Sánchez, S., Estévez, J.I., Sepúlveda, S., & Medina, B. (2016). Comparison of performance between a native app and a mobile web application for monitoring a photovoltaic system. *Sistemas & Telemática, 14*(39), 29-40. doi:10.18046/syt.v14i39.2347

Original Research / Artículo Original / Pesquisa original - Tipo 1

Comparison of performance between a native app and a mobile web application for monitoring a photovoltaic system

Sharon Darling Sánchez Santana / sharonsanchez13@hotmail.com

José Ignacio Estévez / jostevz123@gmail.com

Sergio B. Sepúlveda Mora / sergio.sepulveda@ufps.edu.co

Byron Medina Delgado / byronmedina@ufps.edu.co

Universidad Francisco de Paula Santander, Cúcuta-Colombia.

ABSTRACT A native Android application and a multi-platform web application to monitor solar radiation and the output power of a photovoltaic system were developed, in order to establish which is more efficient using the mobile device resources. The methodology for Mobile Application Development [MAD] was adopted and free software tools –Android Studio and HTML5– were used. CPU consumption, response time in sending data and memory usage for the execution of the applications, were compared; also, the user experience was evaluated through the 6Ms survey. The transfer time of a data from the server to the mobile application executed in Chrome was 0.887 seconds, the native application transfers a data in 0.853 seconds –both times are within the acceptable ranges, since the acquisition system stores one data every 5 minutes; the average memory usage was 182 Mb for executing the application in the Chrome browser, while the native application achieved a reduction of 65%, therefore it can be concluded that the native application requires less memory usage for its execution.

KEYWORDS Android; application; database; multi-platform; monitoring.

Servicio de video bajo demanda basado en la inferencia de emociones de usuario

RESUMEN Se desarrollaron una aplicación nativa Android y una aplicación web multi-plataforma para monitorizar la radiación solar y la potencia de salida de un sistema fotovoltaico, con el fin de determinar cuál es la más eficiente en el uso de los recursos de un dispositivo móvil. Se adoptó la metodología para el Desarrollo de Aplicaciones Móviles [DAM] y se usaron herramientas de software libre [Android Studio y HTML5]. Se comparó: consumo de CPU, tiempo de respuesta en el envío de datos, y memoria en uso para la ejecución de las aplicaciones; además, se evalúo la experiencia de usuario a través de la encuesta de las 6 Ms. El tiempo de transferencia de un dato desde el servidor hasta la aplicación móvil ejecutada en Chrome fue de 0,887 segundos, la aplicación nativa, por su parte, transfirió un dato en 0,853 segundos, ambos tiempos están dentro del rango aceptable, ya que el sistema de adquisición almacena un dato cada 5 minutos. El promedio de uso de memoria fue de 182 Mb para la ejecución de la aplicación en el navegador Chrome, al ejecutar la aplicación nativa se obtuvo una reducción del 65%, por lo que se puede afirmar que la aplicación nativa requiere menos memoria para su ejecución.

PALABRAS CLAVE Android; aplicación; base de datos; multi-plataforma; supervisión.

Serviço de vídeo sob demanda com base na inferência de emoções de usuário

RESUMO Um aplicativo Android nativo e uma aplicação web multiplataforma para monitorar a radiação solar e a potência de saída de um sistema fotovoltaico foram desenvolvidos olhnado determinar qual alternativa é mais eficiente na utilização de recursos de um dispositivo móvel. Na realização das duas aplicações foi adotada a Metodologia para o Desenvolvimento de aplicações móveis [MDAM] e foram utilizadas ferramentas de software livre (Android Studio e HTML5). Comparou-se: consumo de CPU, tempo de resposta no envio de dados e a memória em uso para a execução das aplicações; foi avaliada a experiência do usuário através do Diagrama 6M. O tempo de transferência de um dado desde o servidor até a aplicação móvel executada em Chrome foi 0,887 segundos e a aplicação nativa realizou a transferência de um dado em 0,853 segundos –os tempos de transferência das duas aplicações estão dentro dos limites aceitáveis, sendo que o sistema de aquisição armazena um dado de 5 minutos. O uso médio de memória foi de 182 Mb para a execução do aplicativo no navegador Chrome, na execução do aplicativo nativo foi obtida uma redução de 65%, por isso podemos dizer que o aplicativo nativo requer menos uso de memória para a sua execução.

PALAVRAS-CHAVE Android; aplicativo; banco de dados; multiplataforma; monitoramento.

Received / Recepción: November 10, 2016 - Accepted / Aceptación: December 19, 2016 *doi:10.18046/syt.v14i39.2347*

I. Introduction

The first mobile phones go back to the 80s, during which time the Motorola Company developed the first mobile phone and triggered a continuous advance in the evolution of these devices. In 2007, Apple broke all the schemes related to the models already existing in the market, creating the first Smartphone with touch screen and with its own operating system, a powerful, elegant and efficient machine. With the appearance of the iPhone the other companies, such as Samsung, Blackberry and Nokia, were forced to innovate their models to be able to compete in the mobile phone market.

The evolution of mobile devices brought along with it a wide variety of applications –games and chats, among others–, that make it possible to perform endless activities that were unimaginable some time ago. Some characteristics that stand out of such applications are: remote access to databases, portability, high performance on information transmission and ease of distribution.

With the arrival of these mobile applications and the breakthrough of the mobile telephony, both in software and hardware, industries began to integrate these technologies to achieve greater and better coverage, especially in the area of supervision and control of a given process, thus rapidly obtaining information on the behavior of variables interacting in a system, such as pressure, temperature and liquid level sensors (Robayo, Neira, & Vásquez, 2015). There are different types of applications and each one has special characteristics that are developed to be executed in different devices, such as smartphones and tablets (Delia, Galdámez, Thomas, & Pesado, 2013).

Durán, Peinado and Rosado (2015) compared two technologies for the development of mobile applications; for this purpose, they developed a native application and a Web application in order to evaluate which one of them has a better performance. This application consists of a content compiler which sends and receives messages between server-mobile where the latest news can be observed.

In another work, Barragán, Ruiz and Gómez (2010) developed an application to monitor the consumption of electricity in a domestic or industrial network. The application was developed using the Web technology in which the data of electrical consumption can be visualized through a graphical interface; for this purpose, a

I. Introducción

Los primeros teléfonos móviles se remontan a la época de los años 80, tiempo en el cual la compañía Motorola desarrolló el primer teléfono móvil y desencadenó un continuo avance en la evolución de estos dispositivos. En 2007 Apple rompió todos los esquemas relacionados con los modelos ya existentes en el mercado, creando el primer *Smartphone* con pantalla táctil y con su propio sistema operativo, una máquina potente, elegante y eficiente. Con la aparición del iPhone las demás compañías, como Samsung, Blackberry y Nokia, se vieron obligadas a innovar sus modelos para poder competir en el mercado de la telefonía móvil.

La evolución de los dispositivos móviles trajo consigo una gran variedad de aplicaciones –juegos y chats, entre otras–, que hacen posible realizar un sinfín de actividades que tiempo atrás era inimaginable. Algunas características que se destacan de dichas aplicaciones son: acceso remoto a bases de datos, portabilidad, alto desempeño de trasmisión de información y facilidad de distribución.

Con la llegada de estas aplicaciones móviles y el gran avance de la telefonía celular, tanto en software, como en hardware, las industrias empezaron a integrar dichas tecnologías para lograr una mayor y mejor cobertura, especialmente en el área de supervisión y control de un determinado proceso, teniendo así, de manera rápida, la información del comportamiento de las variables que interactúan en un sistema, como son la presión, la temperatura y los sensores de niveles de líquido (Robayo, Neira, & Vásquez, 2015). Existen diferentes tipos de aplicaciones y cada una posee características especiales que son desarrolladas para ser ejecutadas en diferentes dispositivos, como teléfonos inteligentes y tabletas (Delia, Galdámez, Thomas, & Pesado, 2013).

Duran, Peinado y Rosado (2015) realizaron una comparación de dos tecnologías para el desarrollo de aplicaciones móviles; para ello elaboraron una aplicación nativa y una aplicación Web con el fin de evaluar cuál de las dos tecnologías posee un mejor rendimiento. Esta aplicación consiste en un recopilador de contenidos el cual envía y recibe mensajes entre servidor-móvil en donde se pueden observar las últimas noticias.

En otro trabajo, Barragán, Ruiz y Gómez (2010) desarrollaron una aplicación para supervisar el consumo de energía eléctrica en una red doméstica o industrial. La aplicación se realizó utilizando la tecnología Web en donde se pueden visualizar los datos de consumo eléctrico a través de una interfaz gráfica; para ello se desarrolló una aplicación móvil multiplataforma para la supervisión de las variables de un sistema de generación de energía fotovoltaica, en la cual se puede verificar el estado de cada variable en tiempo real y generar notificaciones en caso de errores en la lectura de datos. La aplicación accede a los valores tomados por el sistema de adquisición y se almacenan en una base de datos en la nube con intervalos de tiempo de 5 minutos.

Por otra parte, Medina, Castro y Camargo (2015), presentan una solución tecnológica para la automatización de un sistema

industrial utilizando herramientas de software y hardware basadas en código abierto en donde incorporan movilidad en los niveles de supervisión y gestión de la pirámide de automatización; esto lo consiguen a través de un sistema embebido como controlador integrado y una aplicación móvil en Android para supervisar y controlar un sistema neumático de estampado de piezas.

La diferencia entre esta propuesta de comparación de aplicaciones Web y nativa con respecto a la comparación hecha por Duran et al., (2015), radica en que se utilizan nuevas tecnologías para el desarrollo de la aplicación Web, como HTML5 [*HyperText Markup Language*] y el entorno de desarrollo para aplicaciones Web móviles *jQuery mobile*, un entorno muy ligero e idóneo para la ejecución en dispositivos móviles, lo que permite mejorar el rendimiento en la aplicación Web. Los tiempos de respuesta en ambas aplicaciones fueron cercanos a la respuesta obtenida en el análisis de Duran et al., (2015), donde la aplicación Web fue muy deficiente en comparación con la aplicación nativa de Android.

El resto del artículo está organizado de la siguiente manera. En la sección II se presenta la metodología utilizada para el desarrollo de las aplicaciones; en la sección III los resultados obtenidos de la comparación de las tecnologías desarrolladas; y en la sección IV las conclusiones del trabajo realizado.

II. Método

La **Figura 1** muestra la arquitectura general del sistema de supervisión, éste sistema multi-plataforma está compuesto por un equipo que recibe los datos del sistema fotovoltaico vía USB [*Universal Serial Bus*] a través de un microcontrolador (Hincapie, Duarte, & Sepúlveda, 2015). En el equipo se ejecuta la aplicación Java del sistema de recepción de datos que almacena los valores de las variables, por medio del protocolo HTTP (*Hypertext Transfer Protocol*) envía dichos valores al *hosting*, que posteriormente se almacenan en una base de datos MySQL.

Las dos aplicaciones creadas para la supervisión del sistema fotovoltaico se aprecian en la **Figura 1**, una versión nativa para Android y una versión Web para múltiples plataformas. La aplicación Web se encuentra almacenada en el hosting, el usuario debe introducir la dirección URL [*Uniform Resource Locator*] en el navegador del dispositivo móvil. Para acceder a la aplicación de Android, se debe instalar y ejecutar la aplicación en el dispositivo para realizar la conexión con la base de datos y extraer la información almacenada del sistema fotovoltaico. Este proceso también se realiza mediante el protocolo de comunicación HTTP.

En el desarrollo de la aplicación móvil para la supervisión del sistema fotovoltaico se aplicó una serie de pasos establecidos en la metodología para el Desarrollo de Aplicaciones Móviles [DAM] (Camargo, Sepúlveda, & Castro, 2010). Esta metodología comprende cinco etapas: Análisis, etapa en la que se reciben las peticiones y los requerimientos para el desarrollo de la aplicación; Diseño, donde se plasma la solución mediante diagrama de bloques; Desarrollo, etapa en la que se implementa el diseño en producto software y se realizan las tareas de

multi-platform mobile application was developed to monitor the variables of a photovoltaic power generation system, in which the status of each variable can be verified in real time and generate notifications in case of errors in the data reading. The application accesses the values taken by the acquisition system and these are stored in a database in the cloud with time intervals of 5 minutes.

On the other hand, Medina, Castro and Camargo (2015), presented a technological solution for the automation of an industrial system using software and hardware tools based on open source where they incorporated mobility in the supervision and management levels of the automation pyramid; this is achieved through an embedded system as an integrated controller and an Android mobile application to monitor and control pneumatic parts in a stamping system.

The difference between this proposal of comparison of Web and native applications in relation to the comparison made by Durán et al., (2015), consists in that new technologies are used for the development of the Web application, such as HTML5 [HyperText Markup Language] and the development environment for mobile web applications *jQuery mobile*, a very light and suitable environment for the execution in mobile devices, which allows to improve the performance in the Web application. Response times in both applications were close to the response obtained in the analysis made by Durán et al., (2015), where the Web application was very poor compared to the native Android application.

The rest of the article is organized as follows. Section II presents the methodology used for the development of the applications; section III shows the results obtained from the comparison of the technologies developed; and section IV points out the conclusions of this research work.

II. Method

Figure 1 shows the general architecture of the monitoring system, this multi-platform system consists of a device that receives data of the photovoltaic system via USB [Universal Serial Bus] through a microcontroller (Hincapié, Duarte, & Sepúlveda, 2015). In the computer, the Java application of the data reception system that stores the values of the variables is executed, through the protocol HTTP (Hypertext Transfer Protocol) that sends those values to the hosting, which are later stored in a MySQL database.

The two applications created for monitoring the photovoltaic system are shown in **Figure 1**, a native version for Android and a Web version for multiple platforms. The Web application is stored in the hosting; the user must enter the URL [Uniform Resource Locator] in the browser of the mobile device. To access the Android application, it must be installed and executed on the device to connect to the database and extract the stored information from the photovoltaic system. This process can also be executed by using the HTTP communication protocol.

In the development of the mobile application for the monitoring of the photovoltaic system, a series of steps were applied and established by the methodology for Mobile Application Development [MAD] (Camargo, Sepúlveda, & Castro, 2010). This methodology comprises five phases: Analysis, phase in which the requests and the requirements are received for the development of the application; Design, phase in which the solution is plotted by block diagram; Development, phase in which the design in software product is implemented and the tasks of codification, unit tests, documentation of the code and codification of aids are executed; Functional Tests, phase in which the operation of the application is verified in different scenarios and conditions –if there is an error, it is returned to the design phase–; and Delivery, in which, once the debugging phase of the system is completed and the last-minute requirements of the customer are met, then the latter is supplied with the executable, source code, documentation and system manual.

The database used to store the information of the variables of the photovoltaic system was carried out in the MySQL database manager, to which the application accesses to obtain the information of the variables whenever the user requires it to verify the operation of the system.

The management of the system database and the handling of requests made by the native application and by the Web application were made using PHP [Hypertext Preprocessor], a programming language that serves as a server-side interpreter that facilitates the interaction of the applications with the database.

The data sent to each application as a result of a request were made using the JSON [JavaScript Object Notation] format, a lightweight format for exchanging information compatible with a wide variety of programming languages –such as Java, JavaScript, PHP, C++–, which facilitates the processing of data that is sent or received in the application (Rincón, 2012).

codificación, pruebas unitarias, documentación del código y codificación de ayudas; Pruebas de Funcionamiento, donde se verifica el funcionamiento de la aplicación en diferentes escenarios y condiciones –si existe algún error se retorna a la etapa de diseño–; y Entrega, en la cual, terminada la fase de depuración del sistema y atendidos los requerimientos de última hora del cliente se suministra al mismo el ejecutable, el código fuente, la documentación y el manual del sistema.

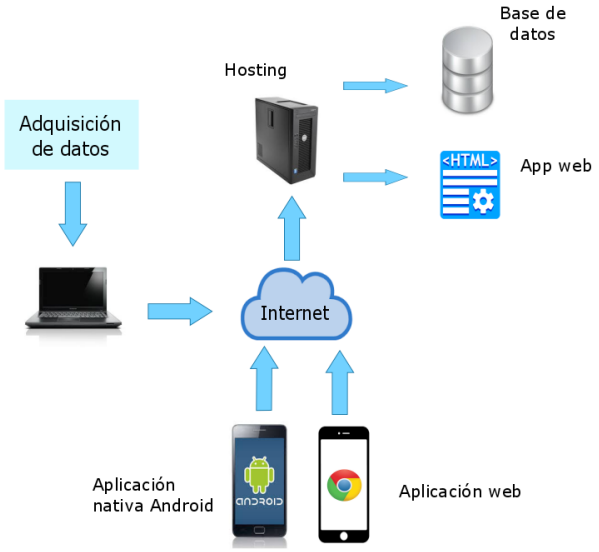

Figure 1. General structure for the monitoring system / Estructura general del sistema de supervisión

La base de datos empleada para el almacenamiento de la información de las variables del sistema fotovoltaico fue realizada en el gestor de base de datos MySQL, al cual la aplicación accede para obtener la información de las variables cada vez que el usuario requiere verificar el funcionamiento del sistema.

La gestión de la base de datos del sistema y el manejo de las peticiones realizadas por la aplicación nativa y por la aplicación Web fueron realizadas mediante el uso de PHP [*Hypertext Preprocesor*], lenguaje de programación que sirve como intérprete del lado del servidor que facilita la interacción de las aplicaciones con la base de datos.

Los datos enviados a cada aplicación, como resultado de una petición, fueron realizados usando el formato JSON [*JavaScript Object Notation*], un formato ligero para el intercambio de información compatible con una gran variedad de lenguajes de programación –como Java, JavaScript, PHP, C++–, que facilita el tratamiento de los datos que se envían o reciben en la aplicación (Rincón, 2012).

La aplicación Web fue desarrollada con el fin de dar soporte a todas las plataformas, especialmente para los sistemas operativos más usados en la actualidad: iOS [*iPhone Operative System*] y Android. HTML5 se utilizó para la estructura e interfaz de la aplicación, lenguaje que además es soportado por los navegadores de los dispositivos móviles para las plataformas mencionadas.

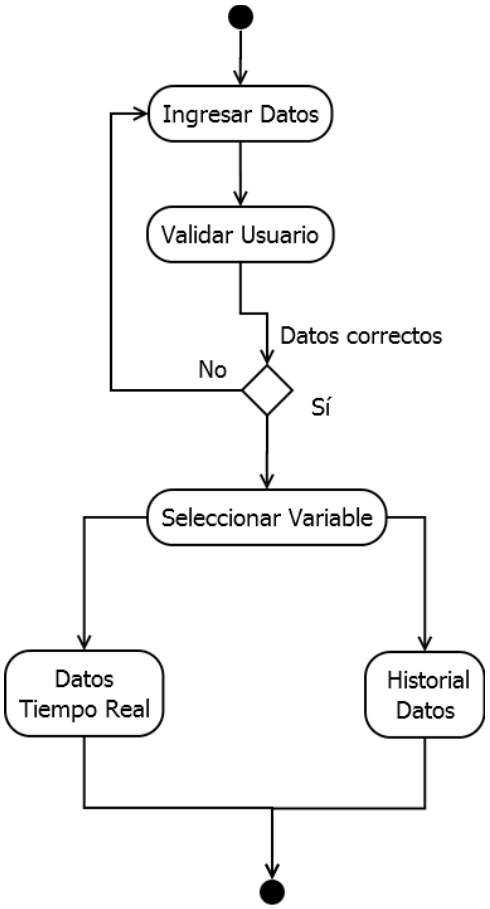

Figure 2. Operating flow chart of the application/ Diagrama de flujo de funcionamiento de la aplicación

HTML es el lenguaje de etiquetas y su versión actual HTML5 pretende proporcionar una plataforma para desarrollar aplicaciones Web más parecidas a las aplicaciones de escritorio, donde la ejecución de un navegador no implique falta de recursos o facilidades para resolver las necesidades reales de los desarrolladores.

La interfaz gráfica fue diseñada usando JQuery Mobile (Casielles, 2015), un *framework* para la creación de interfaces gráficas en el desarrollo de aplicaciones Web para teléfonos móviles que pretende unificar el diseño de interfaces para la mayoría de los dispositivos móviles del mercado, basado, como su nombre lo indica, en el *framework* jQuery de Javascript; debido a que es muy ligero, es idóneo para ejecución en dispositivos móviles, ya que estos cuentan con hardware y software limitados.

Para mejorar la visualización de la aplicación y obtener un entorno más agradable para el usuario, se utilizó el lenguaje CSS [*Cascade Style Sheets*] con el fin de ajustar los elementos gráficos a la pantalla del dispositivo empleado para la supervisión del sistema fotovoltaico.

El entorno de desarrollo de Android Studio está basado en el lenguaje de programación Java y permite utilizar todos los recursos del dispositivo móvil, como el acelerómetro, la cámara y los sensores de proximidad; además, proporciona el lenguaje de programación XML [*eXtensible Markup Language*] para la

The Web application was developed to support all platforms, especially for the most commonly used operating systems: iOS [iPhone Operative System] and Android. HTML5 was used for the structure and interface of the application, which language is also supported by the browsers of the mobile devices for the mentioned platforms.

HTML is the markup language and its current version HTML5 aims to provide a platform to develop Web applications more similar to desktop applications, where the execution of a browser does not imply a lack of resources or facilities to meet the real needs of developers.

The graphical interface was designed using JQuery Mobile (Casielles, 2015), a framework for creating graphical interfaces in the development of Web applications for mobile phones that aims to unify the interface design for most mobile devices existing in the market based, as its name implies, in the Javascript jQuery framework; due to its lightness, is suitable for execution on mobile devices, considering that these have limited hardware and software.

To improve the visualization of the application and to obtain a more pleasant environment for the user, the CSS [Cascade Style Sheets] language was used in order to adjust the graphical elements to the screen of the device used for the monitoring of the photovoltaic system.

The Android Studio development environment is based on the Java programming language and allows using all mobile device resources such as the accelerometer, camera and proximity sensors; in addition, it provides the XML [eXtensible Markup Language] programming language for creating graphical elements of each view of the application. The flow chart in **Figure 2** describes how the application works.

This flow chart represents the operation of both, the native and the Web applications. As a first requirement, the user is requested to enter the data to perform their authentication, this validation is done by a process of comparison between variables entered by the customer and the values that were stored in the database located in the hosting; the Web application uses Ajax [asynchronous JavaScript and XML] to capture and send the data entered in the user and password fields through a POST type request to the server. If the validation is correct, the main view of the application is presented, otherwise the user is asked to enter the data again.

Android platform version / *Versión de Android*	API level / *Nivel API*	Cumulative distribution / *Distribución acumulada (%)*
2.3 Gingerbread	10	100,0
4.0 Ice Cream Sandwich	15	97,3
4.1 Jelly bean	16	94,8
4.2 Jelly bean	17	86,0
4.3 Jelly bean	18	7,3
4.4 KitKat	19	70,9
5.0 Lollipop	21	35,4
5.1 Lollipop	22	18,4
6.0 Marshmallow	23	1,3

Table 1. Distribution of versions on the Android platform / Distribución de versiones en la plataforma de Android

The native Android application was developed in Android Studio, a development environment that Google has made available on its official website, was developed for version 4.4 or higher, which encompasses a large number of mobiles, as shown in **Table 1**.

The native Android application similarly performs the validation process that the Web application does, except in this case it uses the *Httpurlconnection* library for the exchange of data between the user and the system.

After the user validation has been correctly performed, it redirects to the main view of both applications, where the radiation or the power variables can be selected to perform the display of that variable. Monitoring can be performed in real time or observe the history of the monitored variable. The data that are observed when graphing the variables are extracted from the database every five minutes, time in which a new file of the photovoltaic system is saved; the data are stored in a range from five in the morning to seven at night, an optimal space for data collection.

The history option presents the historical evolution of the supervised variable in a bar chart; the application makes a request to the system which extracts the average of all the data for each day and shows in the graph the last ten days of the stored data. All requests, both for data extraction for the real-time graph and for the history graph, are performed using the HTTP protocol and GET requests.

The system can generate notifications to the user in case of a reading of data outside the specified range. For solar radiation, the readings must be between 10 W/m^2 and 1500 W/m^2 during the interval in which the system

creación de elementos gráficos de cada vista de la aplicación. En el diagrama de flujo de la **Figura 2** se describe cómo funciona la aplicación.

Este diagrama de flujo representa el funcionamiento de ambas aplicaciones, la nativa y la Web. Como primer requisito se pide al usuario ingresar los datos para realizar su autenticación, esta validación se realiza mediante un proceso de comparación de las variables ingresadas por el cliente con los valores que se almacenaron en la base de datos ubicada en el *hosting*; la aplicación Web utiliza *Ajax* [Javascript asíncrono y XML] para capturar y enviar los datos ingresados en los campos de usuario y contraseña a través de una petición de tipo POST al servidor. Si la validación es correcta se ingresa a la vista principal de la aplicación, de lo contrario se pide al usuario ingresar nuevamente los datos.

La aplicación nativa de Android fue desarrollada en Android Studio, un entorno de desarrollo que Google ha puesto a disposición en su página oficial, fue desarrollada para la versión 4.4 o superiores, las cuales abarcan un gran número de móviles, como se aprecia en la **Tabla 1**.

La aplicación nativa Android realiza de forma similar el proceso de validación que hace la aplicación Web, sólo que en este caso utiliza la librería *Httpurlconnection* para el intercambio de datos entre el usuario y el sistema.

Después de haber realizado la validación de usuario correctamente se redirige a la vista principal de ambas aplicaciones, en donde se puede seleccionar la variable de radiación o potencia para realizar la visualización de dicha variable. La supervisión se puede realizar en tiempo real u observar el historial de la variable supervisada. Los datos que se observan cuando se grafican las variables son extraídos de la base de datos cada cinco minutos, tiempo en el cuál se guarda un nuevo dato del sistema fotovoltaico; dichos datos se almacenan en un intervalo que comprende desde las cinco de la mañana hasta las siete de la noche, espacio óptimo para la recopilación de datos.

En la opción de historial se presenta la evolución histórica de la variable supervisada en una gráfica de barras; la aplicación realiza una petición al sistema el cual extrae el promedio de todos los datos por cada día y muestra en la gráfica los últimos diez días de los datos almacenados. Todas las peticiones, tanto para la extracción de datos para la gráfica en tiempo real, como para la gráfica de historial, se realizan mediante el protocolo HTTP y peticiones de tipo GET.

El sistema puede generar notificaciones al usuario en caso de ocurrir una lectura de datos fuera del rango especificado. Para la radiación solar las lecturas deben estar entre 10 W/m^2 y 1500 W/m^2 durante el intervalo en el que se estén almacenando los datos del sistema; para la potencia, las lecturas deben ser menores o iguales a 1300 W. Si se produce una lectura en el sistema de adquisición que no se encuentre en los rangos especificados para cada variable, se genera una alerta al usuario; para la generación de dicha notificación, se establece en el sistema una variable denominada error, la cual cambia de estado en el momento de recibir rangos inapropiados de lectura. En la **Figura 3** se muestra el proceso para generar la notificación al usuario después de realizar el cambio de estado de la variable de error.

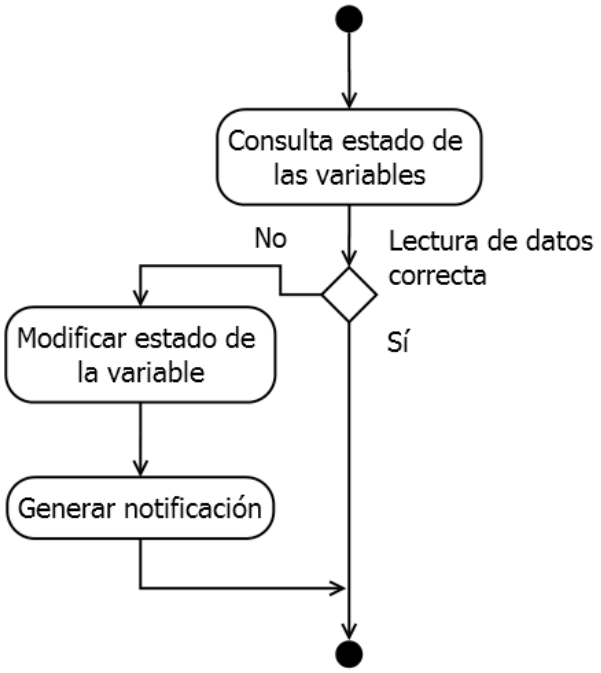

Figure 3. Notifications flow chart / Diagrama de flujo de notificaciones

Para la generación de las notificaciones de la aplicación Web se utilizó un proveedor internacional de mensajería de texto llamado *clickatell*, que permite el envío de un mensaje al dispositivo móvil del usuario con la información que se desee transmitir.

La notificación es generada a través de una petición PHP que se encarga de consultar el estado de la variable error cada vez que hay un nuevo dato en la base de datos; si el estado original de la variable ha cambiado, ejecuta una sentencia en PHP que envía un mensaje de texto al cliente informando que ocurrió un error en la lectura de datos de alguna de las variables supervisadas.

Las notificaciones en la aplicación nativa Android se generan mediante un servicio que está consultando cada cinco minutos el estado de la variable error, tiempo en el cual se genera un nuevo dato en el sistema de adquisición; si la variable ha cambiado su estado inicial, envía una respuesta a la aplicación para que genere la notificación de error, la cual se puede visualizar en la barra de tareas del dispositivo. Cuando las lecturas de datos se encuentren nuevamente en el rango establecido, la variable error retoma su estado inicial, para evitar constantes notificaciones al usuario y evitar así sobrecostos del servicio de mensajería.

Todas las consultas realizadas a la base de datos, generadas por las peticiones de cada aplicación, fueron ejecutadas usando la extensión PDO [*PHP Data Object*], que define una interfaz ligera para el acceso a base de datos en PHP, independiente del tipo de base de datos que se esté usando.

El PDO se caracteriza por las sentencias preparadas que se usaron para las consultas a la base de datos, una vez preparada la consulta es analizada una sola vez, pero puede ser ejecutada múltiples veces con los mismos o con diferentes parámetros. Si

data is stored; for power, the readings must be less than or equal to 1300 W. If a reading occurs in the acquisition system that is not in the ranges specified for each variable, an alert is generated to the user; for the generation of this notification, a variable called error is established in the system, which changes its state at the time of receiving inappropriate reading ranges. **Figure 3** shows the process to generate the notification to the user after performing the change of state of the error variable.

For the generation of notifications of the Web application was used a global text messaging provider called *clickatell*, which allows sending a message to the mobile device of the user with the information to be transmitted.

The notification is generated through a PHP request that is in charge of consulting the status of the error variable whenever there is a new data in the database; if the original state of the variable has changed, it executes a statement in PHP that sends a text message to the costumer informing that an error occurred in the reading of data of any of the supervised variables.

The notifications in the Android native application are generated by a service that is consulting every five minutes the status of the error variable, time in which a new data is generated in the acquisition system; if the variable has changed its initial state, it sends a response to the application to generate the error notification, which can be displayed in the taskbar of the device. When the data readings are again in the established range, the error variable resumes its initial state in order to avoid constant notifications to the user and avoid overcharging of the messaging service.

All queries to the database, generated by the requests of each application, were executed using the PDO extension [PHP Data Object], which defines a light interface for database access in PHP, independent of the type of database that is being used.

The PDO is characterized by the prepared statements that were used for queries to the database, once the query is prepared it is analyzed in a single time, but it can be executed many times with different or the same parameters. If an application only uses prepared statements it can be sure that there is no risk for SQL injections (Pimienta, Aguilar, Ramírez, & Gallegos, 2014), since the prepared statements are responsible for rejecting the invalid characters that can be entered in the user and password fields available for user validation. If the application is not protected against SQL injections, any

person could easily avoid user validation or remove all data from the database or extract data from it, such as user accounts and passwords.

III. Results

A. Graphical interface of the application

As mentioned, two mobile applications were developed for monitoring the photovoltaic system, a Web application for multi-platform execution and a native application on Android, which were named *AppSolar Web* and *AppSolar respectively*. The main purpose of this work is to compare the performance of each type of application and its consumption of device resources. **Figure 4** shows the user validation view for the two types of applications.

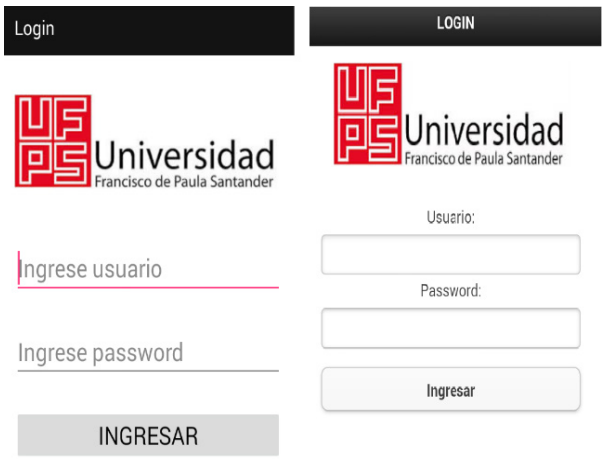

Figure 4. View of Android (a) Web (b) applications / Vista de las aplicaciones Android (a) Web (b)

B. Comparison of mobile technologies

The developed applications were executed in different cell phones with Android operating system, in order to compare the performance and consumption of resources of the device that each application requires for its correct execution. Measurements of resource consumption were made through the use of the CPU *Monitor* tool, an application that allows to examine the resource consumption of the CPU [Central Processing Unit] and memory, of the applications that are being executed in the device in a time interval.

The applications were executed, one at a time, in an interval of twenty minutes for obtaining the data; these data were processed and organized into graphs to better understand their meaning.

una aplicación usa únicamente sentencias preparadas se puede estar seguro de que no hay riesgo para inyecciones SQL (Pimienta, Aguilar, Ramírez, & Gallegos, 2014), ya que las sentencias preparadas se encargan de rechazar los caracteres inválidos que pueden ser introducidos en los campos de usuario y contraseña disponibles para la validación del usuario. Si no se protege la aplicación contra las inyecciones SQL, fácilmente una persona podría evadir la validación del usuario o eliminar todos los datos de la base de datos o extraer datos de ella, tales como cuentas de usuario y contraseñas.

III. Resultados

A. Interfaz gráfica de la aplicación

Como se indicó, se desarrollaron dos aplicaciones móviles para la supervisión del sistema fotovoltaico, una aplicación Web, para ejecución en múltiples plataformas, y una aplicación nativa en Android, que fueron nombradas *AppSolar*, con el fin de comparar el desempeño de cada tipo de aplicación y su consumo de recursos del dispositivo. En la **Figura 4** se observa la vista de validación de usuario en los dos tipos de aplicaciones.

B. Comparación de tecnologías móviles

Las aplicaciones desarrolladas fueron ejecutadas en distintos teléfonos celulares con sistema operativo Android, con el fin de comparar el desempeño y el consumo de recursos del dispositivo que requiere cada aplicación para su correcta ejecución. Las mediciones del consumo de recursos fueron realizadas a través del uso de la herramienta CPU *Monitor*, una aplicación que permite ver el consumo de recursos de la CPU [*Central Processing Unit*] y de la memoria, de las aplicaciones que se están ejecutando en el dispositivo en un intervalo de tiempo.

Las aplicaciones fueron ejecutadas, una a la vez, en un intervalo de veinte minutos para la toma de datos; dichos datos fueron procesados y organizados en gráficas para mejorar su comprensión.

En la **Figura 5** se observa el uso promedio de CPU requerido por cada aplicación. Como se aprecia en ella, la aplicación nativa de Android utiliza de forma más eficiente los recursos del dispositivo, en comparación con la aplicación Web ejecutada en los navegadores Chrome y Mozilla.

En la **Figura 6** se muestra la memoria usada por las dos

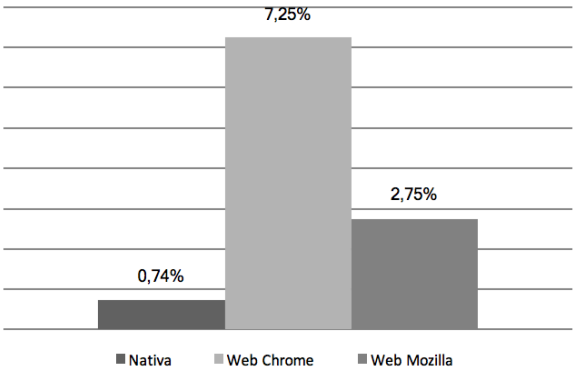

Figure 5. Maximum CPU consumption / Consumo máximo de CPU

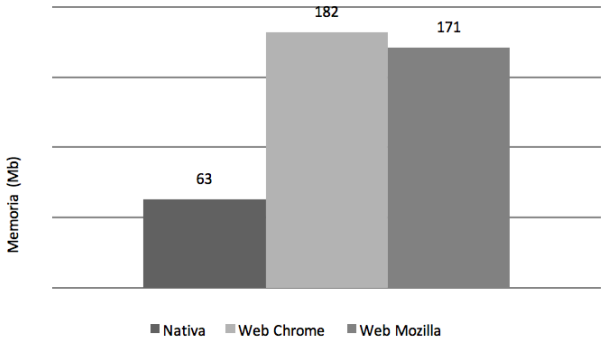

Figure 6. Web and native application memory usage / Uso de memoria por las aplicaciones nativa y web

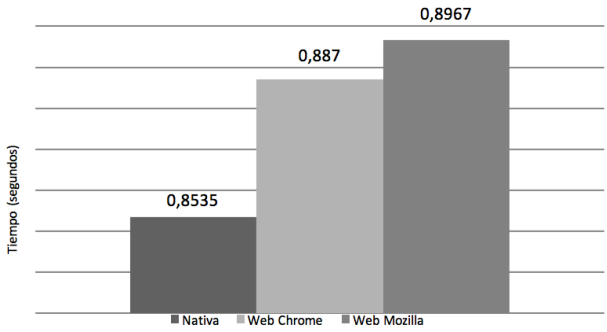

Figure 7. Response time to query a data / Tiempo de respuesta para la consulta de un dato

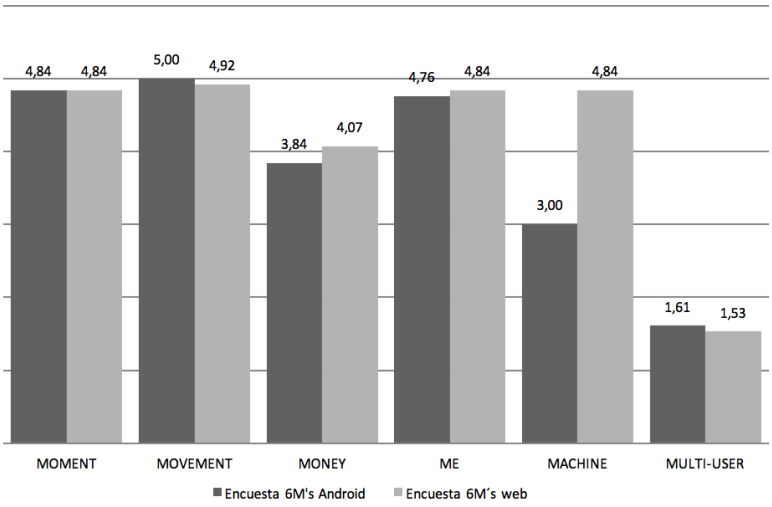

Figure 8. Results of the 6M survey / Resultados de la encuesta 6M

aplicaciones, como se aprecia en ella, la aplicación Web tuvo un uso aproximado de 200 Mb en los navegadores Chrome y Mozilla, mientras que la aplicación nativa en Android requiere tan solo de la mitad de la memoria respecto a lo requerido por la aplicación Web.

En la **Figura 7** se presenta el tiempo que tarda cada aplicación en realizar la consulta de un dato al sistema donde está almacenada la información del sistema fotovoltaico, como se observa en ella, la aplicación nativa de Android realiza con mayor rapidez esta tarea.

En la **Figura 8** se observa el resultado de las encuestas de las 6M –que se evaluó para cumplir con la metodología propuesta por Camargo et al., (2010)–, y se llevó a cabo con la participación de treinta estudiantes de últimos semestres del Programa de Ingeniería Electrónica de la Universidad Francisco de Paula Santander. Tal como se puede observar en ella, la mayoría de los atributos tuvo una buena calificación por parte de los usuarios. El atributo máquina (*machine*) obtuvo una calificación de 4,84, mejor en comparación de la obtenida por la aplicación de Android (3,0), debido a que la aplicación Web puede ser ejecutada por múltiples dispositivos y sistemas operativos, mientras que la aplicación Android solo puede ejecutarse en versiones de Android 4.4 y superiores.

Figure 5 shows the average CPU consumption required by each application. As can be noted, the native Android application uses device resources more efficiently, compared to the Web application executed in Chrome and Mozilla browsers.

Figure 6 shows the memory used by the two applications, as can be seen, the Web application had an approximate use of 200 Mb in Chrome and Mozilla browsers, while the native Android application requires only one third of the memory required by the Web application.

Figure 7 shows the time that each application takes to query a data to the system where the information of the photovoltaic system is stored, as can be seen, the native Android application performs this task faster.

Figure 8 shows the results of the 6M surveys –which was evaluated to comply with the methodology proposed by Camargo et al., (2010)–, and was performed with the participation of thirty students from the last semesters of the Program of Electronic Engineering of the Francisco de Paula Santander University. As can be noted, most of the attributes had a good rating by the users. The machine attribute earned a rating of 4.84, which is better than the Android application (3.0), because the Web application can be execute by multiple devices and operating systems, while the Android application can only be executed on versions of Android 4.4 and higher.

IV. Conclusions

The average CPU usage at the time of executing the Web application in the Chrome browser was approximately 7%, however, executing the application in

the Mozilla browser used one third of the required by Chrome, also the Android application was executed with one tenth part of the average CPU usage used by Chrome; therefore, it is concluded that the native application employs fewer resources for its execution compared to Web browsers.

The average memory usage was 182 Mb for the execution of the application in the Chrome browser; also, executing the Web application on the Mozilla browser had a 6% reduction in memory usage in relation to Chrome; on the other hand, in the execution of the native application a reduction of 65% was achieved compared to Chrome; so, it can be concluded that the native application requires less memory for its execution.

The transfer time of a datum from the server to the mobile application executed in Chrome was 0.887 seconds, while the application executed in Mozilla required 0.896 seconds and the native application performed the transfer of a datum in 0.853 seconds, that is the transfer times of the two applications are suitable for the system because the case study of the photovoltaic station updates its data every five minutes.

In general, applications were well accepted by users in four of the six attributes of the 6M evaluation, however, in the machine attribute there is a difference of 38% because the Web application can be executed on multiple platforms while the native application is restricted to cellphones with Android operating system in versions of 4.4 or higher.

We recommend to migrate toward hybrid technologies with frameworks as AngularJS to improve the web mobile apps performance; although it is probable to obtain a lightly lower performance –respect to a native app, because each page must be rendered from the server–, it supposes a substantial improvement respect to traditional methodologies. ST

IV. Conclusiones

El uso promedio de CPU al momento de ejecutar la aplicación Web en el navegador Chrome obtuvo un valor aproximado de 7%, sin embargo, ejecutar la aplicación en el navegador Mozilla utiliza un tercio del requerido por Chrome, asimismo la aplicación de Android se ejecutó con una décima parte con respecto al uso promedio de CPU usado por Chrome; por lo tanto, se concluye que la aplicación nativa emplea menos recursos para su ejecución en comparación con los navegadores Web.

El promedio de uso de memoria fue de 182 Mb para la ejecución de la aplicación en el navegador Chrome; asimismo, la ejecución de la aplicación Web en el navegador Mozilla tuvo una reducción del 6 % en el uso de memoria con respecto a Chrome; por otra parte, en la ejecución de la aplicación nativa se obtuvo una reducción del 65 % en comparación con Chrome; entonces se puede afirmar que la aplicación nativa requiere menos memoria para su ejecución

El tiempo de transferencia de un dato desde el servidor hasta la aplicación móvil ejecutada en Chrome fue de

0,887 segundos, mientras que la aplicación ejecutada en Mozilla requirió 0,896 segundos y la aplicación nativa realizó la transferencia de un dato en 0,853 segundos, es decir que los tiempos de transferencia de las dos aplicaciones son adecuados para el sistema debido a que el caso de estudio de la estación fotovoltaica actualiza sus datos cada cinco minutos.

En general las aplicaciones obtuvieron una buena aceptación por parte de los usuarios en cuatro de los seis atributos de la evaluación de las 6M, sin embargo, en el atributo máquina hay una diferencia de 38% debido a que la aplicación Web puede ejecutarse en múltiples plataformas mientras que la aplicación nativa está restringida a móviles con sistema operativo Android en versiones de 4.4 o superior.

Para mejorar el rendimiento de aplicaciones web móviles se sugiere migrar a tecnologías híbridas con *frameworks* como AngularJS. En este caso, aunque es probable que el rendimiento sea ligeramente inferior al de una aplicación nativa, puesto que cada página debe ser *renderizada* desde el servidor, supone una mejora sustancial con respecto a las tecnologías tradicionales. ST

References / Referencias

Barragán, A., Ruiz, C., & Gómez, E. (2010). Diseño de una aplicación adaptativa para monitoreo remoto a través de tecnologías móviles. *Redes de Ingeniería, 1*(1), 43-53.

Camargo, L., Sepúlveda, S., & Castro, S. (2010). Aplicación móvil de telemedicina para pacientes hipoglucémicos y diabéticos. *Respuestas, 15*(2), 52-62.

Casielles, J. (2015). *Desarrollo de aplicaciones web para dispositivos móviles con Jquery Mobile* [thesis] Universitat Politécnica de Valencia: España.

Delia, L., Galdámez, N., Thomas, P., & Pesado, P. (2013). Un análisis experimental de tipo de aplicaciones para dispositivos móviles. *XVIII Congreso Argentino de Ciencias de la Computación,* (pp. 766-776).

Durán, Á., Peinado, J., & Rosado, A. (2015). Comparación de dos tecnologías de desarrollo de aplicaciones móviles desde la perspectiva del rendimiento como atributo de calidad. *Scientia Et Technica, 20*(1), 81-87.

Hincapié, D. M., Duarte, G. G., & Sepúlveda, S. B. (2015). Low-cost and reliable wireless communication system for monitoring a photovoltaic source. *In Communications and Computing (COLCOM), 2015 IEEE Colombian Conference on.* IEEE.

Medina, B., Castro, S. A., & Camargo, L. L. (2015). Tecnologías de código abierto para la gestión de un proceso industrial. *Gerencia Tecnológica Informática, 14*(38), 43-58.

Pimienta, R., Aguilar, G., Ramírez, M., & Gallegos, G. (2014). Métodos de programación segura en Java para aplicaciones móviles en Android. *Ciencia Ergo Sum, 21*(3), 243-248.

Rincón, P. (2012). Aplicaciones móviles nativas con consumo de APIS online, estudio comparado con aplicaciones web móviles en iOS y Android y caso práctico de "native client" para WordPress [thesis]. Universidad Carlos III de Madrid: España.

Robayo, F., Neira, J., & Vásquez, M. (2015). Aplicación móvil Android para monitoreo y registro del estado nutricional humano implementada en plataforma de hardware libre. *Sistemas & Telemática, 13*(32), 75-88.

CURRICULUM VITAE

Sharon D. Sánchez Santana Electronics Engineer and member of GIDT (Research and Development in Telecommunications Group) from Universidad Francisco de Paula Santander (Cucuta, Colombia). Her main areas of interest are: mobile solutions, telemetry, and web programming / Ingeniera Electrónica y miembro del Grupo de Investigación y Desarrollo en Telecomunicaciones [GIDT] de la Universidad Francisco de Paula Santander de Cúcuta (Colombia). Sus áreas de interés son: soluciones móviles, telemetría y programación web.

José I. Estévez Mendoza Java programmer at SmartSoft Colombia. Electronics Engineer from Universidad Francisco de Paula Santander (Cucuta, Colombia). His main areas of interest are: mobile solutions, telemetry, and web programming / Desarrollador en Java para SmartSoft Colombia. Ingeniero Electrónico de la Universidad Francisco de Paula Santander de Cúcuta (Colombia). Sus áreas de interés son: soluciones móviles, telemetría y programación web.

Sergio B. Sepúlveda Mora Full-time professor at the Department of Electricity and Electronics, and researcher of GIDT (Research and Development in Telecommunications Group) and GIDMA (Research and Development in Microelectronics Group). Electronics Engineer from Universidad Francisco de Paula Santander (Cucuta, Colombia) and Master of Science in Electrical and Computer Engineering from the University of Delaware (Newark, DE). His main areas of interest are: photovoltaic solar energy, wireless sensor networks, embedded systems and mobile solutions / Profesor tiempo completo del Departamento de Electricidad y Electrónica e investigador del Grupo de Investigación y Desarrollo en Telecomunicaciones [GIDT] y del Grupo de Investigación y Desarrollo en Microelectrónica Aplicada [GIDMA], de la Universidad Francisco de Paula Santander. Ingeniero Electrónico de la Universidad Francisco de Paula Santander de Cúcuta (Colombia) y Magíster en Ingeniería Eléctrica y Computación de la Universidad de Delaware en Newark, DE (Estados Unidos). Sus áreas de interés son: energía solar fotovoltaica, redes inalámbricas de sensores, sistemas embebidos y soluciones móviles.

Byron Medina Delgado Full-time professor at the Department of Electricity and Electronics, and researcher of GIDT (Research and Development in Telecommunications Group). Electronics Engineer from Universidad Francisco de Paula Santander (Cucuta, Colombia) and Master in Electronics Engineering from the National Experimental University of Tachira (San Cristobal, Venezuela). His main areas of interest are: telecommunications, electromagnetic theory, mobile solutions and technology management / Profesor tiempo completo del Departamento de Electricidad y Electrónica e investigador del Grupo de Investigación y Desarrollo en Telecomunicaciones [GIDT], de la Universidad Francisco de Paula Santander. Ingeniero Electrónico de la Universidad Francisco de Paula Santander de Cúcuta (Colombia) y Magíster en Ingeniería Electrónica de la Universidad Nacional Experimental del Táchira de San Cristóbal (Venezuela). Sus áreas de interés son: telecomunicaciones, teoría electromagnética, soluciones móviles y gestión tecnológica.