulta a un PI
Objetivo	Brindar el tiempo total insumido desde que se realiza la consulta del PI hasta que se visualiza por completo la información del mismo en la pantalla del dispositivo

Tabla 4.4. Ficha de Especificación del atributo *Bytes recibidos por la Aplicación Móvil en la consulta a PI*.

Ficha de Especificación de Atributos	**Cód.: 1.1.2**
Característica	Eficiencia
Subcaracteristica	Comportamiento en el tiempo
Categoría	Aplicaciones Móviles
Entidad	Aplicación de m-Turismo
Nombre	Bytes recibidos por la Aplicación Móvil en la consulta a un PI
Definición	Cantidad total de Bytes recibidos por la Aplicación Móvil en la consulta de un PI
Objetivo	Brindar la cantidad total de Bytes recibidos por la aplicación en la consulta de un PI

Tabla 4.5. Ficha de Especificación del atributo *Bytes enviados por la Aplicación Móvil en la consulta a PI*.

Ficha de Especificación de Atributos	**Cód.: 1.1.3**
Característica	Eficiencia
Subcaracteristica	Comportamiento en el tiempo
Categoría	Aplicaciones Móviles
Entidad	Aplicación de m-Turismo
Nombre	Bytes enviados por la Aplicación Móvil en la consulta a un PI
Definición	Cantidad total de Bytes enviados por la Aplicación Móvil en la consulta de un PI
Objetivo	Brindar la cantidad total de Bytes enviados por la aplicación en la consulta de un PI

Tabla 4.6. Ficha de Especificación del atributo *Consumo de energía del CPU*.

Ficha de Especificación de Atributos	**Cód.: 1.2.1**
Característica	Eficiencia
Subcaracteristica	Utilización de los Recursos
Categoría	Aplicaciones Móviles
Entidad	Aplicación m-Turismo
Nombre	Consumo de energía del CPU
Definición	Cantidad total de consumo de energía del CPU
Objetivo	Brindar el consumo de energía total insumido por el CPU en la consulta del PI.

Tabla 4.7. Ficha de Especificación del atributo *Consumo de energía de la pantalla*.

Ficha de Especificación de Atributos	**Cód.: 1.2.2**
Característica	Eficiencia
Subcaracteristica	Utilización de los Recursos
Entidad	Aplicación Móvil
Nombre	Consumo de energía de la Pantalla
Definición	Cantidad total de consumo de energía de la Pantalla
Objetivo	Brindar el consumo de energía total insumido por la Pantalla del dispositivo en la consulta del PI.

Tabla 4.8. Ficha de Especificación del atributo *Consumo de energía de la Interfaz 3G*.

Ficha de Especificación de Atributos	**Cód.: 1.2.3**
Característica	Eficiencia
Subcaracteristica	Utilización de los Recursos
Categoría	Aplicaciones Móviles
Entidad	Aplicación m-Turismo
Nombre	Consumo de energía de la Interfaz 3G
Definición	Cantidad total de consumo de energía de la Interfaz 3G
Objetivo	Brindar el consumo de energía total insumido por la Interfaz 3G del dispositivo en la consulta del PI.

Tabla 4.9. Ficha de Especificación del atributo *Consumo de energía de la Interfaz Wi-Fi*.

Ficha de Especificación de Atributos	**Cód.: 1.2.4**
Característica	Eficiencia
Subcaracteristica	Utilización de los Recursos
Categoría	Aplicaciones Móviles
Entidad	Aplicación m-Turismo
Nombre	Consumo de energía de la Interfaz WI-FI
Definición	Cantidad total de consumo de energía de la Interfaz WI-FI
Objetivo	Brindar el consumo de energía total insumido por la Interfaz WI-FI del dispositivo en la consulta del POI.

4.3.2. DISEÑO DE LAS MEDICIONES

En este apartado se definen las métricas que permiten obtener los valores que cuantifican a los atributos de eficiencia de la entidad *Aplicación de m-Turismo.*

Es decir, una vez que se ejecutan los prototipos, se miden los siete atributos del Árbol de Requisitos No Funcionales. Esas mediciones se hacen siguiendo lo especificado en las métricas. Hay una métrica para cada atributo. Cada métrica indica cuál es el método que se sigue para obtener la medición, qué unidad corresponde, qué herramienta se utiliza. Asimismo, una métrica puede ser directa o indirecta. Una métrica es indirecta cuando utiliza otras métricas predefinidas para llevar a cabo la medición.

De acuerdo a lo establecido en el método GOCAME, el diseño de las mediciones se lleva a cabo considerando: la definición de requisitos no funcionales (el árbol) y la base de datos de métricas conocidas; en el caso que no haya métricas conocidas se debe diseñar una nueva, como ocurre en estos casos.

Las métricas definidas se muestran utilizando las fichas de *Especificaciones de Métricas* propuestas por el grupo de investigación del Dr. Olsina (Lew, Olsina, Becker, & Zhang, 2011). En las tablas 4.10 a 4.16 se muestran las fichas correspondientes a cada uno de los siete atributos de la eficiencia.

En general, las métricas definidas implican: la medición de tiempo de respuesta en segundos usando cronómetros, la cantidad de información enviada/recibida en Kilobytes medidos con Traffic Monitor y el consumo de batería medidos con PowerTutor en Joules.

Tabla 4.10. Ficha de Especificación de la métrica *Tiempo de respuesta en la consulta de un PI (TR)*.

Métrica		**Tipo**	**Directa**
Nombre	*Tiempo de respuesta en la consulta de un PI (TR)*		
Objetivo	Determinar la cantidad de tiempo en segundos insumido en una consulta a un PI		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	Cronómetro digital		
Método de Medición			
Nombre	Cálculo de tiempo		
Especificación	El cronometraje dará inicio en el momento en el que se procesa el código QR del PI obtenido en la lectura a través de la cámara del dispositivo móvil con el aplicativo Barcode Scanner. Finalizará cuando se visualice, en su totalidad, la información del PI en la pantalla del dispositivo.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	Segundos		
Acrónimo	s		
Observaciones			
El Barcode scanner, en su versión para Android, emite un sonido (beep) al obtener el dato codificado en la etiqueta QR, útil para comenzar con el conteo del tiempo.			

Tabla 4.11. Ficha de Especificación de la métrica *Total de Bytes recibidos por la Aplicación Móvil (TKBR)*.

Métrica		**Tipo**	**Directa**
Nombre	*Total de Bytes recibidos por la Aplicación Móvil (TKBR)*		
Objetivo	Determinar la cantidad total de Bytes recibidos en la consulta de un PI		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	Traffic Monitor		
Método de Medición			
Nombre	Cálculo de cantidad total de bytes recibidos		
Especificación	Para dar inicio al cálculo se ejecutará la herramienta Traffic Monitor, luego inicializar los acumuladores de la herramienta y comenzar con las consultas del PI. La herramienta determina la cantidad de bytes recibidos por aplicación en la sección: Consumo. Dicho valor puede ser consultado a través de 2 Interfaces: 3G (Mobile) y WI-FI, de acuerdo al método de conexión de la aplicación al Servidor Web.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	kilobyte		
Acrónimo	KB		
Observaciones: -			

Tabla 4.12. Ficha de Especificación de la métrica *Total de Bytes enviados por la Aplicación Móvil (TKBE)*.

Métrica		**Tipo**	**Directa**
Nombre	*Total de Bytes enviados por la Aplicación Móvil (TKBE)*		
Objetivo	Determinar la cantidad total de Bytes enviados en la consulta de un PI		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	Traffic Monitor		
Método de Medición			
Nombre	Cálculo de cantidad total de bytes enviados		
Especificación	Para dar inicio al cálculo se ejecutará la herramienta Traffic Monitor, luego inicializar los acumuladores de la herramienta y comenzar con las consultas del PI. La herramienta determina la cantidad de bytes enviados por aplicación en la sección: Consumo. Dicho valor puede ser consultado a través de 2 Interfaces: 3G (Mobile) y WI-FI de acuerdo al método de conexión de la aplicación al Servidor Web.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	kilobyte		
Acrónimo	KB		
Observaciones: -			

Tabla 4.13. Ficha de Especificación de la métrica *Consumo de energía del CPU (TECPU)*.

Métrica		**Tipo**	**Directa**
Nombre	*Consumo de energía del CPU (TECPU)*		
Objetivo	Determinar el consumo de energía en Joule insumido en una consulta a un PI por el CPU del dispositivo		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	PowerTutor		
Método de Medición			
Nombre	Cálculo de consumo de energía en PowerTutor		
Especificación	Para dar inicio al cálculo se ejecutará la herramienta PowerTutor y luego iniciar el registro a presionando el botón "Start Profile". Realizar la consulta al PI y una vez finalizada dicha tarea se debe detener registro con el botón "Stop Profile". Guardar el registro (Archivo .log). El consumo se calcula, a partir del archivo generado, mediante la suma de los valores generados de CPU por la aplicación, teniendo en cuenta el id-proceso asociado a la misma.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	Joule		
Acrónimo	J		
Observaciones: -			

Tabla 4.14. Ficha de Especificación de la métrica *Consumo de energía de la Pantalla (TEP).*

Métrica		**Tipo**	**Directa**
Nombre	*Consumo de energía de la Pantalla (TEP)*		
Objetivo	Determinar el consumo de energía en Joule insumido en una consulta a un PI por la pantalla del dispositivo		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	PowerTutor		
Método de Medición			
Nombre	Calculo de consumo de energía en PowerTutor		
Especificación	Para dar inicio al cálculo se ejecutará la herramienta PowerTutor y luego iniciar el registro a presionando el botón “Start Profile”. Realizar la consulta al PI y una vez finalizada dicha tarea se debe detener registro con el botón “Stop Profile”. Guardar el registro (Archivo .log). El consumo se calcula, a partir del archivo generado, mediante la suma de los valores generados de LCD por la aplicación, teniendo en cuenta el id-proceso asociado a la misma.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	Joule		
Acrónimo	J		
Observaciones: -			

Tabla 4.15. Ficha de Especificación de la métrica *Consumo de energía de la Interfaz 3G (TE3G).*

Métrica		**Tipo**	**Directa**
Nombre	*Consumo de energía de la Interface 3G (TE3G)*		
Objetivo	Determinar el consumo de energía en Joule insumido en una consulta a un PI por la interfaz 3G del dispositivo		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	PowerTutor		
Método de Medición			
Nombre	Cálculo de consumo de energía en PowerTutor		
Especificación	Para dar inicio al cálculo se ejecutará la herramienta PowerTutor y luego iniciar el registro a presionando el botón "Start Profile". Realizar la consulta al PI y una vez finalizada dicha tarea se debe detener registro con el botón "Stop Profile". Guardar el registro (Archivo .log). El consumo se calcula, a partir del archivo generado, mediante la suma de los valores generados de 3G por la aplicación, teniendo en cuenta el id-proceso asociado a la misma.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	Joule		
Acrónimo	J		
Observaciones			
Este cálculo en la herramienta PowerTutor no es posible para algunos Smartphone. Ej.: Samsung SII. Es recomendable desactivar todas las aplicaciones de terceros que no estén involucradas en el proceso de medición.			

Tabla 4.16. Ficha de Especificación de la métrica *Consumo de energía de la Interfaz WI-FI (TEW).*

Métrica		**Tipo**	**Directa**
Nombre	*Consumo de energía de la Interfaz WI-FI (TEW)*		
Objetivo	Determinar el consumo de energía en joule insumido en una consulta a un PI por la interface WI-FI del dispositivo		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	PowerTutor		
Método de Medición			
Nombre	Cálculo de consumo de energía en PowerTutor		
Especificación	Para dar inicio al cálculo se ejecutará la herramienta PowerTutor y luego iniciar el registro a presionando el botón "Start Profile". Realizar la consulta al PI y una vez finalizada dicha tarea se debe detener registro con el botón "Stop Profile". Guardar el registro (Archivo .log). El consumo se calcula, a partir del archivo generado, mediante la suma de los valores generados de Wifi por la aplicación, teniendo en cuenta el id-proceso asociado a la misma.		
Tipo	**-**		
Escala Numérica			
Representación	-		
Tipo de Valor	Real		
Escala	Absoluta		
Unidad			
Nombre	Joule		
Acrónimo	J		
Observaciones			
Este cálculo en la herramienta PowerTutor no es posible para algunos Smartphone. Ej.: Samsung SII. Es recomendable desactivar todas las aplicaciones de terceros que no estén involucradas en el proceso de medición.			

4.3.3. DISEÑO DE LA EVALUACIÓN

Para obtener el grado de satisfacción de los requisitos no funcionales es necesario definir un indicador que permita interpretar cada atributo medido. Y, posteriormente, otro indicador que, a partir de los indicadores de los atributos, permita evaluar las subcaracterísticas de la eficiencia y finalmente la eficiencia como característica de la calidad. Los indicadores de los atributos se denominan indicadores elementales, mientras que los indicadores que evalúan las características o subcaracterísticas, se denominan indicadores globales.

Como se vio hasta el momento, a partir de la aplicación de una métrica sobre un atributo, se obtiene un concepto calculable o medición (en segundos, Joules, Kilobytes, etc.). Estas mediciones no permiten definir en forma directa si se satisfacen o no los requisitos. Es necesario definir, entonces, un indicador que permita evaluar si un atributo medido es o no satisfactorio. El indicador de cada atributo se denomina **Indicador Elemental**.

Según el método GOCAME, este diseño de la evaluación se realiza a partir de: la especificación de métricas (las fichas) y la base de datos de indicadores ya conocidos. En este caso no existen indicadores conocidos, por lo tanto, se diseñan indicadores nuevos. Esta etapa tiene, como entrada, las fichas de especificación de atributos y métricas definidas en el apartado anterior y, como salida, las Fichas de Especificación de Indicadores Elementales que se presentan en las tablas 4.17 a 4.23.

Posteriormente, se define el indicador global de cada subcaracterística *Comportamiento del tiempo* y *Utilización de los recursos*. Finalmente, se define el indicador global de la característica *Eficiencia*. Estos se describen en las Fichas de Especificación de Indicadores Globales que se presentan en las tablas 4.24 a 4.26. A continuación se describe la fórmula general seguida para obtener los indicadores elementales y los indicadores globales.

Dado que las mediciones de los atributos arrojan siempre valores numéricos, es factible mapearlos a un rango de números reales del 0 al 10 usando una función de proporción, función lineal. Es por ello que los siete valores obtenidos usando las métricas se interpretan mediante un indicador elemental que, en todos los casos, utiliza la misma función.

Los valores obtenidos en la medición son valores reales, con diferentes unidades, que siempre se encuentran en un rango determinado. Ese rango de la medición se toma en función de mediciones previas registradas en diversos horarios (horarios pico y no pico) y con los 3 casos de prueba (POI sin imágenes, POI con 3 imágenes y POI con 5 imágenes). Por ejemplo, el consumo de energía de la CPU cualquiera sea el caso, oscila entre los 0,5 J y 5 J; por lo tanto estos definen el rango de valores reales obtenidos en las mediciones. Entonces, cada valor que se obtiene se mapea a un indicador elemental que puede tomar valores reales entre 0 y 10; donde a 0,5 corresponde 0 y a 5 corresponde 10.

El razonamiento seguido es el siguiente: una escalamiento es una función uno a uno en la que hay una constante de proporcionalidad, una función del tipo *y = kx*, donde *k* es la constante de proporcionalidad, luego es una función lineal. El problema no se reduce sólo a encontrar la constante y multiplicar ya que los valores dados no parten del 0, luego hay que sumar o restar según sea el rango de valores de las mediciones.

Siguiendo el ejemplo previo, sean los valores en el rango dado *x ϵ [0,5; 5]* e *y ϵ [0, 10]*, luego el cambio de escala es una función lineal del tipo que se muestra en el gráfico de la figura 4.9.

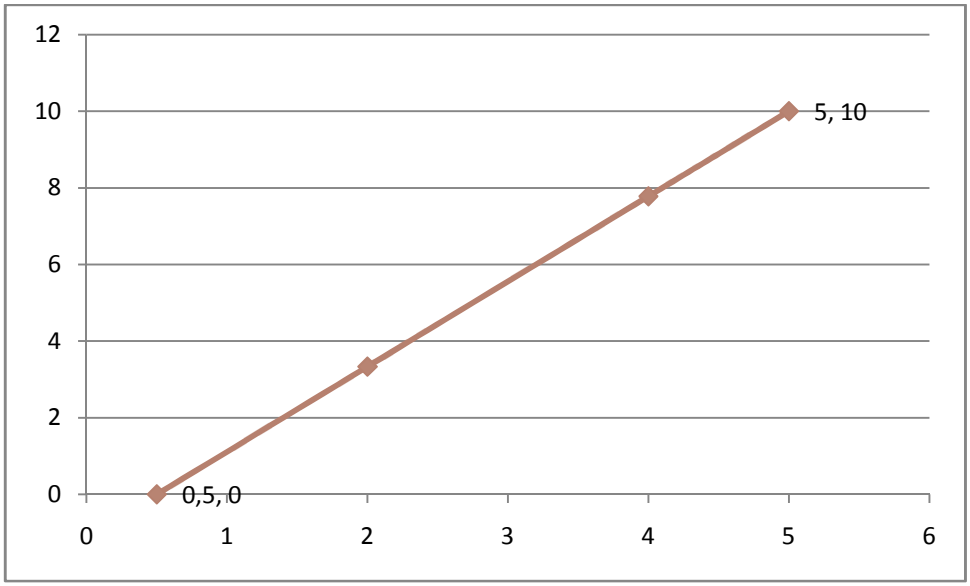


Figura 4.9 - Ficha de Especificación de la métrica *Consumo de energía del CPU (TECPU).*

Para conocer los valores, se puede aplicar la ecuación de la recta que pasa por dos puntos:

$$y - y_0 = \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)$$

En el ejemplo, los puntos son (0,8; 0) y (5, 10). Donde $x_0 = 0,5$, $y_0 = 0$, $x_1 = 5$, $y_1 = 10$.

Reemplazando:

$$y = \frac{10}{4,5}(x - 0,5)$$

Siguiendo esta fórmula, en el caso del consumo de energía de CPU, si el valor obtenido en la métrica es *x = 2,10 J*, el indicador tomará el valor *y = 3,1*.

Generalizando, para obtener los 7 indicadores elementales, considerando que el mapeo es siempre a valores reales en el rango [0,10], se utilizará la siguiente fórmula:

$$y = \frac{10}{v_q - v_i}(x - v_i)$$

Donde v_i **es el límite inferior** del rango de valores posibles de la medición de un determinado atributo. Y v_q **es el límite superior**. En el ejemplo seguido, $v_i = 0,5$ y $v_q = 5$.

Los indicadores elementales se aplican a las mediciones del caso intermedio, es decir, al POI que devuelve texto y 3 imágenes. Esto se debe a que es el caso típico de una consulta de aplicaciones de turismo: siempre involucran texto e imágenes y/o audiovisuales. No se considera el caso de 5 imágenes de alta resolución debido a que, en el caso de las arquitecturas Web, usan navegadores que optimizan la bajada de archivos grandes, reduciendo el tamaño de la imagen; esto puede hacer que desvirtúe el tiempo real que se usaría en una transacción normal.

Una vez explicado el procedimiento general seguido para la definición de indicadores elementales, a continuación se presenta cada uno de ellos en forma detallada.

Para evaluar el atributo *Tiempo de respuesta en la consulta a un POI* se diseñó el indicador elemental ***Nivel de desempeño del tiempo de respuesta (NDTR).*** Para el cual se aplica la función mencionada previamente, donde $v_i = 2$ y $v_q = 18$.

Además, se definen los valores críticos que permitirán tomar decisiones. En este caso, cuando el indicador elemental NDTR es mayor a 8, se debe (obligatorio) revisar la arquitectura y la infraestructura de red utilizada, se lo representa con color rojo. Cuando el indicador NDTR se encuentra en el rango mayor que 2 y menor o igual a 8, el sistema es aceptable, se lo representa con color amarillo. Cuando el indicador NDTR es menor o igual a 2, el sistema funciona en condiciones óptimas y no necesita revisión; se lo representa con color verde. En la Tabla 4.17 se presenta la Ficha con la información detallada de este indicador.

Tabla 4.17. Ficha de Especificación del indicador elemental NDTR.

Indicador Elemental		**Cód.: 1.1.1**	
Nombre	**Nivel de desempeño del tiempo de respuesta (NDTR)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	$$NDTR = \frac{10}{v_q - v_i}(x - v_i)$$		
Especificación	El valor de la medición del atributo Tiempo de Respuesta en la consulta de un PI (considerado como *x*), en segundos, se lo mapea en forma directa a un rango real [0-10]. $v_i = 2$ y $v_q = 18$ El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. A partir del valor que toma el indicador, se toman las siguientes decisiones: 0 <= NDTR <= 2 => el sistema funciona en condiciones óptimas y no necesita revisión; 2 < NDTR <= 8 => el sistema es aceptable; 8 < NDTR <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	8 < NDTR <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Marginal		
Descripción	El sistema es aceptable.		
Color	Amarillo		
Rango	2 < NDTR <= 8		
Criterios de Decisión		**N°**	**3**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NDTR <= 2		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** $y \in R \ \Lambda \ y \in [0, 10]$.		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Para evaluar el atributo *Bytes recibidos por la aplicación móvil en la consulta a un PI* se diseñó el indicador elemental *Nivel de desempeño de Bytes Recibidos (NDBR)*. Para el cual se aplica la función mencionada previamente, donde $v_i = 1$ y $v_q = 800$.

Además, se definen los valores críticos que permitirán tomar decisiones. En este caso, cuando el NDBR es igual a 0, significa que no se ha llevado a cabo la transacción ya que no se recibieron datos como respuesta a la consulta al PI. También cuando NDBR es mayor a 8 y menos o igual a 10, significa que se han recibido más datos que los esperados. En ambos casos, se debe (obligatorio) revisar la arquitectura y la infraestructura de red utilizada. Cuando NDBR se encuentra en el rango mayor que 0 y menor o igual a 8, el sistema funciona correctamente y no necesita revisión. En la tabla 4.18 se presenta la Ficha con la información detallada de este indicador.

Para evaluar el atributo *Bytes enviados por la aplicación móvil en la consulta a un PI* se diseñó el indicador elemental *Nivel de desempeño de Bytes Enviados (NDBE)*. Para el cual se aplica la función mencionada previamente, donde $v_i = 1$ y $v_q = 4$.

Además, se definen los valores críticos que permitirán tomar decisiones. En este caso, cuando el NDBE es igual a 0, significa que no se ha llevado a cabo la transacción ya que no se enviaron datos en la consulta al PI. También cuando NDBE es mayor a 2 y menor o igual a 10, significa que se han enviado más datos que los esperados, es decir, que la consulta armó un paquete de datos de tamaño mayor que el esperado. En ambos casos, se debe (obligatorio) revisar la arquitectura y/o la infraestructura de red utilizada. Cuando NDBE se encuentra en el rango mayor que 0 y menor o igual a 2, el sistema funciona correctamente y no necesita revisión. En la tabla 4.19 se presenta la Ficha con la información detallada de este indicador.

Para evaluar el atributo *Consumo de Energía del CPU* se diseñó el indicador elemental *Nivel de desempeño de Consumo de Energía del CPU (NDECPU)*. Para el cual se aplica la función mencionada previamente, donde $v_i = 0,5$ y $v_q = 5$.

Además, se definen los valores críticos que permitirán tomar decisiones. En este caso, cuando el indicador elemental NDECPU es mayor a 7, se debe (obligatorio) revisar la arquitectura y/o la infraestructura de red utilizada, se lo representa con color rojo. Cuando el indicador NDECPU se encuentra en el rango mayor que 2 y menor o igual a 7, el sistema es aceptable, se lo representa con color amarillo. Cuando el indicador NDECPU es menor o igual a 2, el sistema funciona en condiciones óptimas y no necesita revisión; se lo representa con color verde. En la tabla 4.20 se presenta la Ficha con la información detallada de este indicador.

Tabla 4.18. Ficha de Especificación del indicador elemental NDBR.

Indicador Elemental		**Cód.: 1.1.2**	
Nombre	**Nivel de desempeño de Bytes Recibidos (NDBR)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	$$NDBR = \frac{10}{v_q - v_i}(x - v_i)$$		
Especificación	El valor de la medición del atributo Cantidad de Bytes recibidos por la aplicación en la consulta de un PI (considerado como *x*), en KB, se lo mapea en forma directa a un rango real [0-10]. $v_i = 1$ y $v_q = 800$ El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. A partir del valor que toma el indicador, se toman las siguientes decisiones: (0 = NDBR) ó (8 < NDBR <= 10) => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación. 0 < NDBR <= 8 => el sistema funciona correctamente y no necesita revisión.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	(0 = NDBR) ó (8 < NDBR <= 10)		
Criterios de Decisión		**N°**	**2**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 < NDBR <= 8		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** NDBR ϵ *R* $\wedge$ NDBR ϵ *[0, 10].*		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Tabla 4.19. Ficha de Especificación del indicador elemental NDBE.

Indicador Elemental		**Cód.: 1.1.3**	
Nombre	**Nivel de desempeño de Bytes Enviados (NDBE)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	$$NDBE = \frac{10}{v_q - v_i}(x - v_i)$$		
Especificación	El valor de la medición del atributo Cantidad de Bytes enviados en la consulta a un POI (considerado como *x*), en KB, se lo mapea en forma directa a un rango real [0-10]. $v_i = 1$ y $v_q = 4$. El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. A partir del valor que toma el indicador, se toman las siguientes decisiones: (0 = NDBE) ó (2 < NDBE <= 10) => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación. 0 < NDBE <= 2 => el sistema funciona correctamente y no necesita revisión.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	(0 = NDBE) ∧ (2 < NDBE <= 10)		
Criterios de Decisión		**N°**	**2**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 < NDBE <= 8		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** NDBE ϵ *R* ∧ NDBE ϵ *[0, 10].*		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Tabla 4.20. Ficha de Especificación del indicador elemental NDECPU.

Indicador Elemental		**Cód.: 1.2.1**	
Nombre	**Nivel de desempeño de Consumo de Energía de la CPU (NDECPU)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	$$NDECPU = \frac{10}{v_q - v_i}(x - v_i)$$		
Especificación	El valor de la medición del atributo Consumo de Energía de CPU (considerado como *x*), en Joules (J), se lo mapea en forma directa a un rango real [0-10]. $v_i = 0,5$ y $v_q = 5$. El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. A partir del valor que toma el indicador, se toman las siguientes decisiones: 0 <= NDECPU <= 2 => el sistema funciona en condiciones óptimas y no necesita revisión; 2 < NDECPU <= 7 => el sistema es aceptable; 7 < NDECPU <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	7 < NDECPU <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Marginal		
Descripción	El sistema es aceptable.		
Color	Amarillo		
Rango	2 < NDECPU <= 7		
Criterios de Decisión		**N°**	**3**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NDECPU <= 2		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** $NDECPU \in R \ \wedge \ NDECPU \in [0, 10]$.		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Para evaluar el atributo *Consumo de Energía de la Pantalla* se diseñó el indicador elemental *Nivel de desempeño de Consumo de Energía de Pantalla (NDEP)*. Para el cual se aplica la función mencionada previamente, donde $v_i = 3$ y $v_q = 60$.

Además, se definen los valores críticos que permitirán tomar decisiones. En este caso, cuando el indicador elemental NDEP es mayor a 7, se debe (obligatorio) revisar la arquitectura y/o la infraestructura de red utilizada, se lo representa con color rojo. Cuando el indicador NDEP se encuentra en el rango mayor que 2 y menor o igual a 7, el sistema es aceptable, se lo representa con color amarillo. Cuando el indicador NDEP es menor o igual a 2, el sistema funciona en condiciones óptimas y no necesita revisión; se lo representa con color verde. En la Tabla 4.21 se presenta la Ficha con la información detallada de este indicador.

Para evaluar el atributo *Consumo de Energía de la Interfaz 3G* se diseñó el indicador elemental *Nivel de desempeño de Consumo de Energía de 3G (NDE3G)*. Para el cual se aplica la función mencionada previamente, donde $v_i = 0,05$ y $v_q = 60$.

Además, se definen los valores críticos que permitirán tomar decisiones. En este caso, cuando el indicador elemental NDE3G es mayor a 5, se debe (obligatorio) revisar la arquitectura y/o la infraestructura de red utilizada, se lo representa con color rojo. Cuando el indicador NDE3G se encuentra en el rango mayor que 1 y menor o igual a 5, el sistema es aceptable, se lo representa con color amarillo. Cuando el indicador NDE3G es menor o igual a 1, el sistema funciona en condiciones óptimas y no necesita revisión; se lo representa con color verde. En la Tabla 4.22 se presenta la Ficha con la información detallada de este indicador.

Para evaluar el atributo *Consumo de Energía de la Interfaz Wi-Fi* se diseñó el indicador elemental *Nivel de desempeño de Consumo de Energía de Wi-Fi (NDEWF)*. Para el cual se aplica la función mencionada previamente, donde $v_i = 0,05$ y $v_q = 60$. Se calcula de igual manera que el indicador de interfaz 3G, considerando los mismos rangos, con el propósito de que los valores sean comparables.

Además, se definen los valores críticos que permitirán tomar decisiones. Igual al caso anterior, cuando el indicador elemental NDEWF es mayor a 5, se debe (obligatorio) revisar la arquitectura y/o la infraestructura de red utilizada, se lo representa con color rojo. Cuando el indicador NDEWF se encuentra en el rango mayor que 1 y menor o igual a 5, el sistema es aceptable, se lo representa con color amarillo. Cuando el indicador NDEWF es menor o igual a 1, el sistema funciona en condiciones óptimas y no necesita revisión; se lo representa con color verde. En la Tabla 4.23 se presenta la Ficha con la información detallada de este indicador.

Tabla 4.21 - Ficha de Especificación del indicador elemental NDEP.

Indicador Elemental		**Cód.: 1.2.2**	
Nombre	**Nivel de desempeño de Consumo de Energía de Pantalla (**NDEP**)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	$$NDEP = \frac{10}{v_q - v_i}(x - v_i)$$		
Especificación	El valor de la medición del atributo Consumo de Energía de Pantalla (considerado como *x*), en Joules (J), se lo mapea en forma directa a un rango real [0-10]. $v_i = 3$ y $v_q = 60$. El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. A partir del valor que toma el indicador, se toman las siguientes decisiones: 0 <= NDEP <= 2 => el sistema funciona en condiciones óptimas y no necesita revisión; 2 < NDEP <= 7 => el sistema es aceptable; 7 < NDEP <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	7 < NDEP <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Marginal		
Descripción	El sistema es aceptable.		
Color	Amarillo		
Rango	2 < NDEP <= 7		
Criterios de Decisión		**N°**	**3**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NDEP <= 2		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** $NDEP \in R \ \wedge \ NDEP \in [0, 10]$.		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Tabla 4.22 - Ficha de Especificación del indicador elemental NDE3G.

Indicador Elemental		**Cód.: 1.2.3**
Nombre	**Nivel de desempeño de Consumo de Energía de 3G (NDE3G)**	
Autor	PNajar	
Versión	1.0	
Referencia	-	
Herramienta	-	
Modelo		
Función	$$NDE3G = \frac{10}{v_q - v_i}(x - v_i)$$	
Especificación	El valor de la medición del atributo Consumo de Energía de la Interfaz 3G (considerado como *x*), en Joules (J), se lo mapea en forma directa a un rango real [0-10]. $v_i = 0,05$ y $v_q = 60$. El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. Además, sólo se aplica cuando se utiliza las antenas de telefonía para la conexión a Internet, es decir 3G. No se aplica cuando la conexión a Internet es mediante Wi-Fi. A partir del valor que toma el indicador, se toman las siguientes decisiones: 0 <= NDE3G <= 1 => el sistema funciona en condiciones óptimas y no necesita revisión; 1 < NDE3G <= 5 => el sistema es aceptable; 5 < NDE3G <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.	
Criterios de Decisión		**N°** **1**
Nombre	Insatisfactorio	
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.	
Color	Rojo	
Rango	5 < NDE3G <= 10	
Criterios de Decisión		**N°** **2**
Nombre	Marginal	
Descripción	El sistema es aceptable.	
Color	Amarillo	
Rango	1 < NDE3G <= 5	
Criterios de Decisión		**N°** **3**
Nombre	Satisfactorio	
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.	
Color	Verde	
Rango	0 <= NDE3G <= 1	
Tipo de Escala		
Representación	**Numérica**	
Tipo de Valor	**Real,** $NDE3G \in R \wedge NDE3G \in [0, 10]$.	
Escala	-	
Unidad		
Nombre	-	
Acrónimo	-	

Tabla 4.23 - Ficha de Especificación del indicador elemental NDEWF.

Indicador Elemental		**Cód.: 1.2.4**	
Nombre	**Nivel de desempeño de Consumo de Energía de Wi-Fi (NDEWF)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	$$NDEWF = \frac{10}{v_q - v_i}(x - v_i)$$		
Especificación	El valor de la medición del atributo Consumo de Energía de la Interfaz Wi-Fi (considerado como *x*), en Joules (J), se lo mapea en forma directa a un rango real [0-10]. $v_i = 0,05$ y $v_q = 60$. El indicador se aplica sólo a la medición de la transacción intermedia: consulta a un POI con texto y 3 imágenes. Además, sólo se aplica cuando se utiliza la conexión a Internet mediante redes Wi-Fi. A partir del valor que toma el indicador, se toman las siguientes decisiones: 0 <= NDEWF <= 1 => el sistema funciona en condiciones óptimas y no necesita revisión; 1 < NDEWF <= 5 => el sistema es aceptable; 5 < NDEWF <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	5 < NDEWF <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Marginal		
Descripción	El sistema es aceptable.		
Color	Amarillo		
Rango	1 < NDEWF <= 5		
Criterios de Decisión		**N°**	**3**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NDEWF <= 1		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** $NDEWF \in R \wedge NDEWF \in [0, 10]$.		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Una vez definidos los 7 indicadores elementales, se diseñó el indicador global para las subcaracterísticas del Árbol de Atributos de la tabla 4.2. Es decir se definieron de: *Comportamiento en el tiempo* y de *Utilización de los recursos*.

Para ello, se optó por aplicar una fórmula de sumatoria de cada atributo de la subcaracterística multiplicada por su Peso. Es decir, se aplica la fórmula:

$$IG = \sum_{i=1}^{n} p_i * IE_i$$

Donde,

- IG es cada indicador global
- n es la cantidad de indicadores elementales de una subcaracterística
- p_i es el peso de cada atributo
- IE_i es cada indicador elemental

Se asignaron "pesos" a cada uno de los atributos de acuerdo a la relevancia que tiene cada uno de ellos en la optimización de la calidad del producto. La asignación de pesos es muy importante debido a que, en base a ellos se obtienen los indicadores globales. Los pesos se asignan de tal forma que sumados entre todos los que corresponden a una subcaracterística, el resultado sea igual a 1. En la tabla 4.24 se muestran los pesos otorgados a cada atributo.

Tabla 4.24 - Asignación de Pesos a los Atributos.

Subcaracterísticas de la EFICIENCIA	Atributos	Peso Atributos	Peso Subcaracteristica
Comportamiento en el tiempo	Tiempo de respuesta en la consulta a un PI	**0,25**	**0,45**
	Bytes Recibidos por la Aplicación Móvil en la consulta a un PI	**0,50**	
	Bytes Enviados por la Aplicación Móvil en la consulta a un PI	**0,25**	
Utilización de los recursos	Consumo de energía del CPU	**0,20**	**0,55**
	Consumo de energía de la Pantalla	**0,30**	
	Consumo de energía de la interfaz 3G	**0,50**	
	Consumo de energía de la interfaz Wi-Fi	**0,50**	

Para la subcaracterística **Comportamiento en el tiempo**, se da mayor relevancia a la cantidad de bytes recibidos debido a que es lo que más está impactado por la diferencia de arquitecturas. Se puede apreciar que el tamaño de los datos recibidos difiere para cada caso y va en aumento acorde al tamaño de la información descargada.

En contraste a esto, la cantidad de datos enviados generalmente para una misma arquitectura se comporta en igualdad de condiciones. Es por ello que la verdadera diferencia en el tamaño de los paquetes se debería apreciar en el paquete recibido en la consulta al servidor. Por todo ello, se le asigna mayor peso al atributo referido a esto último.

En cuanto al atributo Tiempo de respuesta en la consulta a un PI, generalmente es aleatorio porque depende en el caso de la interfaz 3G del tráfico de la red de telefonía en el momento de hacer la consulta, también depende de la tecnología de las antenas. Independientemente de la arquitectura, se espera que siempre la consulta que utiliza infraestructura Wi-Fi sea más rápida que 3G. Es un punto muy importante en la funcionalidad de la aplicación pero es difícil medirlo en la Arquitectura Web de una manera óptima para su posterior análisis.

Para la subcaracterística **Utilización de los recursos**, se da mayor relevancia al consumo de energía ocasionado por la interfaz que se usa para la conectividad, 3G o Wi-Fi. Dado que las interfaces son alternativas, nunca se pueden utilizar ambas al mismo tiempo, se asigna el mismo peso para una y para otra. Se tendrán en cuenta una o la otra en la sumatoria, nunca ambos valores. Además, se otorga más peso al consumo de energía de la pantalla que al consumo de CPU, dado que el LCD consume mucho más energía que lo que consume la CPU. Cabe destacar que el uso del CPU permite diferenciar el procesamiento realizado por los prototipos para las Arquitecturas planteadas.

En las tablas 4.25 a 4.27 se presentan las Fichas que describen los indicadores globales Nivel de Desempeño del Tiempo, Nivel de Desempeño de los Recursos y Nivel de Eficiencia. Estos indicadores se calcularán para cada arquitectura utilizada.

Tabla 4.25 - Ficha de Especificación del indicador global NDT.

Indicador Global		**Cód.: 1.1**	
Nombre	**Nivel de desempeño del Tiempo (NDT)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	**NDT = 0,25 * NDTR + 0,50 * NDBR + 0,25 * NDBE**		
Especificación	Dada la fórmula, el indicador global siempre tomará un valor perteneciente al rango real [0-10]. No se requiere mapeo alguno. 0 <= NDT <= 3 => el sistema funciona en condiciones aceptables y no necesita revisión; 3 < NDT <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	3 < NDT <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NDT <= 3		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** NDT ϵ *R* Λ NDT ϵ *[0, 10].*		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Tabla 4.26 - Ficha de Especificación del indicador global NDR.

Indicador Global		**Cód.: 1.2**	
Nombre	**Nivel de desempeño de Recursos (NDR)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	**NDT = 0,20 * NDECPU + 0,30 * NDEP + 0,50 * NDE3G ó** **NDT = 0,20 * NDECPU + 0,30 * NDEP + 0,50 * NDEWF**		
Especificación	Dada la fórmula, el indicador global siempre tomará un valor perteneciente al rango real [0-10]. No se requiere mapeo alguno. 0 <= NDR <= 2 => el sistema funciona en condiciones óptimas y no necesita revisión; 2 <= NDR <= 7 => el sistema funciona en condiciones aceptables y la revisión es optativa; 7 < NDR <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	7 < NDR <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Marginal		
Descripción	El sistema es aceptable. Revisión optativa.		
Color	Amarillo		
Rango	2 < NDR <= 7		
Criterios de Decisión		**N°**	**3**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NDR <= 2		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** NDR ϵ *R* $\wedge$ NDR ϵ *[0, 10].*		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

Tabla 4.27 - Ficha de Especificación del indicador global NE.

Indicador Global		**Cód.: 1**	
Nombre	**Nivel de Eficiencia (NE)**		
Autor	PNajar		
Versión	1.0		
Referencia	-		
Herramienta	-		
Modelo			
Función	**$NE = 0,45 * (NDT_{wifi} + NDT_{3G})/2 + 0,55 * (NDR_{wifi} + NDR_{3G})/2$**		
Especificación	Dada la fórmula, el indicador global siempre tomará un valor perteneciente al rango real [0-10]. No se requiere mapeo alguno. Se calcula el promedio de cada indicador NDT y NDR considerando su implementación en Wi-Fi y 3G. Se calcula el indicador para cada arquitectura diferente (Híbrida y Web). 0 <= NE <= 2 => el sistema funciona en condiciones óptimas y no necesita revisión; 2 <= NE <= 7 => el sistema funciona en condiciones aceptables y la revisión es optativa; 7 < NE <= 10 => es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Criterios de Decisión		**N°**	**1**
Nombre	Insatisfactorio		
Descripción	Es obligación revisar la arquitectura del sistema y la infraestructura de red utilizada en la implementación.		
Color	Rojo		
Rango	7 < NE <= 10		
Criterios de Decisión		**N°**	**2**
Nombre	Marginal		
Descripción	El sistema es aceptable. Revisión optativa.		
Color	Amarillo		
Rango	2 < NE <= 7		
Criterios de Decisión		**N°**	**3**
Nombre	Satisfactorio		
Descripción	El sistema funciona en condiciones óptimas y no necesita revisión.		
Color	Verde		
Rango	0 <= NE <= 2		
Tipo de Escala			
Representación	**Numérica**		
Tipo de Valor	**Real,** $NE \in R \wedge NE \in [0, 10]$.		
Escala	-		
Unidad			
Nombre	-		
Acrónimo	-		

En este capítulo se diseñaron, siguiendo el método GOCAME, los casos de prueba y las mediciones que son necesarias para evaluar la eficiencia de las aplicaciones móviles de turismo que tiene arquitecturas diferentes: Híbrida y Web.

En el próximo capítulo se presenta el resultado de la medición y evaluación de la eficiencia de cada aplicación que responde a una arquitectura diferente.

5. CAPÍTULO 5: RESULTADOS OBTENIDOS EN LA MEDICIÓN Y EVALUACIÓN DE LA EFICIENCIA.

Como se mencionó en el Capítulo 4, la evaluación de la calidad se realiza siguiendo las actividades propuestas en el proceso GOCAME (Lew, Olsina, Becker, & Zhang, 2011):

a. Definición de los requisitos no funcionales o NFR
b. Diseño de las mediciones
c. Diseño de la evaluación
d. Implementación de las mediciones
e. *Implementación de la evaluación*

El producto de las actividades *a*, *b* y *c*, se presentaron en el capítulo anterior. En este capítulo, se sintetizan los resultados del trabajo de campo, es decir, de las mediciones y de la evaluación (actividades *d* y *e*). Para ello se utilizan principalmente tablas y gráficos.

Para la medición se utiliza el prototipo de cada aplicación con arquitectura híbrida y web. La transacción que se usa para la medición responde a la historia de usuario expresada en el apartado 4.1: *Cuando el turista llega al punto turístico, lee con su dispositivo móvil el código QR que se encuentra siempre en un lugar externo (outdoor). De esta manera accede a la descripción detallada del punto.*

Esto implica que en ambos prototipos se efectúan las mediciones desde que el usuario solicita consultar la descripción del PI hasta el momento en que se visualiza **toda** la información en la pantalla del dispositivo.

Las mediciones se realizan siguiendo las métricas preestablecidas, tanto en el ambiente móvil como en el ambiente Wi-Fi. Dichas métricas implican el uso de un cronómetro y las herramientas PowerTutor y Traffic Monitor para la obtención de los valores.

Se obtienen las mediciones para los 3 casos de prueba presentados en el capítulo anterior: consulta de un PI que contiene solo texto, consulta de un PI que contiene texto más 3 imágenes, y consulta de un PI con texto más 5 imágenes de alta resolución.

Posteriormente, para el proceso de evaluación usando los indicadores diseñados en el Capítulo 4, se contempla solamente el caso intermedio, típico de una aplicación de m-turismo: consulta donde el resultado contiene texto e imágenes de resolución normal.

A continuación se presenta, en forma separada y secuencial, los datos obtenidos en la implementación de las mediciones (aplicación de las métricas) y en la implementación de las evaluaciones (aplicación de los indicadores elementales y globales).

5.1. RESULTADOS DE LAS MEDICIONES

Se ejecutó la aplicación de m-turismo, tanto en la versión con arquitectura híbrida como en su versión de arquitectura Web. Se tomaron mediciones para cada uno de los 3 casos de prueba, en base a las métricas definidas en el capítulo anterior.

En la tabla 5.1 se presentan los datos obtenidos en la ejecución de las mediciones de los atributos de la eficiencia, correspondientes al Árbol de Requerimientos, usando el prototipo con arquitectura híbrida. En la misma se encuentran diferenciados los dos ambientes de prueba: ambiente móvil (3G) y Wi-Fi. Además, se presentan las mediciones para cada caso de prueba: PI que contiene solo texto (0-I), PI que contiene texto más 3 imágenes (3-I) y PI con texto más 5 imágenes de alta resolución (5-I).

Tabla 5.1. Mediciones obtenidas ejecutando el prototipo con arquitectura híbrida.

Árbol de Requerimientos	**ARQUITECTURA HIBRIDA**					
	3G			**WI-FI**		
	0-I	**3-I**	**5-I**	**0-I**	**3-I**	**5-I**
Eficiencia						
Comportamiento en el tiempo						
Tiempo de respuesta en la consulta a un PI (s) Métrica: TR	3,9	9,5	14,3	2,9	4,5	6,3
Bytes Recibidos por la Aplicación Móvil en la consulta a un PI (KB) Métrica: TKBR	2,76	79,64	482,45	2,76	79,64	482,45
Bytes Enviados por la Aplicación Móvil en la consulta a un PI (KB) Métrica: TKBE	1,18	1,18	1,18	1,18	1,18	1,18
Utilización de los Recursos						
Consumo de energía del CPU (J) Métrica: TECPU	0,82	3,90	4,40	0,69	1,95	4,86
Consumo de energía de la Pantalla (J) Métrica: TEP	6,29	18,87	40,26	5,66	4,40	11,95
Consumo de energía de la interface 3G (J) Métrica: TE3G	5,66	16,49	28,46	0,00	0,00	0,00
Consumo de energía de la interface Wi-Fi (J) Métrica: TEW	0	0	0	0,08	0,72	0,72

Las métricas utilizadas para obtener cada valor son las indicadas en las tablas 4.10 a 4.16 del Capítulo 4, el nombre de la métrica figura debajo del nombre del atributo, en la columna más a la izquierda de

la tabla 5.1. En la misma columna, al lado de cada atributo, se indican las unidades de medición: segundos (s), kilobytes (KB), Joules (J).

En forma complementaria, se muestran en la tabla 5.2 las gráficas correspondientes a cada medición correspondiente de los atributos correspondientes a la subcaracterística *Utilización de los recursos* y a los atributos de cantidad de información de la subcaracterística *Comportamiento en el tiempo*. En cada caso de prueba se muestran las capturas de pantalla obtenidas en el momento de las mediciones por las herramientas PowerTutor (gráficos de curva y de torta) y Traffic Monitor. El análisis de estos resultados se presenta en el siguiente capítulo.

Cada fila de la tabla 5.2 contiene la información de un caso de prueba (0-I, 3-I y 5-I) en un determinado ambiente (3G y Wi-Fi). La primera columna correspondiente a "PowerTutor" presenta la curva de consumos obtenida en la ejecución de la transacción discriminada por CPU, pantalla y 3G/Wi-Fi; se observan claramente los picos de consumo de cada uno de estos componentes. La segunda columna de "PowerTutor" presenta un gráfico de torta donde se muestra del total de consumo de energía registrado, qué porcentaje corresponde a cada recurso (CPU, pantalla y 3G/Wi-Fi). En la columna correspondiente a "Traffic Monitor" se puede apreciar la captura de pantalla donde se registra la cantidad de datos enviados y recibidos en la transacción; por ejemplo, en la primera fila se indica 3,94 kB total, siendo 2,76 kB la cantidad de datos recibidos y 1,18 kB la cantidad de datos enviados.

En el Anexo 5 se presentan los archivos *.log* generados por la aplicación PowerTutor utilizada para la medición del consumo de energía. En estos archivos se pueden seguir en forma detallada los consumos que se registran desde que se inicia la transacción hasta finalizar la misma, discriminado por cada recurso: CPU, pantalla o LCD, 3G y Wi-Fi.

Tabla 5.2. Gráficas registradas por herramientas de medición para cada caso de prueba de la arquitectura híbrida.

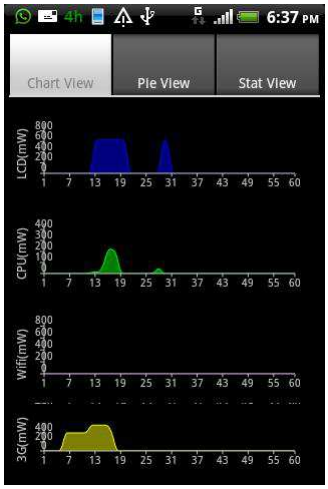
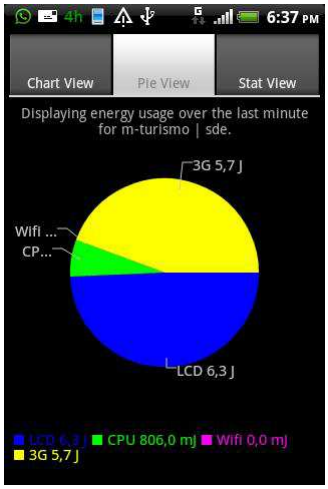
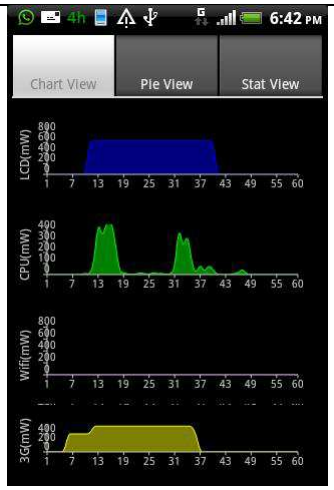
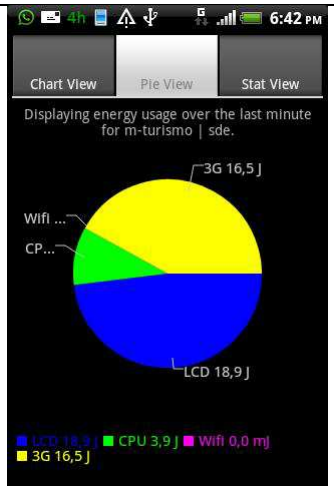
<table>
<tr><td colspan="4">ARQUITECTURA HIBRIDA</td></tr>
<tr><td colspan="4">AMBIENTE MÓVIL</td></tr>
<tr><td></td><td colspan="2">PowerTutor</td><td>Traffic Monitor</td></tr>
<tr><td>0-I</td><td> 
</td><td> 
</td><td> 
</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>3-I</td><td> 
</td><td> 
</td><td> 
</td></tr>
</table>

Tabla 5.2. Gráficas registradas por herramientas de medición para cada caso de prueba de la arquitectura híbrida (continuación).

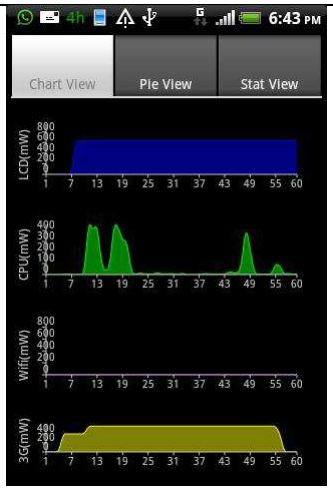
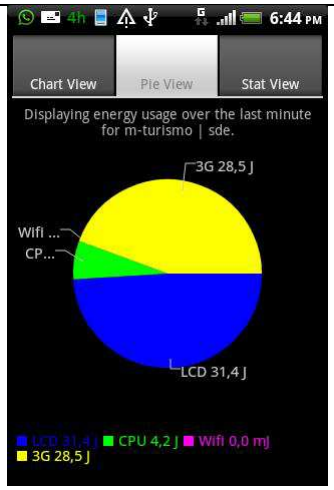
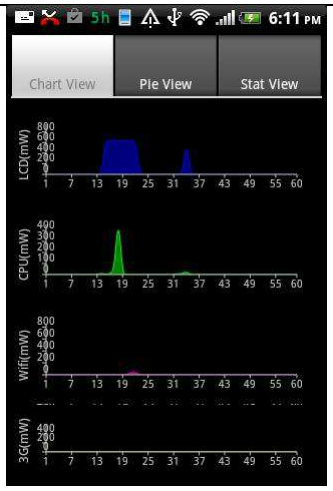
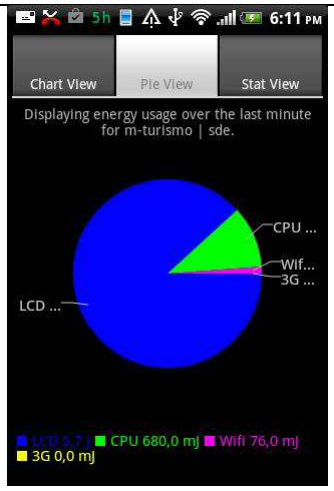
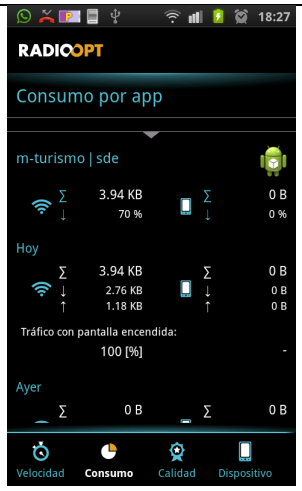
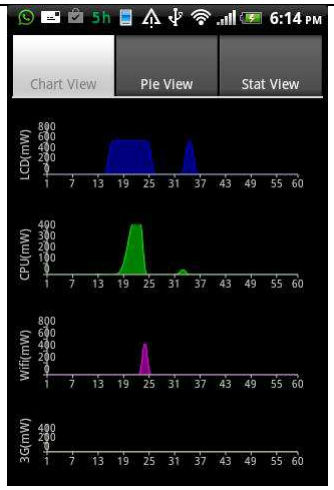
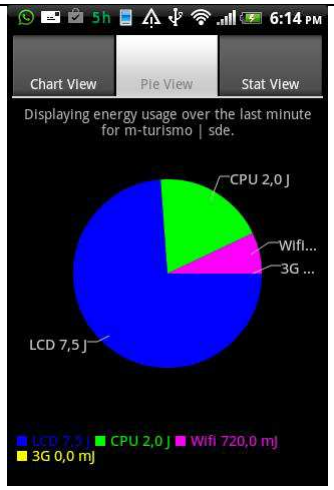
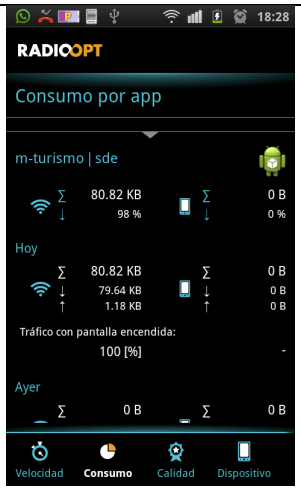
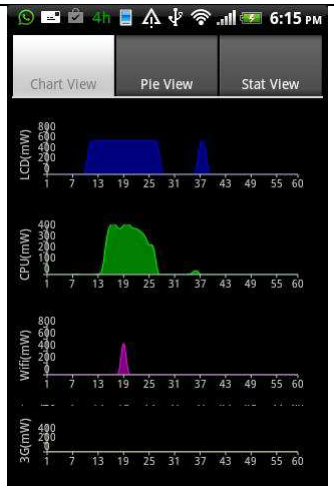
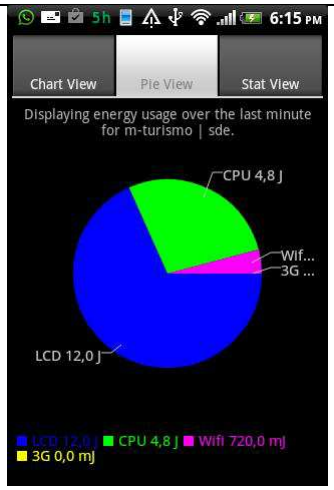
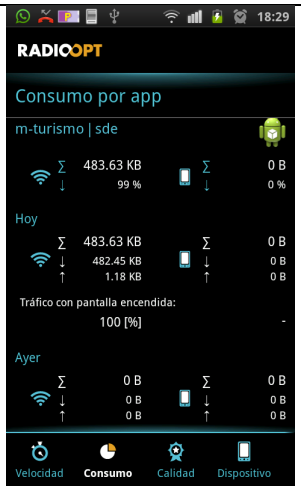
5-I			
AMBIENTE WI-FI			
	PowerTutor		**Traffic Monitor**
0-I			

Tabla 5.2. Gráficas registradas por herramientas de medición para cada caso de prueba de la arquitectura híbrida (continuación).

3-I			
5-I			

En la tabla 5.3 se muestran los datos obtenidos en la ejecución de las mediciones de los atributos de la eficiencia, correspondientes al Árbol de Requerimientos, usando el prototipo con arquitectura web. En la misma se encuentran diferenciados los dos ambientes de prueba: ambiente móvil (3G) y Wi-Fi. Además, se presentan las mediciones para cada caso de prueba: PI que contiene solo texto (0-I), PI que contiene texto más 3 imágenes (3-I) y PI con texto más 5 imágenes de alta resolución (5-I).

Las métricas utilizadas para obtener cada valor son las indicadas en las tablas 4.10 a 4.16 del Capítulo 4, el nombre de la métrica figura debajo del nombre del atributo, en la columna más a la izquierda de la tabla 5.3. En la misma columna, al lado de cada atributo, se indican las unidades de medición: segundos (s), kilobytes (KB), Joules (J).

Tabla 5.3. Mediciones obtenidas ejecutando el prototipo con arquitectura web.

Árbol de Requerimientos	ARQUITECTURA WEB					
	3G			WI-FI		
	0-I	3-I	5-I	0-I	3-I	5-I
Eficiencia						
Comportamiento en el tiempo						
Tiempo de respuesta en la consulta a un PI (s) Métrica: TR	4,2	10,7	16,9	2,94	4,62	6,2
Bytes Recibidos por la Aplicación Móvil en la consulta a un PI (KB) Métrica: TKBR	27,37	96,83	482,86	88,32	106,84	482,86
Bytes Enviados por la Aplicación Móvil en la consulta a un PI (KB) Métrica: TKBE	2,87	2,14	2,14	3,96	2,87	2,14
Utilización de los Recursos						
Consumo de energía del CPU (J) Métrica: TECPU	0,92	2,10	3,01	0,81	1,50	1,63
Consumo de energía de la Pantalla (J) Métrica: TEP	10,69	22,64	51,58	3,77	6,92	6,29
Consumo de energía de la interface 3G (J) Métrica: TE3G	7,54	21,62	50,69	0,00	0,00	0,00
Consumo de energía de la interface Wi-Fi (J) Métrica: TEW	0	0	0	0,72	0,72	2,88

En forma complementaria, se muestran en la tabla 5.4 las gráficas correspondientes a cada medición correspondiente de los atributos correspondientes a la subcaracterística *Utilización de los recursos* y a los atributos de cantidad de información de la subcaracterística *Comportamiento en el tiempo*. En cada caso de prueba se muestran las capturas de pantalla obtenidas en el momento de las mediciones por las herramientas PowerTutor (gráficos de curva y de torta) y Traffic Monitor. El análisis de estos resultados se presenta en el siguiente capítulo.

La tabla 5.4 presenta gráficas similares a las de la tabla 5.2. Es decir la tabla 5.4 puede ser interpretada siguiendo lo indicado para la aplicación con arquitectura híbrida en la tabla 5.2.

Además, también en el Anexo 5 se encontrarán los archivos .log de PowerTutor que corresponden a las gráficas de las mediciones realizadas.

Tabla 5.4. Gráficas registradas por herramientas de medición para cada caso de prueba de la arquitectura web.

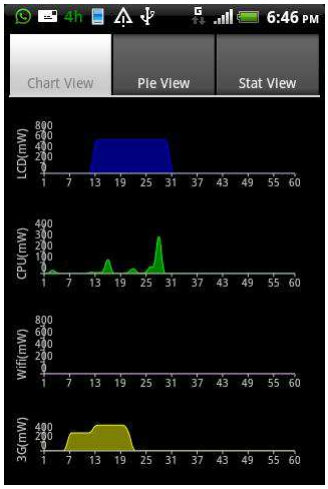
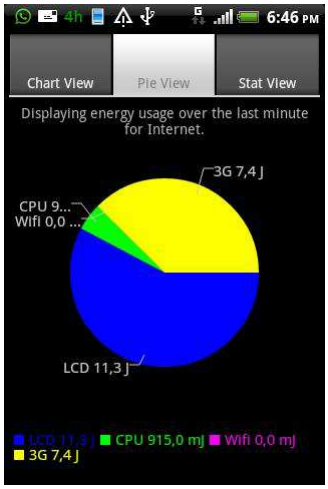
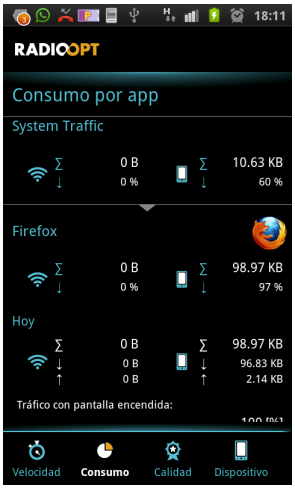
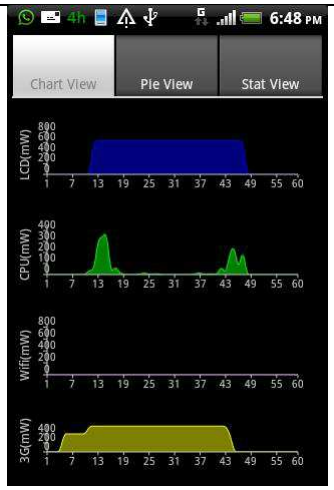
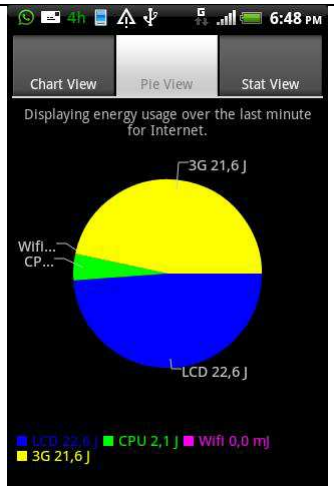
<table>
<tr><td colspan="4">ARQUITECTURA WEB</td></tr>
<tr><td colspan="4">AMBIENTE MÓVIL</td></tr>
<tr><td></td><td colspan="2">PowerTutor</td><td>Traffic Monitor</td></tr>
<tr><td>0-I</td><td> 
</td><td> 
</td><td> 
</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>3-I</td><td> 
</td><td> 
</td><td> 
</td></tr>
</table>

Tabla 5.4. Gráficas registradas por herramientas de medición para cada caso de prueba de la arquitectura web (continuación).

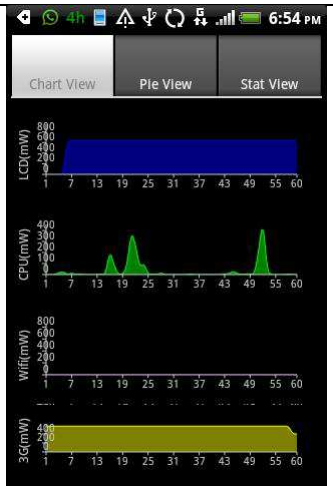
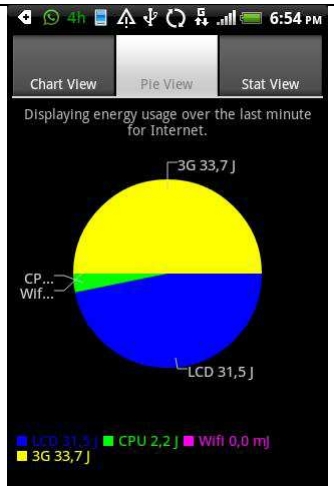
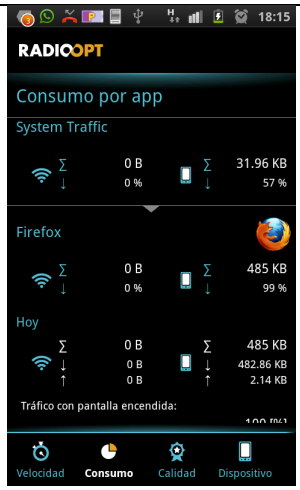
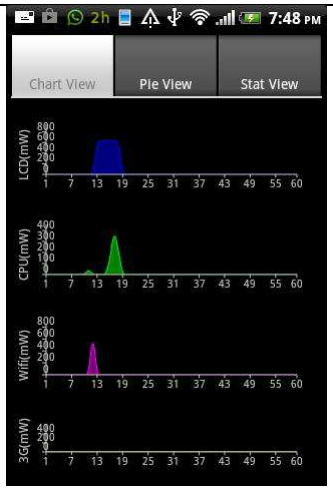
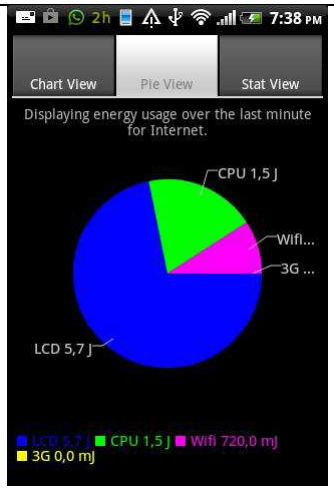
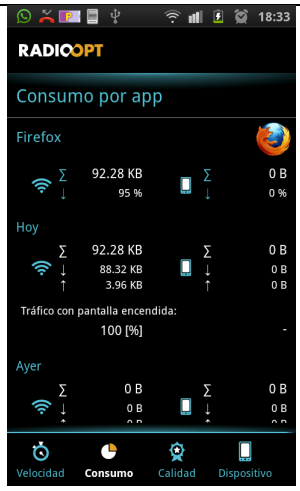
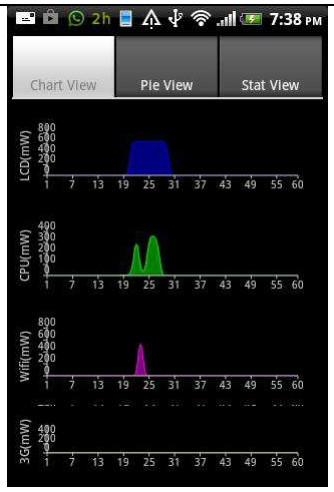
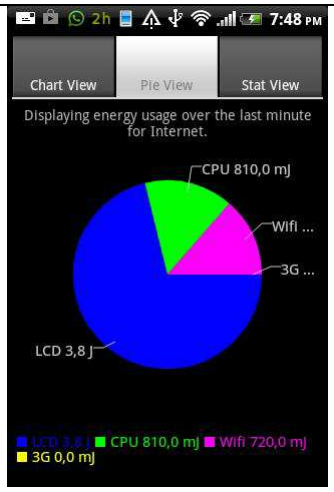
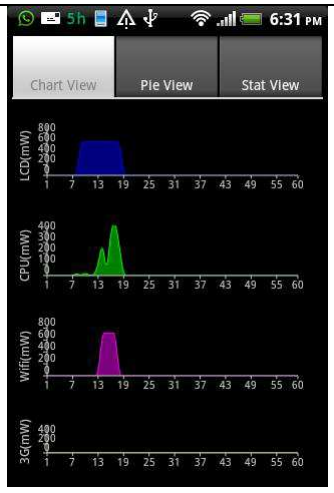
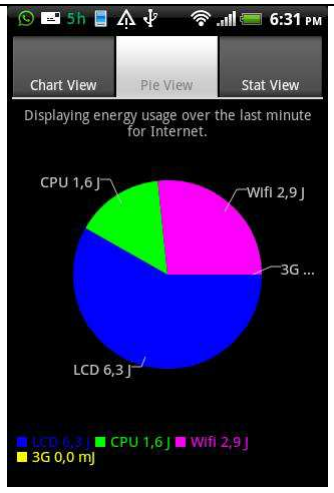
5-I			
AMBIENTE WI-FI			
	PowerTutor		**Traffic Monitor**
0-I			

Tabla 5.4. Gráficas registradas por herramientas de medición para cada caso de prueba de la arquitectura web (continuación).

3-I			
5-I			

5.2. RESULTADO DE LA EVALUACIÓN DE LA EFICIENCIA

Luego tomar las mediciones de los atributos aplicando sus métricas, se aplican sobre los valores obtenidos las fórmulas de los indicadores elementales. De esta manera, se obtiene un número que permite interpretar el atributo para poder luego tomar decisiones.

Se aplican a las mediciones los indicadores elementales diseñados, luego se interpreta el valor obtenido asignándole un color, teniendo en cuenta el criterio de evaluación. Para hacer esto, se tienen en cuenta los indicadores definidos en las tablas 4.17 a 4.23 en el capítulo anterior.

Para este proceso de evaluación se tienen en cuenta solamente el caso de prueba intermedio, típico de una aplicación de m-turismo: consulta donde el resultado contiene texto e imágenes de resolución normal. Es decir, se toma el caso **3-I**.

En este apartado se presentan los valores obtenidos para los indicadores elementales: NDTR, NDBR, NDBE, NDECPU, NDEP, NDE3G y NDEWF. Asimismo los colores **verde**, **amarillo** y **rojo** permiten hacer una rápida interpretación de los mismos, corresponden a los criterios **satisfactorio**, **marginal** e **insatisfactorio**.

En la tabla 5.5 se muestran los indicadores de la arquitectura híbrida mientras que en la tabla 5.6 se presentan los indicadores de la arquitectura Web. En la columna de la izquierda se presenta el nombre de cada indicador y una breve síntesis de los criterios de decisión. Se recuerda que en todos los casos, los indicadores diseñados mapean los valores obtenidos en las mediciones a un valor real entre 0 y 10. El análisis de estos resultados se realiza en el capítulo siguiente.

Tabla 5.5. Indicadores elementales de los atributos en la arquitectura híbrida.

Árbol de Requerimientos	**ARQUITECTURA HÍBRIDA**	
	3G	**WI-FI**
Eficiencia		
Comportamiento en el tiempo		
Nivel de Desempeño del Tiempo de Respuesta (NDTR) vi = 2 y vq= 18 verde: 0 <= NDTR <= 2; amarillo: 2 < NDTR <= 8; rojo: 8 < NDTR <= 10.	4,69	1,56
Nivel de Desempeño de Bytes Recibidos (NDBR) vi = 1 y vq= 800 verde: 0 < NDBR <= 8; rojo: (NDBR = 0) ó (8 < NDBR <= 10)	0,98	0,98
Nivel de Desempeño de Bytes Enviados (NDBE) vi = 1 y vq= 4 verde: 0 < NDBE <= 2; rojo: (NDBE = 0) ó (2 < NDBE <= 10)	0,60	0,60
Utilización de los Recursos		

Nivel de Desempeño de Consumo de Energía de CPU (NDECPU)
vi = 0,5 y vq= 5
verde: 0 <= NDEP <= 2;
amarillo: 2 < NDEP <= 7;
rojo: 7 < NDEP <= 10.
	7,56	3,23

Nivel de Desempeño de Consumo de Energía de Pantalla (NDEP)
vi = 3 y vq= 60
verde: 0 <= NDEP <= 2;
amarillo: 2 < NDEP <= 7;
rojo: 7 < NDEP <= 10.
	2,78	0,25

Tabla 5.5. Indicadores elementales de los atributos en la arquitectura híbrida (continuación).

Nivel de Desempeño de Consumo de Energía de Interfaz 3G (NDE3G)
vi = 0,05 y vq= 60
verde: 0 <= NDEP <= 1;
amarillo: 1 < NDEP <= 5;
rojo: 5 < NDEP <= 10.
	2,74	

Nivel de Desempeño de Consumo de Energía de Interfaz Wi-Fi (NDEWF)
vi = 0,05 y vq= 60
verde: 0 <= NDEP <= 1;
amarillo: 1 < NDEP <= 5;
rojo: 5 < NDEP <= 10.
		0,11

Tabla 5.6. Indicadores elementales de los atributos en la arquitectura Web.

Árbol de Requerimientos	**ARQUITECTURA WEB**	
	3G	**WI-FI**
Eficiencia		
Comportamiento en el tiempo		
Nivel de Desempeño del Tiempo de Respuesta (**NDTR)** vi = 2 y vq= 18 verde: 0 <= NDTR <= 2; amarillo: 2 < NDTR <= 8; rojo: 8 < NDTR <= 10.	5,44	1,64
Nivel de Desempeño de Bytes Recibidos **(NDBR)** vi = 1 y vq= 800 verde: 0 < NDBR <= 8; rojo: (NDBR = 0) ó (8 < NDBR <= 10)	1,20	1,32
Nivel de Desempeño de Bytes Enviados **(NDBE)** vi = 1 y vq= 4 verde: 0 < NDBE <= 2; rojo: (NDBE = 0) ó (2 < NDBE <= 10)	3,80	6,23
Utilización de los Recursos		

Nivel de Desempeño de Consumo de Energía de CPU **(NDECPU)**
vi = 0,5 y vq= 5
verde: 0 <= NDEP <= 2;
amarillo: 2 < NDEP <= 7;
rojo: 7 < NDEP <= 10.
	3,56	2,22

Nivel de Desempeño de Consumo de Energía de Pantalla **(NDEP)**
vi = 3 y vq= 60
verde: 0 <= NDEP <= 2;
amarillo: 2 < NDEP <= 7;
rojo: 7 < NDEP <= 10.
	3,45	0,69

Nivel de Desempeño de Consumo de Energía de Interfaz 3G **(NDE3G)**
vi = 0,05 y vq= 60
verde: 0 <= NDEP <= 1;
amarillo: 1 < NDEP <= 5;
rojo: 5 < NDEP <= 10.
	3,60	

Tabla 5.6. Indicadores elementales de los atributos en la arquitectura Web (continuación).

Nivel de Desempeño de Consumo de Energía de Interfaz Wi-Fi (NDEWF)
vi = 0,05 y vq= 60
verde: 0 <= NDEP <= 1;
amarillo: 1 < NDEP <= 5;
rojo: 5 < NDEP <= 10.
		0,11

Una vez obtenidos los indicadores elementales, se calcularon los indicadores globales *Nivel de Desempeño del tiempo (NDT)* y *Nivel de Desempeño de Recursos (NDR)*. Para ello, se consideran los pesos asignados a cada atributo, determinados en el capítulo anterior (ver tabla 4.24).

En la tabla 5.7 se muestran los indicadores globales NDT y NDR correspondientes a la ejecución de la aplicación con arquitectura híbrida. Por otro lado, en la tabla 5.8 se muestran los indicadores globales NDT y NDR correspondientes a la ejecución de la aplicación con arquitectura web.

Tabla 5.7. Aplicación de los indicadores globales de las subcaracterísticas, arquitectura híbrida.

Árbol de Requerimientos	**Peso**	**ARQUITECTURA HÍBRIDA**	
		3G	**WI-FI**
Eficiencia			
Nivel de Desempeño del tiempo (NDT) Mayor que 3 => ROJO		**1,81**	**1,03**
Nivel de Desempeño del Tiempo de Respuesta (NDTR)	**0,25**	4,69	1,56
Nivel de Desempeño de Bytes Recibidos (NDBR)	**0,50**	0,98	0,98
Nivel de Desempeño de Bytes Enviados (NDBE)	**0,25**	0,60	0,60
Nivel de Desempeño de Recursos (NDR) verde: 0 <= NDR <= 2; amarillo: 2 < NDR <= 7; rojo: < NDR <= 10.7		**3,72**	**0,78**
Nivel de Desempeño de Consumo de Energía de CPU (NDECPU)	**0,20**	7,56	3,23
Nivel de Desempeño de Consumo de Energía de Pantalla (NDEP)	**0,30**	2,78	0,25

Tabla 5.7. Aplicación de los indicadores globales de las subcaracterísticas, arquitectura híbrida (continuación).

Nivel de Desempeño de Consumo de Energía de Interfaz 3G (NDE3G)	**0,50**	2,74	
Nivel de Desempeño de Consumo de Energía de Interfaz Wi-Fi (NDEWF)	**0,50**		0,11

Tabla 5.8. Aplicación de los indicadores globales de las subcaracterísticas, arquitectura web.

Árbol de Requerimientos	**Peso**	**ARQUITECTURA WEB**	
		3G	**WI-FI**
Eficiencia			
Nivel de Desempeño del tiempo (NDT) Mayor que 3 => ROJO		**2,91**	**2,63**
Nivel de Desempeño del Tiempo de Respuesta (NDTR)	**0,25**	5,44	1,64
Nivel de Desempeño de Bytes Recibidos (NDBR)	**0,50**	1,20	1,32
Nivel de Desempeño de Bytes Enviados (NDBE)	**0,25**	3,80	6,23
Nivel de Desempeño de Recursos (NDR) verde: 0 <= NDR <= 2; amarillo: 2 < NDR <= 7; rojo: < NDR <= 10.7		**3,55**	**0,71**
Nivel de Desempeño de Consumo de Energía de CPU (NDECPU)	**0,20**	3,56	2,22
Nivel de Desempeño de Consumo de Energía de Pantalla (NDEP)	**0,30**	3,45	0,69
Nivel de Desempeño de Consumo de Energía de Interfaz 3G (NDE3G)	**0,50**	3,60	
Nivel de Desempeño de Consumo de Energía de Interfaz Wi-Fi (NDEWF)	**0,50**		0,11

Finalmente, se evalúa la característica ***Eficiencia*** utilizando el indicador global definido para ella, *Nivel de Eficiencia (NE)*.

Se utiliza los pesos asignados a cada subcaracterística. Se obtiene un valor promedio y definitivo del indicador de cada subcaracterística, promediando los valores obtenidos para cada tipo de conectividad (3G o Wifi). Luego, se aplica el indicador global NE. En la tabla 5.9 se muestran los resultados obtenidos.

Tabla 5.9. Nivel de Eficiencia para las aplicaciones con arquitectura híbrida y web.

Árbol de Requerimientos	**Peso**	**ARQUITECTURA HIBRIDA** *Promedio 3G-Wifi*	**ARQUITECTURA WEB** *Promedio 3G-Wifi*
Nivel de Eficiencia (NE) Verde: 0 <= NE <= 2; Amarillo: 2 <= NE <= 7; Rojo: 7 < NE <= 10		**1,88**	**2,42**
Nivel de Desempeño del tiempo (NDT)	**0,45**	1,42	2,77
Nivel de Desempeño de Recursos (NDR)	**0,55**	2,25	2,13

Sintetizando, en este capítulo se presentan los resultados obtenidos en las mediciones de los atributos usando las métricas y también los resultados del cálculo de los indicadores de las subcaracterísticas de Eficiencia y de la Eficiencia.

Si bien los colores en las tablas están indicando cómo deben interpretarse estos resultados, en el capítulo siguiente se discuten los resultados obtenidos.

6. CAPÍTULO 6: ANÁLISIS DE RESULTADOS.

En este capítulo se presenta la discusión sobre los resultados obtenidos en el capítulo anterior en el proceso de medición y evaluación de la eficiencia de aplicaciones móviles con arquitectura híbrida y arquitectura web.

Para una mejor comprensión, el análisis se divide en función de las dos subcaracterísticas de la *Eficiencia*: *Comportamiento en el Tiempo* y *Utilización de Recursos*.

6.1. COMPORTAMIENTO EN EL TIEMPO

El tiempo de respuesta es un punto crítico en éste análisis. Por un lado, es muy importante para la satisfacción del usuario final (turista) pero, por otro, la medición del tiempo de respuesta en aplicaciones con arquitectura web es difícil de realizar por código. No ocurre lo mismo en una aplicación con arquitectura híbrida, en la cual es posible medir la variable tiempo sin ningún inconveniente durante la ejecución. Dada la dificultad en las primeras aplicaciones, la medición en ambas arquitecturas, fue realizada manualmente mediante un cronómetro digital. Esto permite comparar ambas usando la misma métrica.

En el análisis de los resultados referidos al tiempo, deben ser considerados factores externos que no pueden ser controlados directamente por el método involucrado en la métrica TR. Dentro de estos factores se encuentran:

- El congestionamiento del canal de comunicaciones en horarios pico,
- La tecnología de cacheo de páginas web utilizada por los Servidores Proxy.

En las siguientes líneas se analiza cómo impacta cada uno de estos factores en el tiempo de respuesta.

El congestionamiento del canal de comunicaciones se produce generalmente en horarios pico que se dan generalmente entre las 09:00 y las 21:00 Hs. Este congestionamiento aumenta el tiempo de respuesta de la aplicación, en ambas arquitecturas.

En cuanto a la tecnología de cacheo de páginas web, disminuye el tiempo de respuesta dado que aceleran el acceso a páginas web que se encuentran en el repositorio de páginas visitadas de los servidores proxy. Por lo tanto, una aplicación con arquitectura web es beneficiada si en la comunicación al servidor de PI se encuentra un Servidor Proxy.

Analizando directamente los valores obtenidos en las mediciones realizadas aplicando las métricas, se observa una mínima diferencia de segundos en el tiempo insumido por los prototipos al consultar la información detallada del PI en cada caso de prueba (0-I, 3-I, 5-I). El prototipo con arquitectura híbrida registró un tiempo de respuesta menor. Se puede apreciar esta diferencia en la tabla 6.1; el único caso

en que el tiempo de respuesta es menor en la arquitectura web es en 5-I donde se supone actúa el Servidor Proxy, según lo explicado en el párrafo anterior. Dicha diferencia se aprecia mejor si se comparan los indicadores correspondientes.

Por lo tanto, se podría sostener que la aplicación con arquitectura híbrida es más rápida que la aplicación con arquitectura web.

Es muy importante considerar que, en las mediciones realizadas (ver tabla 6.1), se observa que el tiempo de respuesta en un *ambiente móvil* duplica al tiempo de respuesta obtenido en un ambiente Wi-Fi (en una misma arquitectura). De aquí, se podría sostener firmemente que: ***cualquiera sea la arquitectura de la aplicación móvil, siempre la misma es más rápida utilizando una conexión Wi-Fi en vez de una 3G.***

Tabla 6.1 – Comparación del Tiempo de Respuesta en prototipos con arquitectura híbrida y web.

Árbol de Requerimientos	ARQUITECTURA WEB					
	3G			WI-FI		
	0-I	3-I	5-I	0-I	3-I	5-I
Eficiencia						
Comportamiento en el tiempo						
Tiempo de respuesta en la consulta a un POI (s)	4,2	10,7	16,9	2,94	4,62	6,2

Árbol de Requerimientos	ARQUITECTURA HIBRIDA					
	3G			WI-FI		
	0-I	3-I	5-I	0-I	3-I	5-I
Eficiencia						
Comportamiento en el tiempo						
Tiempo de respuesta en la consulta a un POI (s)	3,9	9,5	14,3	2,9	4,5	6,3

Por otra parte, considerando los indicadores elementales de los atributos referidos a la cantidad de información enviada y recibida, NDBR y NDBE, estos atributos tienen un mejor desempeño en la arquitectura híbrida. Como se observa en las tablas 5.5 y 5.6 del Capítulo 5, los indicadores en la arquitectura híbrida se encuentran siempre dentro de valores aceptables (color verde) mientras que los indicadores de la información enviada se muestran en un color rojo para la arquitectura web; esto pone de manifiesto que: ***una aplicación con arquitectura hibrida tiene un mejor desempeño en la transferencia de datos (envío y recepción) entre el dispositivo móvil y el servidor de datos permitiendo recuperar la información del PI en menor Tiempo.***

6.2. UTILIZACIÓN DE RECURSOS

En cuanto a la utilización de recursos del dispositivo, se observaron comportamientos interesantes en diversos aspectos, que se presentan a continuación.

a) El del consumo de energía de las aplicaciones crece de manera exponencial cuando la información a visualizar contiene imágenes. En efecto, en la figura 6.1 se puede apreciar el crecimiento en el consumo de energía en la arquitectura web, en un ambiente móvil, al pasar del caso 0-I (sólo texto) al 3-I (texto e imágenes de baja resolución).

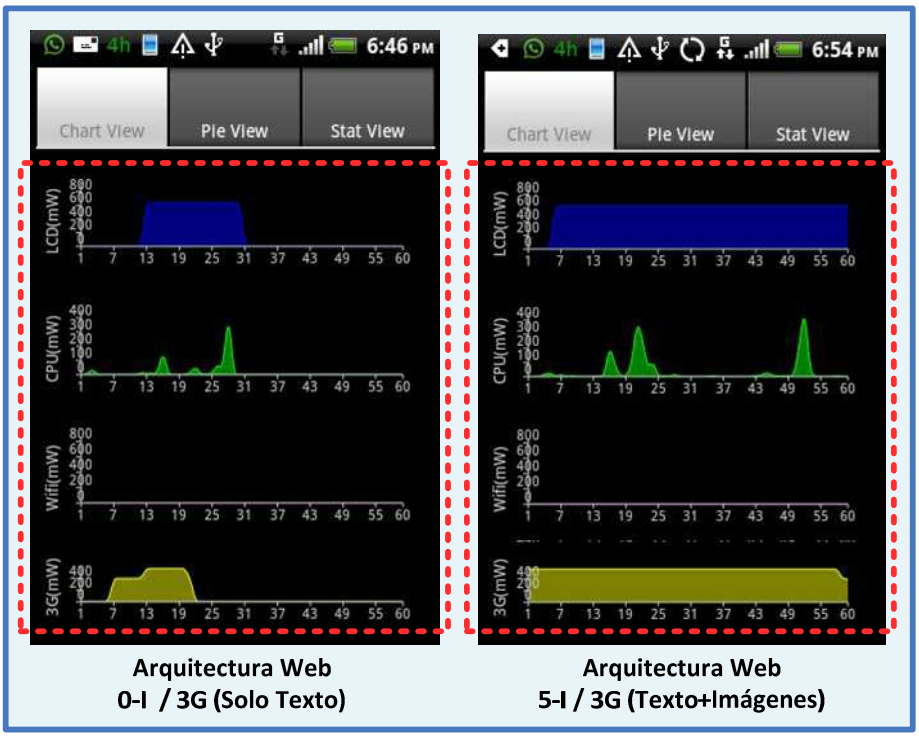


Figura: 6.1 – Comparación entre casos de pruebas de la Arquitectura Web teniendo en cuenta el consumo de energía.

b) La aplicación con arquitectura web utiliza menos CPU que la aplicación híbrida, eso surge del indicador de consumo de CPU de cada una. Es el único caso registrado con mejor desempeño que la híbrida, que descarga una menor cantidad de bytes. Esto se puede observar en la tabla 6.2.

Esta situación se puede atribuir al diseño del software (browser) que se utiliza para acceder a la aplicación web contenida en el Servidor de Web. Sin embargo, al poder tener control del código fuente de la aplicación en una arquitectura híbrida, se podría mejorar el método utilizado para la descarga y posterior visualización de las imágenes; con esto se reduciría el consumo de CPU por parte de la aplicación híbrida.

Tabla 6.2 – Comparación entre indicadores de consumo de energía de CPU en aplicaciones con arquitectura híbrida y web.

Árbol de Requerimientos	Peso	ARQUITECTURA HÍBRIDA	
		3G	WI-FI
Nivel de Desempeño de Consumo de Energía de CPU (NDECPU)	0,20	7,56	3,23

Árbol de Requerimientos	Peso	ARQUITECTURA WEB	
		3G	WI-FI
Nivel de Desempeño de Consumo de Energía de CPU (NDECPU)	0,20	3,56	2,22

c) La aplicación con arquitectura híbrida consume menos energía para el uso de la Pantalla (LCD) que la aplicación web en el ambiente Móvil. Esto se puede apreciar en la figura 6.2. Sin embargo, cabe acotar que el ambiente Wi-Fi, la arquitectura web mejoró su desempeño en los casos 0-I y 5-I. La pantalla o LCD es el recurso del dispositivo que consume más energía, para ambos prototipos. Esto es lógico, dado que la pantalla consume energía mientras los procesos asignados a dichas aplicaciones se mantengan activos.

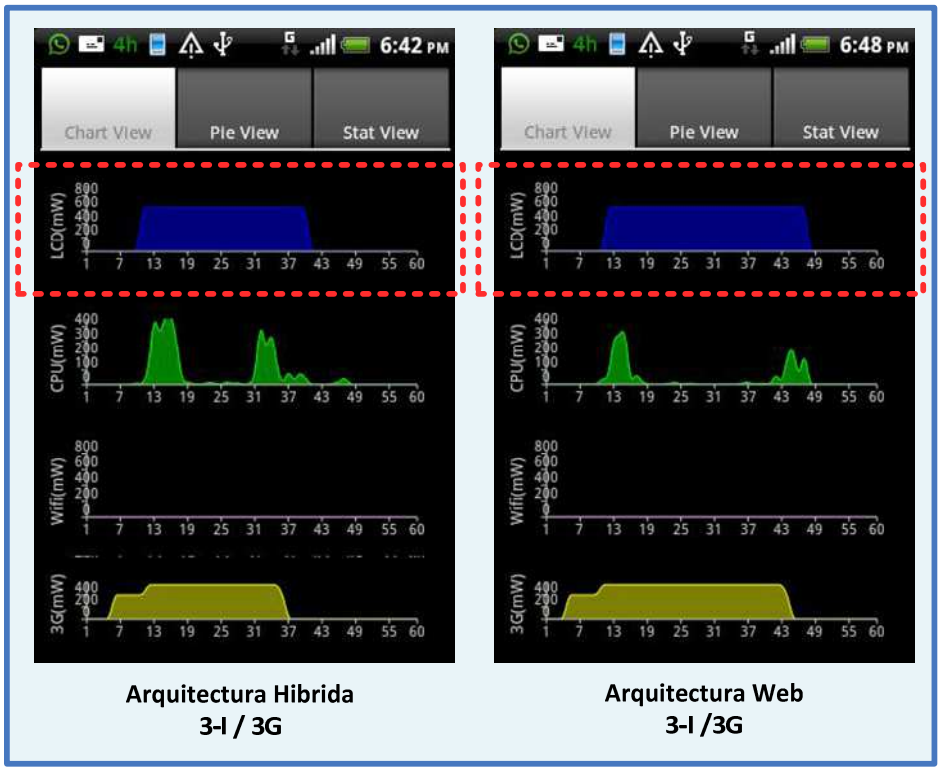


Figura: 6.2 – Comparación del consumo de energía de las aplicaciones con arquitectura híbrida y web en el ambiente móvil.

6.3. ANÁLISIS GENERALIZADO

A pesar que en el marco de este proyecto no se comparan directamente los escenarios donde se desarrollaría una aplicación de m-turismo, es importante señalar que se observó que la interfaz Wi-Fi es más eficiente que la interfaz móvil, en función del consumo de energía y tiempo de respuesta, independientemente de las arquitecturas. Por lo cual se pude sostener que: ***una aplicación de m-turismo es más eficiente en un ambiente Wi-Fi.***

En cuanto a las arquitecturas, tema central de este trabajo, se puede concluir que: ***una arquitectura híbrida es más eficiente que una arquitectura web para una aplicación de m-turismo***, en función de los valores obtenidos para el indicador global *Nivel de Eficiencia* usando los prototipos de cada arquitectura:

NE= 1,88 *(Arquitectura Híbrida)* > **NE=2,42** *(Arquitectura Web)*

En síntesis:

Una aplicación de m-turismo basada en una arquitectura híbrida es más eficiente que aquella basada en una arquitectura web

7. CAPÍTULO 7: CONCLUSIONES Y LINEAS FUTURAS DE INVESTIGACIÓN

El presente trabajo tuvo como objetivo principal determinar qué arquitectura es la más adecuada para que una aplicación de m-turismo se ejecute de manera eficiente en diferentes escenarios. Es importante considerar que el Turismo es una actividad que necesita de este tipo de estudios por sus requerimientos de movilidad, tiempo de respuesta y conectividad.

Para llevar a cabo el estudio, se analizó el entorno móvil actual y se estudiaron aspectos tecnológicos que intervienen en el diseño de una aplicación móvil, como los son: las arquitecturas, las redes de comunicaciones, los sistemas operativos móviles, las herramientas de desarrollo y aplicaciones de terceros para la medición de parámetros del dispositivo móvil. A partir de ello, se realizó la modelización de las arquitecturas de diseño de aplicaciones móviles, a la vez que se estudiaron estándares y procesos para la medición de la Eficiencia (como característica de la Calidad) de aplicaciones en general. Considerando todos los aspectos mencionados, se construyeron los prototipos de aplicaciones de m-Turismo con arquitecturas híbrida y web, utilizando Programación Extrema como metodología de desarrollo.

Se diseñaron dos ambientes de prueba, móvil y Wi-Fi, que permitieron ejecutar los prototipos usando tres casos de prueba. De esta manera, se simuló un escenario donde se ejecutaría una aplicación de m-turismo.

El uso de los prototipos en ambientes Wi-Fi y móvil, contemplando distintos casos de prueba y aplicando métricas e indicadores de eficiencia especialmente diseñados, permitieron determinar que **la arquitectura más eficiente para una aplicación móvil es la arquitectura híbrida**.

Por lo anterior descripto, que sintetiza lo realizado en los capítulos 1 a 6, se concluye que los objetivos de este trabajo fueron cumplidos en su totalidad, habiendo arribado como conclusión a la afirmación resaltada en el párrafo inmediato anterior (ver Capítulo 6).

Para llegar a esta conclusión, se concentró la investigación para la medición de la eficiencia en la ISO 25000, Calidad del Software, siendo esto un punto crítico en el proyecto ya que existen escasos antecedentes sobre la medición de la eficiencia en aplicaciones móviles. Se recurrió a ejemplos que tienen como objetivo medir la calidad de un producto software enfocados en la *usabilidad*. Este avance obtenido en el diseño de métricas y procesos de medición de Eficiencia de aplicaciones, conduce a sostener que **este trabajo constituye un aporte valioso al área de Calidad**.

En cuanto a los trabajos futuros, en el inmediato plazo el prototipo desarrollado para Android (arquitectura híbrida) podría ser utilizado en *investigaciones* que contribuyan a la optimización de aplicaciones móviles orientadas al Turismo. Simultáneamente, podría ser continuado en su desarrollo

para completar los requerimientos y ofrecer una aplicación completa para el mercado de turismo de la ciudad de Termas de Río Hondo o de Santiago del Estero, como proyecto de *extensión* al medio.

En cuanto a líneas de investigación, que específicamente pueden ser continuadas en esta temática, se sugiere el diseño de un marco sistémico que centralice toda la información de puntos de interés (de diversa índole) que permita involucrar no solo al usuario final (en este caso el turista) sino a todas las entidades relacionadas con la promoción del turismo, como los son entidades gubernamentales, prestadoras de servicios, empresas privadas de turismo, etc. El turismo es una actividad dinámica; el turista quiere alternativas, información actualizada de puntos de interés. Los prestadores y empresas de turismo necesitan llegar al turista para ofrecer sus servicios. Todos estos elementos deben relacionarse mediante el uso de nuevas tecnologías que optimicen la transacción de información entre ambos, de manera gratuita y eficiente.

Desde el punto de vista de la calidad del software, se advirtió la necesidad de contar con un proceso automatizado que oriente al informático en la medición y evaluación de las características, subcaracterísticas y atributos de la calidad. Es por ello que, se sugiere como probable línea futura, el desarrollo de una aplicación que automatice el proceso de evaluación de la estrategia GOCAME.

8. REFERENCIAS

Proyecto ZXing. (2012). *ZXing*. Obtenido de http://code.google.com/p/zxing/

Acerenza, M. Á. (1999). *Administración del turismo (6ª edición).* México.

Alba, J. (Febrero-Marzo de 2008). [Julio Alba, 2008] SOA Arquitectura Orientada al Servicio, 2008. LINK: http://dialnet.unirioja.es/servlet/articulo?codigo=2551490&orden=152333&info=link. *Bit*(167), 52-53.

Asoke K. Talukder, R. Y. (2006). *Mobile Computing: Technology, Applications, and Service Creation.*

Becker, P., Lew, P., & Olsina, L. (2012). Specifying Process Views for a Measurement, Evaluation, and Improvement Strategy. (O. Mizuno, Ed.) *Advances in Software Engineering Journal, 2012*(27).

Burbeck, S. (1997). *Applications Programming in Smalltalk-80(TM).* Obtenido de How to use Model-View-Controller (MVC): http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html

Challiol, C. (Septiembre de 2012). Computación Móvil. *Curso de Posgrado*. Universidad Nacional de Santiago del Estero.

Deacon, J. (Septiembre de 2013). *Model-View-Controller (MVC) Architecture.* Obtenido de John Deacon Computer System Devolopment, Consulting & Trainment: http://www.jdl.co.uk/briefings/MVC.pdf

Fielding Roy, T. (22 de Abril de 2008). *Architectural Styles and the Design of Network-based Software Architectures. PhD Thesis.* Obtenido de http://roy.gbiv.com/pubs/dissertation/rest_arch_style.htm

FUSDA. (14 de 02 de 2013). *FUSDA.* Obtenido de http://www.fusda.org/Revista%2014/Revista14-1ELTURISMO.pdf

International Standar Organization. (Agosto de 2013). ISO/IEC 25000:2005. *Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE*.

Komatineni, S., & MacLean, D. (2012). *Pro Android 4 .*

Lawrence, E., Pernici, B., & Krogstie, J. (2004). Mobile Information Systems. *IFIP TC8 Working Conference on Mobile Information System* (págs. -). Oslo: Springer.

Lew, P., Olsina, L., Becker, P., & Zhang, L. (2011). An integrated strategy to systematically understand and manage quality in use for Web applications. *Requirements Engineering*.

Lyytinen, K., & Yoo, Y. (2002). Issues and challenges in ubiquitous computing: Introduction. *Communications of ACM, 45*(12), 62-65.

Mezo, B., Chaparro, T., & Heras, A. (2008). Características de las empresas que utilizan arquitecturas orientadas a servicios y de su contexto de operación. *Revista de Gestão da Tecnologia e Sistemas de Informacão, 5*(2), 269-304.

Micorosoft ASP.NET Team. (Enero de 2007). *MVC Overview*. Obtenido de http://www.asp.net/mvc/tutorials/older-versions/overview/asp-net-mvc-overview

Olsina, L., & Rossi, G. (Octubre-Diciembre de 2002). Measuring Web application quality with WebQEM. *MultiMedia IEEE, 9*(4), 20-29.

Olsina, L., Lew, P., Dieser, A., & Rivera, B. (2011). Using Web Quality Models and a Strategy for Purpose-Oriented Evaluations. *Journal of Web Engineering, 10*(4), 316-352.

Olsina, L., Lew, P., Dieser, A., & Rivera, B. (2012). Updating Quality Models for Evaluating New Generation Web Applications. *Journal of Web Engineering, Special issue: Quality in new generation Web applications, 11*(3), 209-246.

Olsina, L., Pappa, F., & Molina, H. (2008). How to Measure and Evaluate Web Applications in a Consistent Way. *Web Engineering: Modelling and Implementing Web Applications, Human-Computer Interaction*, 385-420.

Open Geospatial Consortium. (10 de 05 de 2012). *Geospacial and Localization Standars*. Obtenido de Sitio web de OGC: http://www.opengeospatial.org/standards/is

Pernici, B. (2006). *Mobile Information Systems. Infraestracture an Design for Adaptivity and Flexibility.* Germany: Springer-Verlag.

Piattini, M., Garcia, F., & Caballero, I. (2007). *Calidad de Sistemas Informáticos.*

QR Code Group. (2012). *Sitio oficial del Estandar QR Code*. Obtenido de http://www.qrcode.com/en/index.html

Roy, N., Scheepers, H., & Kendall, E. (2003). Mapping the Road for Mobile Systems Development. *Pacific Asia Conference on Information Systems* (págs. 1358-1371). -: -.

Schiller, J. H. (2004). *Location-Based Services.* -: AgnèsVoisard.

Talukder, A., Ahmed, H., & Yavagal, R. (2010). *Mobile Computing: Technology, Applications, and Service Creation. Second Edition.* -: McGraw-Hill Professional.

Talukder, S., & Yavagal, G. (2006). *Mobile Computing: Technology, applications, and Service Creation.* -: McGraw-Hill Professional.

Users. (Marzo de 2009). Desarrollador. NET. *Revista Users*, -.

W3C Consortium. (11 de Febrero de 2004). *Web Services Architecture*. Obtenido de http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#whatis

W3C Consortium. (27 de Abril de 2007). *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)* . Obtenido de http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

Web, S. d. (30 de 01 de 2013). *Página de la Subsecretaria de Turismo de la Provincia de Santiago del Estero.* Obtenido de http://www.turismosantiago.gov.ar/productos/circuitos/Termas.asp

I. ANEXO 1: TURISMO

1. DEFINICIÓN

Los antecedentes de la palabra *turismo* se remontan al siglo XIX. En *The Shorter Oxford English Dictionary* se citan, con fechas de 1800 y 1811, respectivamente, los términos *tourist* y *tourism*, a los cuales se daban las acepciones siguientes (FUSDA, 2013):

a) Turista: persona que hace una o más excursiones, especialmente alguien que hace esto por recreación; alguien que viaja por placer o cultura, visitando varios lugares por sus objetos de interés, paisaje, etcétera.

b) Turismo: la teoría y la práctica de viajar, viajando por placer.

A esta primera definición siguieron otras más, las cuales con el tiempo han ido alimentando el concepto de turismo hasta llegar a definiciones más complejas. Algunas definiciones de turismo y/o turista a través de la historia han sido las siguientes:

Turismo es el concepto que comprende todos los procesos, especialmente los económicos, que se manifiestan en la afluencia, permanencia y regreso del turista hacia, en, y fuera, de un determinado municipio, estado o país (Herman von Schullern, 1911).

Turismo es la visita del espacio por otras personas que afluyen a un sitio donde no poseen lugar fijo de residencia (Robert Glücksmann, 1929).

Turismo es el conjunto de viajes cuyo objeto es el placer o los motivos comerciales o profesionales, u otros análogos, y durante los cuales la ausencia de la residencia habitual es temporal. No son turismo los viajes realizados para trasladarse al lugar de trabajo (Bormann, 1930).

Turista es la persona que entra en un país extranjero con una finalidad completamente diferente a la de fijar su residencia en él, o a la de trabajar ahí regularmente, y que gasta en este país, de residencia temporal, dinero que ha ganado en otra parte (A. J. Norwal, 1936).

Turista es toda persona que viaje durante veinticuatro horas o más por cualquier otro país distinto al de su residencia habitual (Sociedad de Naciones, 1937).

Turismo es el conjunto de las relaciones y fenómenos producidos por el desplazamiento y permanencia de personas, fuera de su lugar de domicilio, en tanto dichos desplazamientos y permanencia no estén motivados por una actividad lucrativa (Hunziker y Krapf, 1942).

Turismo es todo desplazamiento temporal, determinado por causas ajenas al lucro; el conjunto de bienes, servicios y organización que en cada nación determinan y hacen posible esos desplazamientos, y las relaciones y hechos que entre éstos y los viajeros tienen lugar (De Arrillaga, 1955).

El turismo es un fenómeno social que consiste en el desplazamiento voluntario y temporal de individuos o grupos de personas que, fundamentalmente por motivos de recreación, descanso, cultura o salud, se trasladan de su lugar de residencia habitual a otro, en el que no ejercen ninguna actividad lucrativa ni remunerada, generando múltiples interrelaciones de importancia social, económica y cultural (Óscar De la Torre Padilla).

El turismo el conjunto de actividades que realizan las personas durante sus viajes y estancias en lugares distintos al de su entorno habitual, por un período de tiempo consecutivo inferior a un año, con fines de ocio, por negocios y otros motivos, no relacionados con el ejercicio de una actividad remunerada en el lugar visitado (Organizacion Mundial del Turismo). La utilización de este amplio concepto permite identificar tanto el turismo entre países como el turismo dentro del propio país.

El turismo es un fenómeno social de carácter complejo, que puede ser interpretado de distintas formas, según sea la función que, en un momento dado, tengan las personas relacionadas con él. Pero independientemente del punto de vista particular que puedan tener los diferentes sectores dedicados a esta actividad, el turismo, desde el punto de vista conceptual, no es sino un conjunto de relaciones y fenómenos producidos por el desplazamiento y permanencia de personas fuera de su lugar normal de domicilio, motivadas fundamentalmente por una actividad no lucrativa. El turismo es, por consiguiente, una forma particular del uso del tiempo libre, y una forma especial de recreación y no incluye, por tanto, todas las formas de uso que puede hacer el hombre de su tiempo libre ni todas las formas posibles de recreación. Es, esencialmente, una actividad relacionada con la educación, el placer, el descanso y la recreación, aunque puede estar relacionado, también, con algún otro tipo de actividad… En la práctica, y para determinados propósitos, el turismo puede ser clasificado de diversas formas, cada una de ellas orientada a una necesidad específica e, incluso, puede ser identificado en función de más de una de sus características (Acerenza, 1999). Esta última definición se la considera la más completa y es la que se tuvo en cuenta para el desarrollo del trabajo.

2. CLASIFICACIÓN GENERAL

Es interesante tener en cuenta al desarrollar un trabajo relacionado con el Turismo, que existen diversos tipos de turismo. A continuación se presenta algunas de las clasificaciones más usadas.

1. Según el motivo del viaje. Consiste en identificar al turismo según el motivo principal del viaje o por el propósito de la visita a un determinado destino. De acuerdo con este criterio, el turismo puede ser clasificado en tres grandes categorías, las cuales, a su vez, se dividen en toda una gama de tipos. Estas categorías son las siguientes:

a) Turismo convencional o de tipo vacacional: es el que obedece a motivaciones relacionadas con la educación, el placer, el descanso o con la recreación.
b) Turismo especializado: es el que responde a motivaciones ligadas con las expectativas de emoción y aventura o con el interés científico.
c) Turismo de afinidad o de interés común: es el que se encuentra ligado a motivaciones de índole profesional, religiosa o filosófica.

2. Según la forma de viaje. En este caso, el turismo, independientemente del tipo de arreglo efectuado por el turista para realizar el viaje, puede ser clasificado como:
   a) Turismo individual.
   b) Turismo de grupo.

3. Según el tipo de viaje. En consideración a la forma como se hayan efectuado los arreglos relativos al viaje, el turismo puede ser clasificado en:
   a) Turismo independiente: en este caso es el propio turista el que compra en forma directa, y además por separado, cada uno de los componentes que lo integran. En otras palabras, es él quien contrata la transportación para llegar hasta el destino, contrata el alojamiento en el hotel y efectúa separadamente todos los otros gastos relativos a la alimentación, las distracciones y amenidades que desee disfrutar durante su permanencia en el lugar.
   b) Turismo organizado o todo comprendido: el turista adquiere en un solo acto de compra todos los servicios por un precio global. Este tipo de producto es conocido en la industria de viajes con el nombre de *paquete turístico* o *package*.

4. Según el tipo de operación. Las empresas que integran la industria de viajes, para efectos de una mejor identificación del campo de los negocios turísticos, clasifican al turismo en:
   a) Turismo receptivo: aquel turismo que llega al destino donde la empresa está afincada, y presta sus servicios, independientemente del punto de origen del visitante, el cual puede provenir tanto del exterior, como de cualquier otro punto localizado en el propio territorio nacional.
   b) Turismo emisivo: Por turismo emisivo se debe entender el que tiene su origen en el lugar donde está establecida la empresa o situado en sus alrededores, y que tiene como destino cualquier punto del territorio nacional o del extranjero.

5. Según la permanencia en el lugar de destino. De acuerdo con el tiempo que dure la permanencia del turista en el lugar de destino, el turismo puede ser clasificado en:
   a) Turismo itinerante: se caracteriza por mantener una permanencia muy corta en el lugar de destino y, normalmente, está relacionada con un tour, un package-tour o con un circuito, aunque, de hecho, puede dirigirse solamente a un único destino.

b) Turismo residencial o de estadía: se caracteriza por mantener una mayor permanencia en el lugar de destino y, generalmente, está asociado al uso de sistemas de alojamiento extrahoteleros (residencias, apartamentos y condominios), de donde se deriva precisamente su denominación y no de su calidad migratoria.

6. Según la Organización Mundial del Turismo, se clasifica en:
   a) Turismo interno: es el turismo de los visitantes residentes, en el territorio económico del país de referencia.
   b) Turismo receptor: es el turismo de los visitantes no residentes, en el territorio económico del país de referencia.
   c) Turismo emisor: es el turismo de los visitantes residentes, fuera del territorio económico del país de referencia.
   d) Turismo interior: es el turismo de los visitantes, tanto residentes como no residentes, en el territorio económico del país de referencia.
   e) Turismo nacional: es el turismo de los visitantes residentes, dentro y fuera del territorio económico del país de referencia.

Esta misma Organización también clasifica a los turistas según el propósito del viaje. Así, los turistas pueden ser clasificados en las siguientes categorías:

a) Ocio y recreación
b) Visitas a amigos y parientes. c) Negocios y profesionales.
c) Tratamientos de salud.
d) Religión/peregrinaciones.
e) Otros motivos (tripulación de aeronaves y embarcaciones de transporte pú- blico en tránsito y otras actividades).

3. TURISMO EN LA PROVINCIA DE SANTIAGO DEL ESTERO

Santiago del Estero es una provincia ubicada en el Noroeste Argentino, tiene una superficie de 150.536 km^2 y cuenta con más de 800.000 habitantes. Encierra un pasado de culturas ancestrales. En ella se dirimió la gesta de la emancipación del dominio español de la que cuentan valiosos testimonios de aquella época exhibidos en museos y casonas antiguas.

La capital de la provincia es la ciudad que lleva su mismo nombre, Santiago del Estero. Es la ciudad más antigua fundada por los españoles en el territorio argentino, por eso se la denomina *Madre de Ciudades* Es cuna de talentosos artesanos, poetas y músicos populares. Reserva sus tradiciones, mitos y leyendas que se transmiten de generación en generación.

Desde el punto de vista turístico la provincia posee dos ciudades importantes: Santiago del Estero y Termas de Río Hondo. En ambas, es importante el turismo cultural y religioso. Pero Termas de Río Hondo tiene algo especial: aguas termales. Esto la transformó en una ciudad que se evolucionó rápidamente desde el punto de vista turístico, favorecida por situarse en medio de dos ciudades capitales de provincia, Santiago del Estero y San Miguel de Tucumán.

Las Termas de Río Hondo actualmente está recibiendo beneficios importantes a favor de su desarrollo turístico, apoyada tanto desde el gobierno provincial como nacional. En los últimos 3 años se han construido obras de gran envergadura: aeropuerto internacional, autódromo internacional, hoteles cuatro estrellas, museo del automóvil, costanera e Isla del Sol, cancha de golf de 19 hoyos, entre otras.

En forma más detallada, se presentan a continuación los principales circuitos turísticos de la ciudad de Termas de Río Hondo (Web, 2013).

- CITY TOUR: Visita al Parque Agua Santa y monumento a San Francisco Solano, Mercado Municipal, vertiente natural La Olla, Campo de Golf, Centro de Convenciones Gral. San Martín, recorrido por Villa del Lago, Club Náutico Santiago del Estero, Yacht Club del Faro, complejo Yacu Rupaj, Represa hidroeléctrica Río Hondo, vista panorámica del dique Frontal y nacimiento del Río Dulce, regreso por Villa Balnearia, Parque de los Apóstoles, Vivero Municipal, cruce por puente del Río Dulce. Dentro de este recorrido se incluye la visita a la nueva Isla del Sol o “Tara Inti”, la cual fue tomada como Punto de Interés dentro de la aplicación de m-turismo desarrollada.
- AUTODROMO DE LAS TERMAS DE RIO HONDO: Está ubicado a 6 Km. del centro de la ciudad de Las Termas de Río Hondo en un predio de 150 hectáreas en el peri-lago del Dique Frontal. Está preparado para recibir a más de 65.000 espectadores y además cuenta con 33.000 m2 de boxes, 1000 m^2 de edificios entre ingresos, torres de control, salas de prensa y galpón de revisión técnica y estacionamiento para 25.000 automóviles dentro del perímetro del autódromo.
- VILLA RÍO HONDO: A 15 km., zona agrícola donde relata una historia milagrosa que San Francisco Solano pasó por esta antigua villa, rumbo al Tucumán para proveerse de madera de nogal y construir el templo que hoy se levanta en la Ciudad Capital de Santiago del Estero. Al regresar, se encontró en las cercanías de Villa Río Hondo con el gran río crecido (Río Dulce).
- VINARÁ: 15 km., su nombre significa "lugar de todos" y sus tierras fueron centro de las tribus de Atacama y lugar por donde pasaron también los conquistadores. Fue declarado "Lugar Histórico Nacional" por Ley Nacional Nº 12.665 en el año 1942, tanto por la firma del "Tratado de Paz entre Santiago del Estero y Tucumán", firmado por Juan Felipe Ibarra y Bernabé Araoz el 5 de Junio de 1821, cuyo monolito lo recuerda en este lugar, como también por ser declarada "Posta Histórica" ya que pernoctaron en 1816 los congresales que firmaron la "Independencia del País" en Tucumán provenientes de Buenos Aires.

En cuanto al turismo termal, Las Termas de Río Hondo constituye el principal Centro Termal y Spa de América Latina. Ubicada en la ribera del Río Dulce, se encuentra asentada sobre una terma mineralizada de un radio de 15 km., conformada por 14 napas de agua meso termal que alcanza los 65º y compuesta por una gran cantidad de minerales que le dan un gran valor curativo. Las Termas de Río Hondo está ubicada, estratégicamente, sobre la Ruta Nacional Nº 9, en comunicación directa con Tucumán, Catamarca, La Rioja, Salta y Jujuy, y a pocos kilómetros de los aeropuertos de Santiago del Estero y de San Miguel de Tucumán. Cuenta con una infraestructura hotelera con 170 establecimientos de distintas categorías que permiten disfrutar del baño termal en la propia habitación, caso único en el país.

4. TURISMO 3.0 Y LA WEB 3.0

Mientras en América Latina recién hace poco tiempo atrás se comenzó a hablar del Turismo 2.0, la Web 3.0 quiere hacer su aparición, prometiendo revolucionar la manera de navegar en Internet siendo más amigable, más accesible, más móvil, más interactiva.

Esta nueva concepción de navegación en internet puede ser aplicada al Turismo trayendo más beneficios al sector, y permitiendo, entre otras cosas, integrar contenidos multimedia e información turísticas en varios resultados de búsqueda de una manera más intuitiva y coloquial.

Para el turismo 3.0 habrá una herramienta clave: el dispositivo móvil, en especial los Smartphone y tabletas, en el que convergen las nuevas tecnologías y la Web 2.0. La razón por la que el móvil tiene tanto valor en la era 2.0 y lo tendrá en la 3.0 es que hay más dispositivos móviles que computadoras y que las personas pasan muchas más horas con sus móviles que con sus computadoras. De hecho, dichos dispositivos rara vez se apagan y pocas veces se desconectan de la red.

Características de la Web 3.0:

- Web como modelo de Base de datos distribuida y semántica.
- Uso extendido de las tecnologías evolucionadas AJAX y servicios web, además de otras como 3D.
- Integración y especialización de contenidos en un portal Web.
- Innovación en tecnologías de interfaz usuario (capa de presentación).
- Los servicios web son la tecnología para integrar contenidos de heterogéneos dentro de un mismo portal Web

II. ANEXO 2: ENTREVISTA A GERENTE DE MOVISTAR

A continuación se presenta la entrevista realizada al Gerente de Ingeniería de Clientes de una Empresa que brinda servicios de Telefonía Móvil.

Fecha de Entrevista: 08/06/2012, 10:55 A.M.

Lugar de Entrevista: Av. Rivadavia 347, Santiago del Estero, Capital

Entrevistado: Ing. Alejandro Pérez

Área o dependencia: Ingeniería de Clientes, Región NOA.

Objetivo: Realizar un análisis del estado actual de la conectividad móvil en la provincia de Santiago del Estero. Para ello es necesario conocer en detalle, ubicación, tipo de cobertura y alcance de las torres/antenas de la empresa Movistar en la Provincia de Santiago del Estero.

Desarrollo:

P: ¿Qué cobertura móvil posee en la actualidad la Provincia de Santiago del Estero?

R: La Provincia posee cobertura en casi todas sus principales localidades, concentradas en el departamento Capital y Río Hondo. Del 100 % de esta cobertura un 75% es 2G y el 25% restante 3G, concentrado este último, en el micro centro de las localidades de Santiago del Estero, la Banda y Termas de Río Hondo.

Se estipula que el número de torres con cobertura 3G, crecerán en un gran porcentaje a fines del año 2012. Pero siempre este crecimiento se concentrara en las principales localidades.

P: ¿La venta de equipos Smartphone es creciente, en la provincia con respecto a otros años?

R: La venta de equipos Smartphone y tabletas creció en porcentajes admirables con respecto a otros años, producto del uso de aplicaciones de redes sociales, mensajería, etc., que usan paquetes de datos para su funcionamiento. Es probable que en poco tiempo, la mayoría de los planes incluya en su abono el paquete de datos para la conexión del dispositivo a Internet.

P: ¿Cómo se podrá solucionar el inconveniente que se tiene en cobertura móvil in-door en algunos centros turísticos de la provincia?

El uso de amplificadores/booster para la red de telefonía móvil es una alternativa factible para mejor la conectividad en lugares específicos en donde, por razones de infraestructura, distancia, etc., no se posea una cobertura adecuada para la ejecución de aplicaciones móviles desde teléfonos celulares.

III. ANEXO 3: DOCUMENTACIÓN DE LA APLICACIÓN DE PROGRAMACIÓN EXTREMA EN EL DESARROLLO DE LOS PROTOTIPOS

Para documentar el desarrollo del prototipo desarrollado para la arquitectura híbrida usando Programación Extrema, se presenta a continuación los formularios que se usaron en dicho proyecto.

En primera instancia se definieron los integrantes del proyecto en la tabla III.1, seguido a esta se definen la historia de usuario en la tabla III.2. La definición detallada de esta última se describe en la tabla III.3. Seguido a esto, se definen los formularios que detallan las tareas que se llevaron a cabo para satisfacer las historias de usuario en las tablas III.4 a III.6.

Tabla III.1 – Definición de Integrantes.

ID	Rol	Apellido, Nombre
1	Turista	Paladea, Graciela
2	Tester	Herrera, Susana
3	Desarrollador.1 / Manager	Najar Ruiz, Pablo Javier
4	Desarrollador.2	Di Natale, Cristian

Tabla III.2 – Definición de Historias de Usuarios.

ID	Rol	Nombre	Descripción
H1	Turista	Visualización de la Información de un PI	*El turista puede visualizar la información de un PI en su dispositivo móvil a través de la lectura de su etiqueta QR.*

Tabla III.3 – Ficha H1 correspondiente a la Historia de Usuario N° 1.

Historia de usuario		**ID**	H1
		Fecha	01/07/2013
Nombre	Visualización de la Información de un PI		
Usuario	Turista		
Prioridad	Alta	**Puntos Estimados**	7.5 Puntos (1 Punto = 1 Semanma)
Descripcion			
El turista puede visualizar la información de un PI en su dispositivo móvil a través de la lectura de su etiqueta QR. La información devuelta por el servidor de puntos de interes continene texto e imágenes de alta resolución			

Tabla III.4 –Tarea T1 asignada a H1.

Tarea			**ID**	T1
			HIstoria	H1
			Inicio	10/07/2012
			Fin	31/07/2012
Nombre	Lectura de etiqueta QR			
Programador	Najar Ruiz, Pablo Javier			
Tipo	Desarrollo	**Puntos Estimados**	3 Puntos	
Descripcion				
La aplicación debe ejecutar la aplicación Barcode Scanner para obtener la URL codificada en la etiqueta QR del POI a consultar. Una vez obtenido la URL, se debe obtener el ID del POI contenida en dicha URL para su posterior consulta.				

Tabla III.5 –Tarea T2 asignada a H1.

Tarea			**ID**	T2
			HIstoria	H1
			Inicio	01/08/2013
			Fin	15/08/2013
Nombre	Consulta de POI al Servidor Web			
Programador	Najar Ruiz, Pablo Javier			
Tipo	Desarrollo	**Puntos Estimados**	2 Puntos (Semanas)	
Descripcion				
La aplicación debe consultar, a travez de Internet, a la capa de servicios alojada en el Servidor Web la información del POI identificado con la etiqueta QR leida por el aplicativo Barcode Scanner.				

Tabla III.6 –Tarea T3 asignada a H1.

Tarea			**ID**	T3
			HIstoria	H1
			Inicio	09/08/2013
			Fin	16/08/2013
Nombre	Procesamiento de información detallada del POI			
Programador	Najar Ruiz, Pablo Javier			
Tipo	Desarrollo	**Puntos Estimados**	1 Puntos (Semanas)	
Descripcion				
Una vez obtenido el resultado de la consulta, preparar dicha información para su posterior visualización en otra ventana (formulario). La misma contiene texto y puede contener imágenes de alta resolución.				

Tabla III.7 –Tarea T4 asignada a H1.

Tarea			**ID**	T4
			HIstoria	H1
			Inicio	13/08/2013
			Fin	24/08/2013
Nombre	Visualización de información detallada del POI			
Programador	Najar Ruiz, Pablo Javier			
Tipo	Desarrollo	**Puntos Estimados**	1.5 Puntos	
Descripcion				
Mostrar la información del POI obtenida en la consulta al servidor. Obtener las imagénes del servidor Web de manera asincrona para evitar el “tildado” del prototipo al descargar las imágenes para su visualziación.				

IV. ANEXO 4: CÓDIGO FUENTE ARQUITECTURA HIBRIDA

A continuación se describen algunas secciones de código del prototipo de la aplicación de m-turismo para la arquitectura híbrida. Se seleccionan las funciones más importantes, que corresponden a las tareas definidas en el Anexo 3.

```
public class PrincipalActivity extends Activity {

        private Button btnConsultar;
        private Button btnScannear;
        private EditText txtId;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_principal);

        btnConsultar = (Button)findViewById(R.id.btnConsutar_Manual);
        btnScannear = (Button)findViewById(R.id.btnScanear);
        txtId  = (EditText)findViewById(R.id.txtId_a_Consultar);
        btnConsultar.setOnClickListener(new OnClickListener() {
                        @Override
                        public void onClick(View v) {
                                ejecutar_obtener_detalles_puntos_interes();
                        }
                });
        btnScannear.setOnClickListener(new View.OnClickListener() {

                        @Override
                        public void onClick(View v) {
                                Intent intent = new
Intent("com.google.zxing.client.android.SCAN");
                         startActivityForResult(intent, 0);

                        }
                });
    }

    @Override
        public void onActivityResult(int requestCode, int resultCode, Intent intent) {
        if (requestCode == 0) {
            if (resultCode == RESULT_OK) {

                String contents = intent.getStringExtra("SCAN_RESULT");
                txtId.setText(Obtener_Id_QR(contents));
                ejecutar_obtener_detalles_puntos_interes();
            } else if (resultCode == RESULT_CANCELED) {

                 txtId.setText ("");
            }
        }
    }
```

Figura IV.1 - Sección de código de la clase principal *PrincipalActivity*

En la figura IV.1 se puede observar la sección de código perteneciente a la clase inicial *PrincipalActivity.* En la misma se puede visualizar cómo el prototipo, a través, del método *setOnClickListener* del botón *btnScannear* solicita, mediante el uso de *Intent* (ver apartado 2.2.1.2), la lectura de un código QR a la librería *Zxing*. Una vez finalizada esta tarea se procede al procesamiento del dato contenido en la etiqueta QR leída.

La función *mostrar_puntos_interes_detalles,* descripta en la figura IV.2, es la encargada de: recibir los datos obtenidos en la consulta al servidor de puntos de interés, cargar dichos datos en el intent a través del método putExtra(), y por último, lanzar la actividad *PuntosInteresDetallesActivity* para la visualización de la información del PI.

```
   public void mostrar_puntos_interes_detalles(String... arrPunto_Interes) {
       Intent i = new Intent(this, PuntosInteresDetallesActivity.class );
       i.putExtra("Id_Punto_Interes", arrPunto_Interes[0]);
       i.putExtra("Fecha_Hora", arrPunto_Interes[1]);
       i.putExtra("Titulo", arrPunto_Interes[2]);
       i.putExtra("Descripcion", arrPunto_Interes[3]);
       i.putExtra("Razon_Social_Organizador", arrPunto_Interes[6]);
       startActivity(i);
 }
```

Figura IV.2 - Código de la función *mostrar_puntos_interes_detalles*

Para obtener las imágenes alojadas en el servidor Web de manera asíncrona, se utiliza la clase *DownloadImageTask,* figura IV.3. La misma permite descargar las imágenes del servidor sin que la misma se "tilde" en el proceso de descarga.

```
package com.example.appeturismosde403;

import java.io.InputStream;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.util.Log;
import android.widget.ImageView;

class DownloadImageTask extends AsyncTask<String, Void, Bitmap> {
        ImageView bmImage;

        public DownloadImageTask(ImageView bmImage) {
                this.bmImage = bmImage;
        }

        @Override
        protected Bitmap doInBackground(String... urls) {
                String urldisplay = urls[0];
                Bitmap mIcon11 = null;
                try {
                        InputStream in = new java.net.URL(urldisplay).openStream();
                        mIcon11 = BitmapFactory.decodeStream(in);
                } catch (Exception e) {
                        Log.e("Error", e.getMessage());
                        e.printStackTrace();
                }
                return mIcon11;
        }

        @Override
        protected void onPostExecute(Bitmap result) {
                bmImage.setImageBitmap(result);
        }
}
```

Figura IV.3 - Código de la clase *DownloadImageTask*

La Función *Obtener_Punto_Interes* perteneciente a la clase *PrincipalActivity* es la encargada de acceder al servidor de puntos de interés a través de la librería HttpClient, y de esta forma, obtener la información del punto de interés a visualizar. En la figura IV.4 se puede visualizar su código.

```
private class Obtener_Punto_Interes extends AsyncTask<String, Float, String[]>{
        @Override
        protected String [] doInBackground(String... urls) {
                        HttpClient httpClient = new DefaultHttpClient();
                        String [] Resultado = new String[9];

                        String id = txtId.getText().toString();
                        HttpGet del = new HttpGet("http://m-turismo.no-ip.org:8933/
svcPuntos_Interes.svc/Puntos_Interes/" + id);
                        del.setHeader("content-type", "application/json");

                        Resultado[1]="Error al ejecutar consulta a wRESTful";

                        try
                {
                        HttpResponse resp = httpClient.execute(del);
                        String respStr = EntityUtils.toString(resp.getEntity());

                        JSONObject respJSON = new JSONObject(respStr);

                        String Id_Punto_Interes =
respJSON.getString("Id_Punto_Interes");
                        Date Fecha =
JsonDateToDate(respJSON.getString("Fecha_Hora").toString());
                        SimpleDateFormat sdf= new java.text.SimpleDateFormat("dd/MM/
yyyy");
                        String Fecha_Hora= sdf.format(Fecha);

                        String Titulo = respJSON.getString("Titulo");
                        String Descripcion = respJSON.getString("Descripcion");
                        Fecha =
JsonDateToDate(respJSON.getString("Fecha_Hora_Inicio").toString());
                        String Fecha_Hora_Inicio= sdf.format(Fecha);
                        Fecha =
JsonDateToDate(respJSON.getString("Fecha_Hora_Fin").toString());
                        String Fecha_Hora_Fin= sdf.format(Fecha);
                        String Razon_Social_Organizador =
respJSON.getString("Razon_Social_Organizador");

                        Resultado[0]=Id_Punto_Interes;
                        Resultado[1]=Fecha_Hora;
                        Resultado[2]=Titulo;
                        Resultado[3]=Descripcion;
                        Resultado[4]=Fecha_Hora_Inicio;
                        Resultado[5]=Fecha_Hora_Fin;
                        Resultado[6]=Razon_Social_Organizador;

                        return Resultado;

                }
                catch(Exception ex)
                {
                        Resultado[1]= ex.getMessage().toString();
                        System.out.println("ERROR: "+ ex.getMessage().toString());
                        return Resultado;
                }
         }
        @Override
                protected void onPostExecute(String... result) {
                mostrar_puntos_interes_detalles(result);
        }
    }
```

Figura IV.4 - Código de la función *Obtener_Punto_Interes*

V. ANEXO 5: ARCHIVO .LOG GENERADO POR LA APLICACIÓN POWERTUTOR

En este anexo se presenta como ejemplo de un archivo *.log* generado por la herramienta PowerTutor. En la figura V.1 se muestran las primeras líneas de registro para la arquitectura híbrida (Caso 0-I – PI solo texto) para el ambiente móvil.

```
phone-service in-service
phone-network GPRS
signal 25
phone-call idle
data connected
battery-change 2 100/100 4206312
signal 22
batt_temp 31.200000000000003
batt_charge 4320.0
setting_brightness automatic
setting_screen_timeout 60000
time 1377639432007
localtime_offset -10800000
model dream
batt_full_capacity 4320.0
associate 10075 edu.umich.PowerTutor@15
begin 0
total-power 629
meminfo 428264 24380 1484 94396
LCD 629
LCD-brightness 142
LCD-screen-on true
LCD-10075 629
Wifi 0
Wifi-on false
GPS 0
GPS-state-times 1.0 0.0 0.0
GPS-sattelites 0
Audio 0
Audio-on false
associate 10001 com.android.vending@80230011
associate 10002 com.htc.bg.uid.shared:10002@305177779
associate 10008 com.htc.lmw@100160314
associate 10010 com.android.browser@10
associate 10012 com.htc.fm@200174485
associate 10016 com.htc.recommend@100170784
associate 10019 android.media:10019@100145255
associate 10020 com.htc.taskmanager@100168721
associate 10022 android.uid.shared:10022@305198746
associate 10027 com.google.uid.shared:10027@10
associate 10030 com.google.android.apps.maps:10030@614040001
associate 10033 com.htc.sdm@100185202
associate 10037 com.htc.usagemonitor@100189932
associate 10038 com.android.mms@305216757
associate 10039 com.htc.reportagent@305151414
associate 10041 com.htc.cspeoplesync@305133687
associate 10042 com.google.android.googlequicksearchbox@133247963
associate 10044 com.htc.streamplayer@105184138
associate 10046 com.htc.android.mail@405199069
associate 10047 com.htc.music.uid.shared:10047@200211987
associate 10050 com.google.android.apps.uploader@1513
associate 10053 com.htc.resetnotify@10
associate 10054 com.htc.android.htcime.uid.shared:10054@401227535
associate 10057 com.google.android.gm@176
associate 10068 com.htc.htcMessageUploader@305182934
associate 10073 com.whatsapp@48219
associate 10074 com.sonyericsson.extras.liveware@30310
associate 10077 com.mymobiler.android@7
associate 10078 pry.tesis.mturismo@1
associate 10082 com.google.zxing.client.android@91
```

```
begin 1
total-power 864
LCD 629
LCD-brightness 142
LCD-screen-on true
LCD-10075 629
CPU 225
CPU-sys 52
CPU-usr 0
CPU-freq 480.0
CPU-0 13
CPU-1000 143
CPU-1001 6
CPU-1002 0
CPU-1013 0
CPU-1016 0
CPU-1017 0
CPU-2000 109
CPU-9999 0
CPU-10001 0
CPU-10002 0
CPU-10008 0
CPU-10010 0
CPU-10012 0
CPU-10016 0
CPU-10019 0
CPU-10020 0
CPU-10022 0
CPU-10027 0
CPU-10030 6
CPU-10033 0
CPU-10037 0
CPU-10038 0
CPU-10039 0
CPU-10041 0
CPU-10042 0
CPU-10044 0
CPU-10046 0
CPU-10047 0
CPU-10050 0
CPU-10053 0
CPU-10054 0
CPU-10057 0
CPU-10068 0
CPU-10073 0
CPU-10074 0
CPU-10075 81
CPU-10077 6
CPU-10078 0
CPU-10082 0
Wifi 0
Wifi-on false
3G 10
3G-on true
3G-uplinkBytes 0
3G-downlinkBytes 0
3G-packets 0
3G-state IDLE
3G-oper Movistar
3G-2000 401
3G-10077 401
GPS 0
GPS-state-times 1.0 0.0 0.0
GPS-sattelites 0
Audio 0
Audio-on false
```

```
begin 5
total-power 829
LCD 629
LCD-brightness 142
LCD-screen-on true
LCD-10082 629
CPU 190
CPU-sys 44
CPU-usr 0
CPU-freq 600.0
CPU-0 0
CPU-1000 150
CPU-1001 0
CPU-1002 0
CPU-1013 0
CPU-1016 0
CPU-1017 0
CPU-2000 167
CPU-9999 167
CPU-10001 0
CPU-10002 16
CPU-10008 0
CPU-10010 0
CPU-10012 0
CPU-10016 0
CPU-10019 0
CPU-10020 0
CPU-10022 0
CPU-10027 0
CPU-10030 0
CPU-10033 0
CPU-10037 0
CPU-10038 8
CPU-10039 0
CPU-10041 0
CPU-10042 0
CPU-10044 0
CPU-10046 0
CPU-10047 0
CPU-10050 0
CPU-10053 0
CPU-10054 16
CPU-10057 0
CPU-10068 0
CPU-10073 0
CPU-10074 0
CPU-10075 16
CPU-10077 8
CPU-10078 58
CPU-10082 0
Wifi 0
Wifi-on false
3G 10
3G-on true
3G-uplinkBytes 0
3G-downlinkBytes 0
3G-packets 0
3G-state IDLE
3G-oper Movistar
3G-2000 570
3G-10077 570
GPS 0
GPS-state-times 1.0 0.0 0.0
GPS-sattelites 0
Audio 0
Audio-on false
signal 21
```

Figura V.1 – Sección del Archivo *.log* generado por PowerTutor.

Para el cálculo de consumo de una aplicación se debe tener en cuenta el proceso asociado a la misma. En el ejemplo es el asociado con el Identificador *10078* *pry.tesis.mturismo@01* perteneciente al prototipo de m-turismo. El archivo .log se organiza por lapsos de 1 segundo (begin 1…n) y en esta figura se puede apreciar el Lapso 1 y 5, donde se registran consumos por parte de la aplicación en el CPU (58mj) en Begin 5.

En el CD anexo a este documento se presentan todos los archivos *.log* de las mediciones realizadas.