Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

“Plugin para PhoneGap: un plugin para el ambiente de trabajo PhoneGap permitiendo utilizar el SDK de la plataforma TestFlight”

Informe Práctica de Especialidad para optar por el título de Ingeniero en Computación con el grado académico de Bachiller

Brian Alberto Corella Pérez

San Carlos, Noviembre, 2013

Resumen Ejecutivo

Este proyecto se realizó con el objetivo de realizar el diseño y desarrollo de un plugin, para la empresa Avantica San Carlos, que permita integrar las funciones ofrecidas por el SDK de TestFlight en aplicaciones desarrolladas con PhoneGap.

Avantica es una empresa que, desde hace varios años, brinda servicios a clientes de dispositivos móviles, y últimamente se ha dado la necesidad de implementar aplicaciones híbridas para facilitar el desarrollo multiplataforma; con el desarrollo de grandes proyectos, también es importante controlar de mejor forma la manera en que los usuarios interactúan con los sistemas. Básicamente, este documento se compone de tres secciones principales, donde en la primera sección se encuentra la descripción del problema, en la segunda parte se puede encontrar la solución implementada y al final del documento se encuentran las conclusiones del proyecto según la experiencia adquirida.

En esta primera sección se exponen los motivos principales por los que surgió el proyecto, debido a la dificultad de poder implementar el SDK de TestFlight en aplicaciones híbridas desarrolladas con PhoneGap, que en ciertas ocasiones, hace difícil su depuración y control. Debido a esto, es que el proyecto consistió en una iniciativa que buscaba solucionar esta problemática presente dentro de la empresa, relacionadas al desarrollo de las aplicaciones móviles. Este sería un producto que iniciaba desde cero e incluso no tenía presente alguna solución actual, sino que, como estudiante de práctica de especialidad se debió realizar una propuesta de solución al mismo.

En la segunda sección se muestra, de manera detallada, la solución que se brindó a esta problemática y la estructura implementada para poder resolver el problema que se estaba presentando. Esto dada la necesidad de adaptar herramientas enfocadas para aplicaciones nativas a aplicaciones hibridas, por lo que como solución del proyecto, se desarrollaría un plugin cuyo objetivo principal fue simplificar y agilizar las tareas de pruebas y reproducción de errores, así como poder analizar el comportamiento que tenían los usuarios dentro de la aplicación.

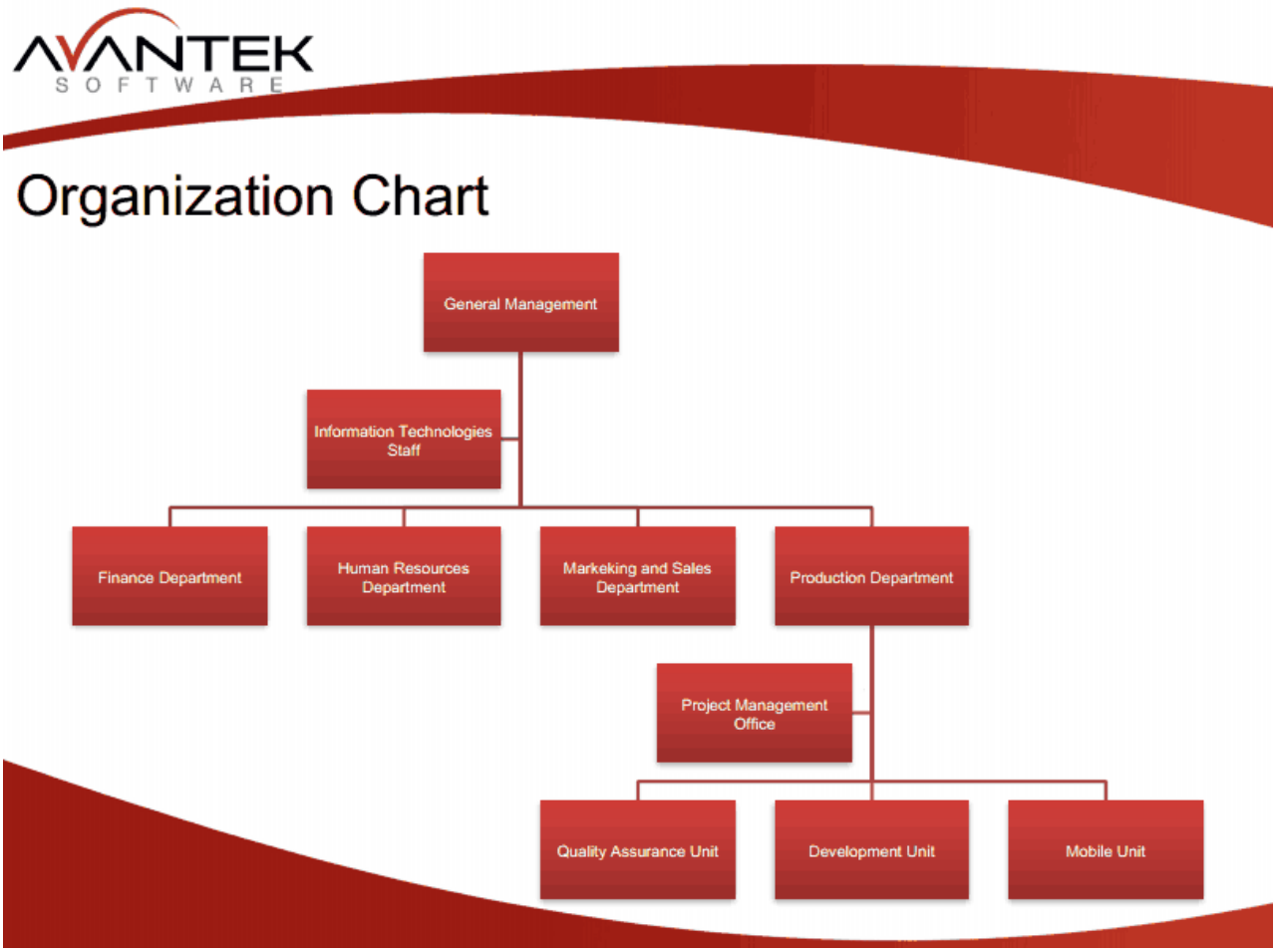
Por último, en la sección final se registra una síntesis de lo vivido durante las dieciséis semanas que forman parte de la práctica de especialidad, donde se encuentran experiencias y comentarios personales relacionado con el proyecto trabajado; además comentarios sobre la empresa, incluyendo los beneficios de haber realizado la práctica en Avantica San Carlos.

Palabras clave: PhoneGap, TestFlight, Android, iOS, Dispositivos Móviles.

Tabla de Contenidos

Descripción de la Empresa

Avantica Technologies fue fundada en 1993 con sede en Silicon Valley, California y un centro de Ingeniería de Software en San José, Costa Rica. Actualmente la empresa ha expandido sus centros de ingeniería por toda Costa Rica, en Perú y se encuentra en medio del mayor centro de servicios de analistas software cercano a una costa en Latinoamérica. Avantica ha sido rentable desde su comienzo y ha crecido un 30% anualmente desde 1995.



El nuevo desarrollo se ubicará en las áreas de Development Unit y Mobile Unit ya que este plugin está enfocado al desarrollo de aplicaciones móviles.

Descripción del problema

El proyecto surgió ante la dificultad de poder implementar el SDK de TestFlight en aplicaciones híbridas desarrolladas con PhoneGap, las que en ciertas ocasiones hace que sea difícil su depuración y control.

Debido a esto, es que el proyecto consistió en una iniciativa que buscaba solucionar esta problemática presente dentro de la empresa, relacionadas al desarrollo de las aplicaciones móviles. Este sería un producto que iniciaba desde cero e incluso no tenía presente alguna solución actual, sino que como estudiante de práctica de especialidad se debió realizar una propuesta de solución al mismo.

Es un gran número de proyectos los que se realizan en una empresa tan grande como lo es Avantica, así como la gran cantidad de clientes con los que cuenta y se relaciona durante todo el desarrollo de un proyecto, incluyendo la etapa final de pruebas por parte de estos clientes.

Precisamente en este punto es cuando se daban diferentes situaciones, porque el cliente utilizaba el sistema para validar que todo esté funcionando correctamente, pero cuando el cliente detectaba que algo no está bien, entonces enviaba la observación a la empresa para que se realice la corrección de este error. Sin embargo, en ocasiones lo que un cliente reportaba como error, realmente no era así ocasionando gran atraso en el desarrollo del producto.

Como lo fue este caso en el que, por parte de un cliente de Avantica, se reportó un error que encontraron al realizar pruebas de un proyecto, después de muchas pruebas y recursos invertidos, se determinó que ni siquiera era un error; ocasionando que todo el tiempo invertido en busca de este problema fuera en vano.

Por esto es que se optó por el uso de una herramienta como lo es TestFlight, para tratar de contrarrestar estas situaciones que se podrían presentar en proyectos futuros desarrollados en la empresa. TestFlight es una herramienta de desarrollo para ofrecer versiones beta de las aplicaciones a los clientes y usuarios que realizan pruebas de las mismas. Cuenta con un SDK nativo que permite al desarrollador obtener información de lo que hacen los usuarios dentro de la aplicación. Entre sus principales características están:

- Registro remoto: el registro remoto de TestFlight le permite ver todos los registros (logs) creados por los usuarios de manera remota.

- Puntos de control (Checkpoints): Cuando un tester hace algo relevante en la aplicación, se puede crear un punto de control, por ejemplo completar un nivel, añadir una tarea. Esto puede ser útil para asegurarse de que los usuarios están usando todas las partes de la aplicación, así como comprobar que las pruebas están siendo suficientemente profundas.

- Sesiones: Dan una idea de la cantidad de veces que la gente ha utilizado la aplicación, el tiempo que lo usaron, cuándo se utilizaron, así como información básica sobre el dispositivo (por ejemplo, el idioma, el modelo de dispositivo, sistema operativo, soporte, zona horaria, etc.).

- Informes de errores: Los informes de errores se envían de forma automática y con el registro remoto, los errores son visibles de manera remota, sin la intervención del usuario.

- Actualizaciones In-App: Cuando un usuario ejecuta una versión anterior de una aplicación y una nueva versión está disponible, se les pedirá que actualicen a la nueva versión dentro de la misma aplicación.

Estas funcionalidades eran importantes dentro de los proyectos implementados en Avantica, pero no podían ser totalmente implementadas en aplicaciones creadas con PhoneGap debido a que el SDK era compatible únicamente con un lenguaje nativo, en este caso iOS. Es por esto que era necesaria la creación de un plugin que permitiera poder utilizar de una manera sencilla y completamente funcionales las características presentes en el SDK de TestFlight.

Justificación

El aumento de dispositivos móviles a nivel mundial provoca cada día que los desarrolladores deban aprender nuevos lenguajes de programación, para poder mantenerse al día con el desarrollo de aplicaciones. Apple con iOS y Google con Android proveen sus diferentes y potentes kits de desarrollo, pero requieren una curva de aprendizaje que se convierte en un alto precio para los clientes. Es por eso que surgen soluciones intermedias como PhoneGap, que permite a personas con conocimientos en programación web, poder fácilmente desarrollar aplicaciones para dispositivos móviles.

Por lo que ha ido surgiendo la necesidad de lograr adaptar herramientas enfocadas para aplicaciones nativas a aplicaciones hibridas. Este plugin tenía como objetivo principal simplificar y agilizar las tareas de pruebas y reproducción de errores, así como poder analizar el comportamiento que tenían los usuarios dentro de la aplicación.

Por lo tanto, la solución a este problema se basó fundamentalmente en el diseño y desarrollo de un plugin para la empresa Avantica, el cual permitió integrar las funciones, ofrecidas por el SDK de la plataforma TestFlight, dentro de las aplicaciones desarrolladas en el ambiente de trabajo PhoneGap.

Características del proyecto

Características	Aporte
Desarrollar plugin para el ambiente de trabajo PhoneGap.	Se permitió que la plataforma TestFlight contemplara la tecnología PhoneGap.
Implementar el SDK de la plataforma TestFlight.	Se utilizó el kit de desarrollo para implementar características útiles brindadas por esta plataforma.
Implementación del Registro Remoto.	Se permitió ver los registros (logs) creados por los usuarios de manera remota.
Implementación de Puntos de Control.	Se pudo crear un punto de control, por ejemplo, completar un nivel, añadir una tarea cuando un tester hace algo relevante en la aplicación. Esto fue útil para asegurarse de que los usuarios estén usando todas las partes de la aplicación, también verificar que las pruebas estén siendo profundas.
Implementación de Sesiones.	Se da una idea de la cantidad de veces que los usuarios han utilizado la aplicación, el tiempo que lo usan, cuándo se utilizan, así como información básica sobre el dispositivo (por ejemplo, el idioma, el modelo de dispositivo, sistema operativo, soporte, zona horaria, etc).
Implementación de Informes de errores. (Tentativo)	Los informes de errores se envían de forma automática y con el registro remoto, los errores son visibles de manera remota, sin la intervención del usuario.
Implementación de Actualizaciones In-App. (Tentativo)	Cuando un usuario ejecuta una versión anterior de una aplicación y una nueva versión está disponible, se implementó la posibilidad de pedir pide que actualicen a la nueva versión dentro de la misma aplicación.
Guía de Instalación y Uso	Se redactó un manual de Instalación y Uso, así se brindó un fácil uso y comprensión para los desarrolladores al utilizar el plugin, cuando se desee implementar en alguno de los proyectos.

La descripción de los patrocinadores

Nombre	Puesto en la empresa	Experiencia	Relación con el proyecto	Objetivo
Rodrigo Vargas	Director de Desarrollo	10 + años	Administrador de Proyecto	• Fue el intermediario, cuando se requirió, entre el profesor asesor y la contraparte de la empresa.
Marco Salazar	Desarrollador		Líder del Proyecto	• Asignó y llevó un control de las tareas establecidas para el avance del proyecto. • Dio apoyo en cada una de las tareas realizadas según lo requerido. • Se encargó de la evaluación y comunicación entre las partes (Profesor asesor, Contraparte de la empresa, Estudiante).
Maickol Chinchilla	Administrador de Proyectos		Gerente del Proyecto	• Se encargó de supervisar y asesorar que el proyecto se realizara de la mejor manera.
Brian Corella Pérez	Estudiante Práctica Especialidad		Desarrollador	• Encargado de realizar la especificación del sistema. • Responsable de desarrollar el sistema. • Debió implementar el sistema.

Resumen de necesidades y expectativas

Necesidad	Prioridad	Problema	Solución Actual	Solución Propuesta
Desarrollar un plugin para el ambiente de trabajo PhoneGap.	Alta	Limitación al querer implementar PhoneGap en la plataforma TestFlight.	Ninguna	Desarrollar un plugin para el ambiente de trabajo PhoneGap, permitiendo que la plataforma TestFlight contemple esta tecnología.
Implementar el SDK de la plataforma TestFlight.	Alta	Limitación al querer implementar características de TestFlight en PhoneGap.	Ninguna	Utilizar el kit de desarrollo para implementar características útiles brindadas por esta plataforma, mediante el plugin PhoneGap.
Implementación del Registro Remoto.	Alta	Limitación al querer analizar los pasos que realiza un usuario cuando se registra.	Ninguna	Utilizar el SDK para ver los registros (logs) creados por los usuarios de manera remota.
Implementación de Puntos de Control.	Alta	Limitación al querer analizar el estado en que se encuentra una aplicación después de que el usuario haya accedido a un punto específico.	Ninguna	Utilizar el SDK para ver cuándo un tester hace algo relevante en la aplicación, creando un puesto de control, por ejemplo, completar un nivel, añadir una tarea.
Implementación de Sesiones.	Alta	Limitación al querer analizar la cantidad de veces, la duración, cuándo y la información del dispositivo con que algún usuario ha hecho uso la aplicación.	Ninguna	Utilizar el SDK para tener una idea de la cantidad de veces que la gente ha utilizado la aplicación, el tiempo que lo usaron, cuando se utilizaron, así como información básica sobre el dispositivo

Implementación de Informes de errores. (Tentativo)	Baja	Limitación al querer analizar los informes de errores ya que algunos no son reportados.	Ninguna	Utilizar el SDK para poder enviar los informes de errores de forma automática y con el registro remoto, los errores serían visibles de manera remota, sin la intervención del usuario.
Implementación de Actualizaciones In-App. (Tentativo)	Baja	Limitación al querer actualizar de manera automática la versión actual de la aplicación si se encuentra una más reciente.	Ninguna	Utilizar el SDK para que cuando un usuario ejecuta una versión anterior de una aplicación y una nueva versión está disponible, se pedir que actualice a la nueva versión dentro de la misma aplicación.
Guía de Instalación y Uso	Alta	Dificultad de comprensión y uso de una herramienta nueva sin su debida documentación y guía.	Ninguna	Elaborar un manual de instalación y de uso para que sea de fácil uso y comprensión, para los desarrolladores, al utilizar el plugin cuando se vaya a implementar en alguno de los proyectos.

Requerimientos del Proyecto

A continuación se detallará el resultado del proceso de levantamiento de requerimientos, de esta manera se pudo determinar cómo se realizaba el proceso en la actualidad y se establecieron los requerimientos que se tomarían en cuenta para el desarrollo del proyecto.

Requerimientos Funcionales

1. Se Implementó el diseño y especificación de un plugin iOS para PhoneGap que permitía utilizar las características ofrecidas por la herramienta de desarrollo TestFlight.

    1.1. Registro remoto: permitió ver todos los registros (logs) creados por los usuarios de manera remota.

    1.2. Puntos de control: que permitió, cuando un tester hace algo relevante en la aplicación, poder crear un puesto de control, por ejemplo, completar un nivel, añadir una tarea. Esto fue útil para asegurarse de que los usuarios estaban usando todas las partes de la aplicación, así como ver que las pruebas eran lo suficientemente profundas.

    1.3. Sesiones: dio una idea de la cantidad de veces que la gente había utilizado la aplicación, el tiempo que lo usaron, cuando se utilizaron, así como información básica sobre el dispositivo (por ejemplo, el idioma, el modelo de dispositivo, sistema operativo, soporte, zona horaria, etc.).

2. Se Implementó el diseño y especificación para el SDK de TestFlight en una aplicación desarrollada con PhoneGap.

3. Se validó el lenguaje en que se está desarrollando el proyecto.

    3.1. Se determinó el desarrolló en iOS.

    3.2. Se determinaron los cambios respectivos dentro del plugin (trabajando con el “console.log”).

4. Se implementó la propuesta de software como un modulo independiente.
    4.1. Plugin debió ser implementado como un modulo.
    4.2. Se debió poder utilizar para múltiples proyectos.
5. Se desarrolló un producto completo.
    5.1. El sistema fue implantado por los desarrolladores.
    5.2. El sistema pudo ser utilizado después de su implantación.

Requerimientos No Funcionales

1. Este plugin debió cumplir con las reglas y estándares de programación en el proceso de desarrollo.
    1.1. Requerimientos de calidad del software (usuario)
        1.1.1. Interoperabilidad.
        1.1.2. Confiabilidad.
        1.1.3. Robustez.
        1.1.4. Usabilidad.
        1.1.5. “Amigable al usuario”.
        1.1.6. Instalación
    1.2. Requerimientos de calidad del software (desarrollador)
        1.2.1. Mantenibilidad.
        1.2.2. Estándares de documentación.
        1.2.3. Indentación.
        1.2.4. Metodología de diseño.
        1.2.5. Estructura de directorios.
        1.2.6. Documentos de diseño.
        1.2.7. Portabilidad.
        1.2.8. Reusabilidad.
        1.2.9. Facilitar pruebas.

2. El plugin elaborado debió contar con su documentación interna de manera completa.

    2.1. Todas las funciones, nombres de variables, métodos y el código en general fue redactado en inglés.

    2.2. Se debió indicar el funcionamiento y la composición para facilitar su mantenimiento.

3. Se desarrolló un manual de instalación.

    3.1. Se especificaron los procesos necesarios para implementar el plugin dentro de algún proyecto.

    3.2. Se especificaron los requerimientos necesarios para implementar el plugin dentro de algún proyecto.

4. Se debió desarrollar un manual de usuario.

    4.1. Se especificó cada uno de los métodos implementados.

    4.2. Se especificó cuáles son los parámetros que recibe.

    4.3. Se especificó cuáles son las formas en que se inicializan.

    4.4. Se especificó cualquier otra información relevante, de acuerdo a las funciones que se deban utilizar en el desarrollo del plugin.

5. Este proyecto se debió trabajar y mantener dentro de un repositorio que se encuentra dentro de la empresa.

6. El plugin debió ser sencillo de instalar y fácil de mantener, el cual puede ser implementado en las aplicaciones híbridas que la empresa desarrolle.

Análisis de los riesgos

Nombre o Descripción	Categoría	Causa	Impacto (I)*	Probabilidad (P)*	Exposición (I*P)	Estrategia Evasión
Plugins en cordova no fueran compatibles	Tecnológico	Que la manera de implementar los plugins cambiara presentando dificultades de compatibilidad	5	2	10	Realizar el plugin de manera genérica para que siempre sea compatible
Estrategia Mitigación			**Estrategia Contingencia**			
Realizar una documentación profunda sobre el uso del plugin, para que sea sencillo un posible cambio en el plugin			Aplicar los cambios necesarios sobre el plugin para adaptarlo a la nueva forma en que se implementan los plugins.			

* Del 1 al 5 donde 1 representa el valor más bajo y 5 el valor más alto.

Nombre o Descripción	Categoría	Causa	Impacto (I)*	Probabilidad (P)*	Exposición (I*P)	Estrategia Evasión
Plugin no funcionara con versiones recientes del SDK	Tecnológico	Un nuevo SDK tuviera características distintas a las que se implementaron en el plugin	4	2	8	Uso de las características más importantes que brinde el SDK
Estrategia Mitigación			**Estrategia Contingencia**			
Realizar una documentación profunda sobre las características implementadas, para que sea sencillo determinar			Aplicar los cambios para implementar las nuevas características del SDK que sean relevantes para la funcionalidad del plugin			

* Del 1 al 5 donde 1 representa el valor más bajo y 5 el valor más alto.

Nombre o Descripción	Categoría	Causa	Impacto (I)*	Probabilidad (P)*	Exposición (I*P)	Estrategia Evasión
Posible cierre de la plataforma TestFlight	Tecnológico	Que no se brindara continuidad en la plataforma y sea cerrada	5	1	5	Dar seguimiento sobre la continuidad de la plataforma.
Estrategia Mitigación			**Estrategia Contingencia**			
Realizar una documentación profunda sobre el uso del plugin, para que sea sencillo de implementar en otra plataforma			Aplicar los cambios necesarios sobre el plugin para adaptarlo a la nueva plataforma en que se estaría implementando			

* Del 1 al 5 donde 1 representa el valor más bajo y 5 el valor más alto.

Objetivos y alcances del sistema

1. Objetivo general

Desarrollar un plugin que permita utilizar las características de TestFlight en aplicaciones desarrolladas con PhoneGap.

2. Objetivos específicos

- Generar un plugin para PhoneGap iOS.
- Implementar el SDK de TestFlight en una aplicación desarrollada con PhoneGap.
- Generar una guía de instalación y uso.

Productos esperados

Objetivo Específico	Productos	Prioridad
Se buscó generar un plugin para PhoneGap en iOS.	Un plugin para PhoneGap.	Alta
Se buscó implementar el SDK de TestFlight en una aplicación desarrollada con PhoneGap.	Una aplicación desarrollada en PhoneGap implementando el plugin creado.	Alta
Se necesitó crear una guía de instalación y uso.	Un documento en el que se muestren los pasos de instalación y uso de la herramienta.	Alta

En términos generales se esperó que el plugin sea sencillo de instalar y que fuera fácil de mantener, el cual puede ser implementado en las aplicaciones hibridas que la empresa desarrolle. Es decir, se buscaba un producto completo, un sistema que pudiera ser implantado y utilizado al terminar la práctica.

Implementación del plugin

La implementación de este plugin se realizó tomando como base un proyecto donde se utilice PhoneGap; se debió configurar cada uno de estos archivos:

- Se agregó el SDK de TestFlight al proyecto.
- Se creó una clase JavaScript que consiste en el plugin implementado.
- Se creó una clase nativa del lenguaje en que se implementó PhoneGap.
- Se configuró el proyecto para hacer el uso del plugin.

Una vez que se agregó el SDK de TestFlight al proyecto, se procedió a crear una clase JavaScript que funciona como interface para comunicar el proyecto de PhoneGap con el SDK nativo, que en este caso se implementó para iOS. Consiste en el plugin conteniendo las funciones que reflejen la lógica expuesta por el código nativo y así permite la comunicación entre PhoneGap y el SDK.

Se tomará como ejemplo el plugin implementado para simplificar la lectura y entendimiento de la implementación dada, contiene el siguiente formato:

```
var Plugin = {
call NativeFunction: function (success, fail,
resultType) {
return Cordova.exec( success, fail,
                    "serviceName",
                    "nativeFunction",
                    [resultType]);
    }
};
```

La comunicación se realizó mediante la función de Cordova.exec, cuando ésta es invocada necesita los siguientes parámetros:

1. Una referencia a una función callback de éxito (función invocada cuando la respuesta desde el código nativo fue exitosa).
2. Una referencia a una función callback de error (función invocada cuando la respuesta desde el código nativo fue errónea).
3. Una referencia String con la clase del código nativo.
4. Una referencia String con el nombre de la función que debe ser llamada en el código nativo.
5. Un arreglo de parámetros para ser pasados al código nativo y que son utilizados en la función llamada.

Una vez que se configuró el SDK en la clase nativa, el plugin ya puede ser llamado dentro de la aplicación PhoneGap de la siguiente manera:

1. En primer lugar, se agregó una referencia al plugin creado con la clase JavaScript; mediante un tag <script> en el documento *html*.

```
<script type="text/javascript" charset="utf8"
src="AV.TestFlight.js"></script>
```

2. Luego se agregó el archivo JavaScript para invocar al plugin, así llamar la función del código nativo y recibir los resultados.

```
//Funciones para llamar el código nativo de TestFlight
//Para ver los logs que el app muestra remotamente
function callRemoteLogging( logValue ) {
    TestFlight.remoteLogging( nativePluginResultHandler,
    nativePluginErrorHandler, logValue );
}
//-------------Funciones por defecto-------------
//Función de éxito
function nativePluginResultHandler (result) {
    alert("SUCCESS: \r\n"+result );
}
//Función de error
function nativePluginErrorHandler (error) {
    alert("ERROR: \r\n"+error );
}
```

3. Finalmente, las funciones ya podían ser llamadas desde cualquier objeto o sección de la aplicación PhoneGap para así invocar al plugin creado.

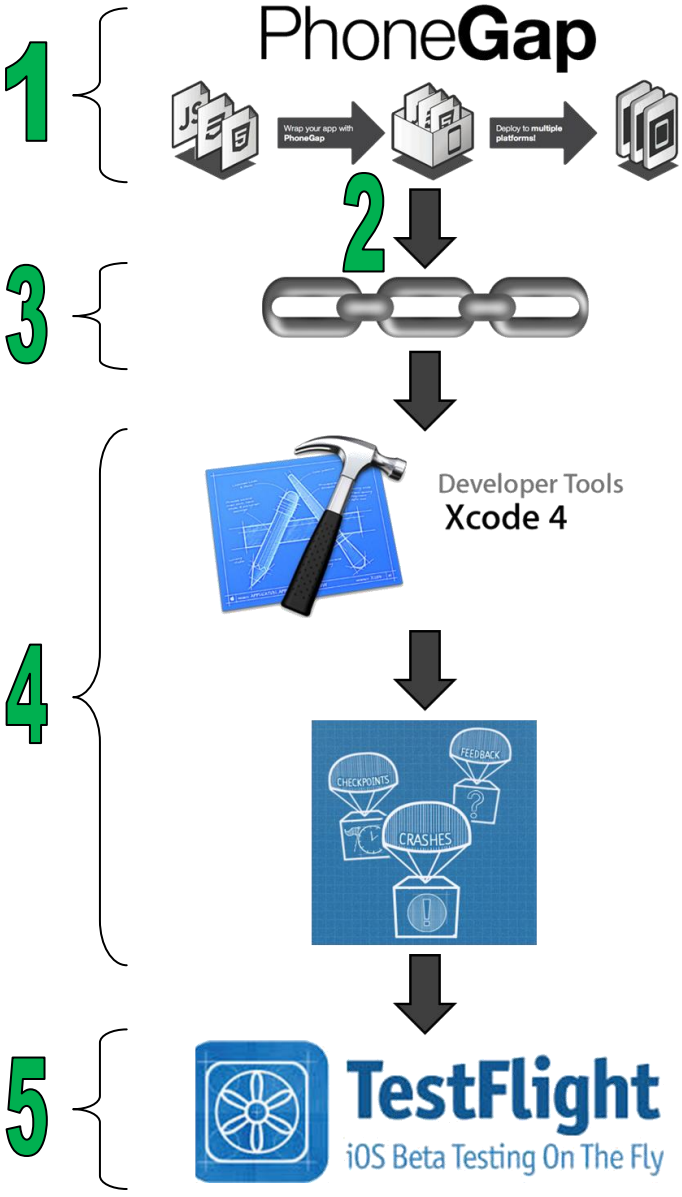
```
var serviceName = "TestFlight";
//Contiene todas las funciones para llamar el código nativo iOS
varTestFlight = {
/*Utilizado cuando el usiario envía un console.log.
  @param   successCallback function
  @param   failureCallback function
  @param   newLog string */
remoteLogging: function(successCallback, failureCallback, newLog)
{   Cordova.exec(successCallback,  failureCallback,  serviceName,
    "remoteLogging", [{newLog: newLog}]);
}
```

De esta manera es como se implementó el plugin; tanto a la hora de ser creado como la forma en que se le dio uso al mismo dentro del proyecto trabajado.

Modelo de Diseño

Arquitectura conceptual de la solución, componentes y servicios, Diagrama de clases

A continuación se representarán los diferentes componentes que se implementaron en el proyecto; básicamente las características correspondientes a las tres funcionalidades más importantes que se pidieron: Registro Remoto, Puntos de Control, Sesiones; así como la interacción del usuario (desarrollador) con el sistema implementado (plugin), el lenguaje nativo (iOS) y la plataforma TestFlight que posibilita la utilización de estas funcionalidades mencionadas anteriormente.

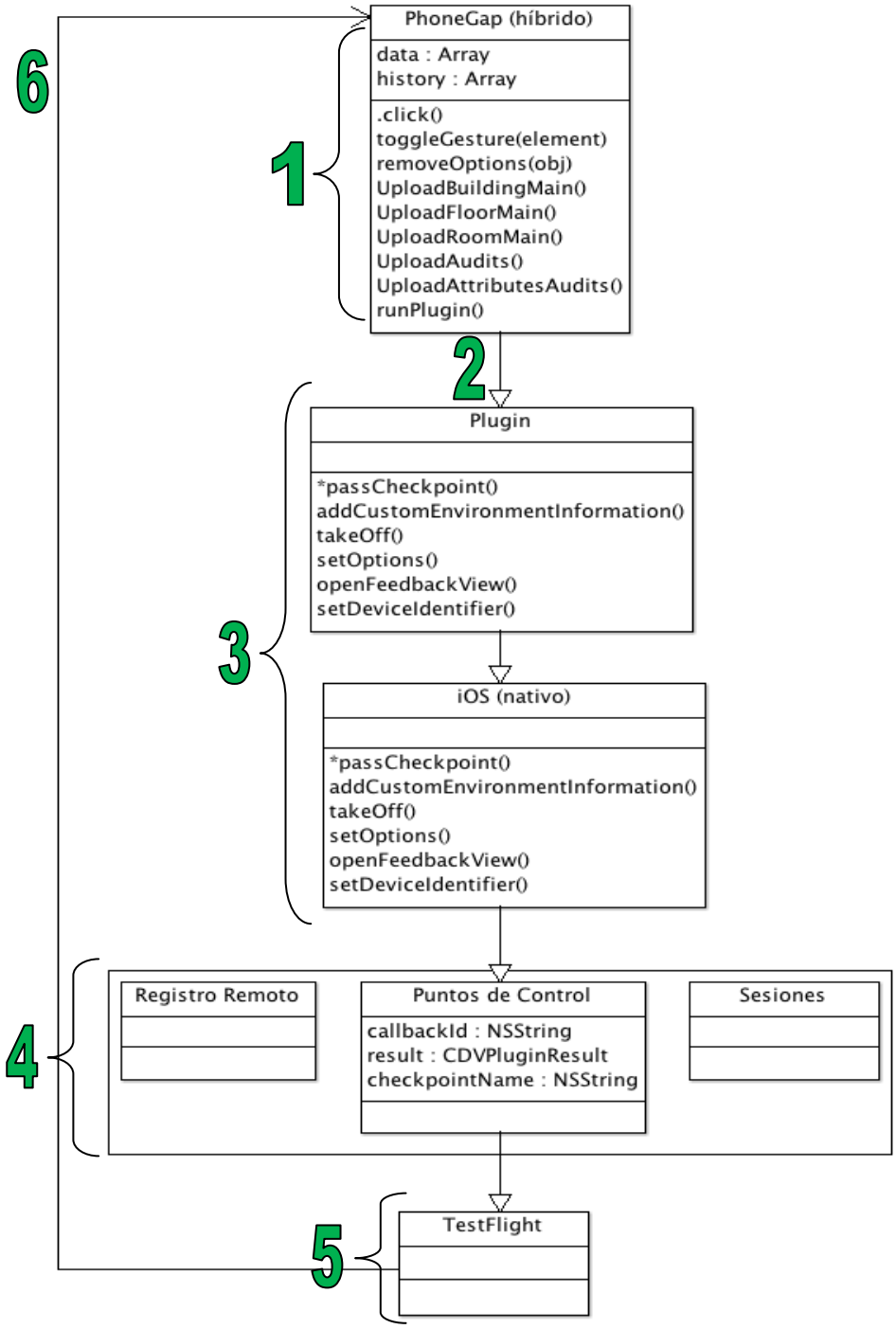


El flujo básico de ejecución dentro de la aplicación sería como el siguiente:

1. Se inicia con la aplicación en PhoneGap, donde el desarrollador llama a la característica de acuerdo a la funcionalidad que se desee aplicar.
2. El sistema recibe la llamada desde la aplicación híbrida (PhoneGap), donde se comunica con el plugin desarrollado, encargado de comunicar la parte híbrida con la nativa.
3. El plugin recibe la solicitud enviada por el desarrollador a través del sistema, después determina a cuál funcionalidad es a la que se está haciendo referencia para así poder realizar la implementación y utilizar el lenguaje nativo de iOS.
4. Luego de comunicarse de manera nativa, el sistema utiliza la característica correspondiente brindado por el SDK de TestFlight, así responder a la solicitud enviada inicialmente por el desarrollador.
5. Cuando el SDK de TestFlight realiza la solicitud pedida por el desarrollador; se envía un informe de los resultados obtenidos mediante el sistema.

Finalmente el desarrollador recibe este informe brindado para así poder controlar la interacción de los usuarios con el proyecto correspondiente.

El siguiente diagrama muestra la arquitectura conceptual de la solución mostrada anteriormente, consiste en un diagrama de clases en donde se pueden visualizar los distintos componentes a implementar, así como la interacción entre cada uno de ellos:



La interacción entre cada uno de los componentes se puede resumir de la siguiente forma:

1. La aplicación PhoneGap contiene todos los métodos, variables y demás implementación necesaria para ejecutar la aplicación a la que se le desee realizar las pruebas mediante el SDK.
2. La comunicación entre un componente y otro se representa mediante una flecha descendente, en este caso representa la llamada que se realiza desde la aplicación desarrollada para comunicarse, mediante el plugin desarrollado, con el SDK de TestFlight.
3. El plugin se encarga de entender cuál característica es la que se desea implementar, y así realizar esta llamada en el lenguaje nativo de iOS. Ya que el SDK está implementado para este lenguaje.
4. Luego de comunicarse de manera nativa, el sistema implementa la característica correspondiente del SDK de TestFlight, así responde a la solicitud enviada por el desarrollador.
5. Cuando el SDK de TestFlight realiza la solicitud pedida; se envía un informe de los resultados obtenidos, los mismos son almacenados automáticamente dentro de la página de TestFlight.
6. Finalmente se representa mediante una flecha que la característica del SDK se ha implementado de manera exitosa y regresa nuevamente el ciclo a la aplicación de PhoneGap para poder implementar nuevamente el SDK.

Característica	Aporte
Implementación del Registro Remoto.	Permite ver los registros (logs) creados por los usuarios de manera remota.
Implementación de Puntos de Control.	Cuando un tester hace algo relevante en la aplicación, se puede crear un puesto de control, por ejemplo, completar un nivel, añadir una tarea. Esto puede ser útil para asegurarse de que los usuarios están usando todas las partes de la aplicación, así como que las pruebas están profundas.
Implementación de Sesiones.	Da una idea de la cantidad de veces que la gente ha utilizado la aplicación, el tiempo que lo usaron, cuando se utilizaron, así como información básica sobre el dispositivo (por ejemplo, el idioma, el modelo de dispositivo, sistema operativo, soporte, zona horaria, etc).
Implementación de Informes de errores. (Tentativo)	Los informes de errores son enviados de forma automática y con el registro remoto, los errores son visibles de manera remota, sin la intervención del usuario.
Implementación de Actualizaciones In-App. (Tentativo)	Cuando un usuario ejecuta una versión anterior de una aplicación y una nueva está disponible, se le pide que actualicen a la nueva versión dentro de la misma aplicación.
Guía de Instalación y Uso	Facilita el uso y comprensión para los desarrolladores al utilizar el plugin cuando se vaya a implementar en alguno de los proyectos.

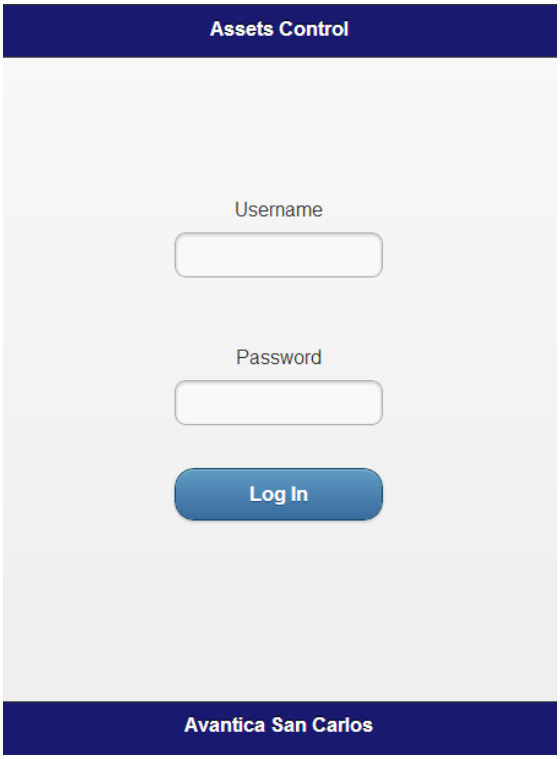
Interfaces de Usuario

Debido a que este proyecto consistió en la creación de un plugin, éste no contenía su propia interfaz gráfica de manera directa; sin embargo, se debió desarrollar un proyecto como prototipo para realizar las pruebas de funcionalidad del plugin.

Este programa desarrollado, consistió en un control de activos implementado para la plataforma PhoneGap que al ser híbrido se desarrolla de la misma manera que plataformas de internet mediante el uso de archivos html, CSS y JavaScript.

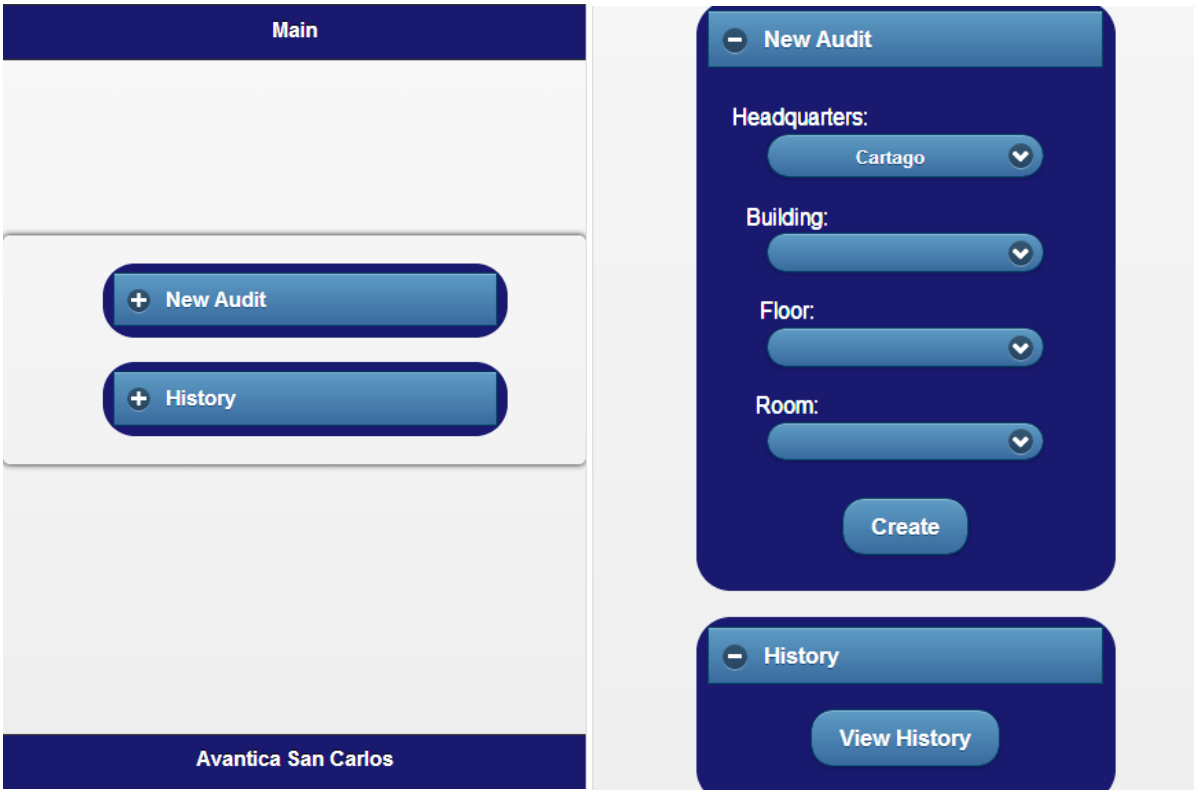
Consta de 4 vistas principales que se detallarán a continuación:

Vista de inicio



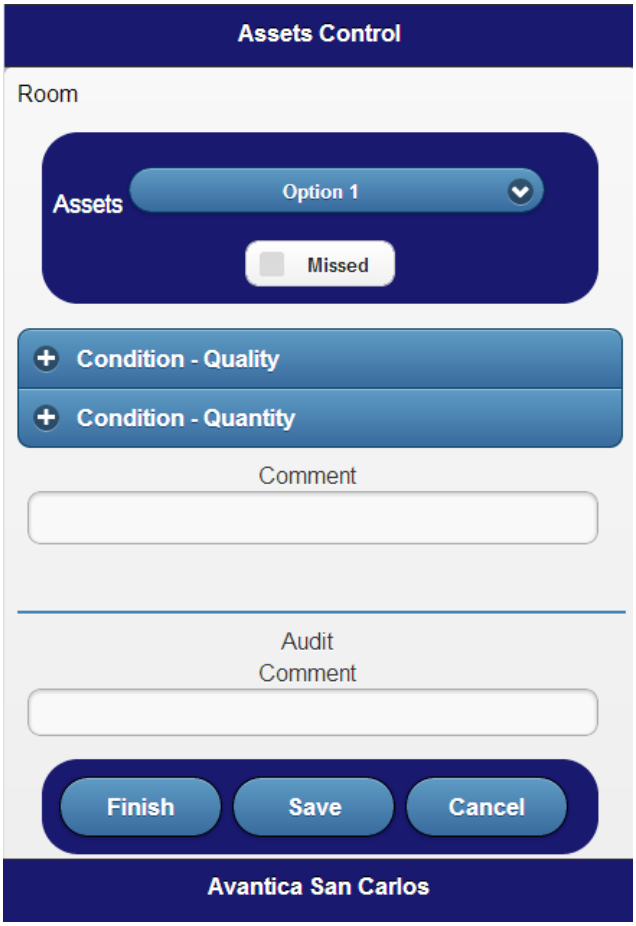
En cuanto a este punto de vista, el usuario tenía que escribir su nombre y contraseña para iniciar sesión en la aplicación; esto se realiza con una vista en la que el usuario puede ver dos objetos que contiene los componentes necesarios para introducir el nombre y la contraseña. Cuando esta información es escrita, el usuario puede hacer clic en un botón de “Log In” para iniciar sesión en la aplicación; cada usuario puede iniciar sesión en la aplicación sin ninguna validación ya que únicamente se buscaba una aplicación funcional para realizar las pruebas.

Vista principal



El usuario puede tener el acceso para ver la situación de todas las auditorías mediante la opción de Historial; o el acceso a crear una nueva auditoría para una habitación específica mediante la opción de Auditoría Nueva, en donde se puede elegir la sede, edificio y piso.

Vista de auditorías

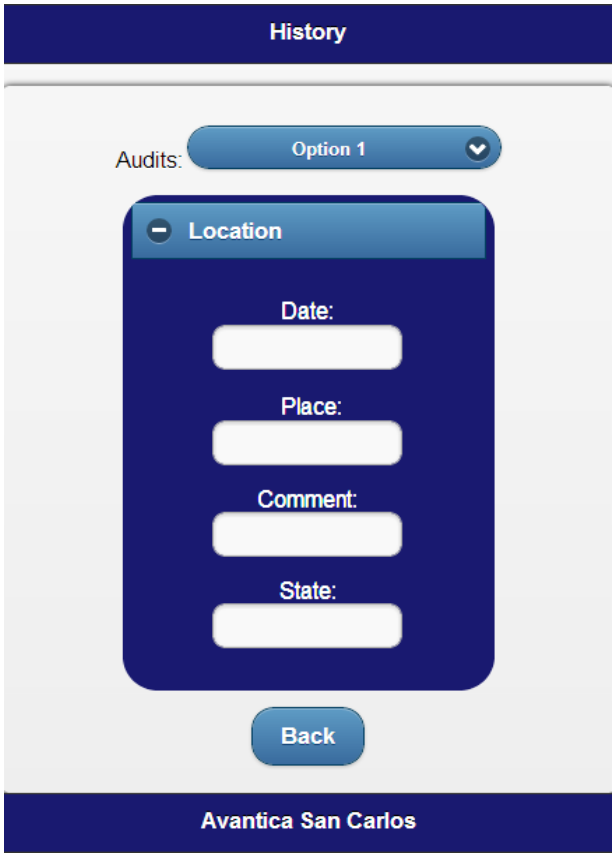


El usuario puede ver todos los activos que ya están preparados para la auditoría en una habitación específica, así ser evaluado de acuerdo a algunas características.

También es posible indicar si el activo está en la habitación y escribir comentarios en los diferentes activos o bien en la auditoría en general; cuando la información está escrita, el usuario puede elegir entre finalizar, guardar o cancelar la auditoría.

Después de que el usuario puede ver un objeto, que contiene la opción de evaluar las condiciones de cada uno de los activos; es posible realizar una evaluación de acuerdo a la calidad o cantidad. Por último es posible agregar comentarios específicos para cada activo o un comentario general de la auditoría.

Vista historial



Se permite al usuario ver todos los activos que no están listos para la auditoría, mediante la fecha, la sede central, edificio, piso, habitación y el estado. Cuando una auditoría completa se selecciona, se muestra toda la información de la auditoría pero los valores no pueden ser modificados, sólo si la auditoría seleccionada no está todavía completada, entonces el usuario puede modificarla.

La implementación del SDK se puede realizar en cualquier punto de la aplicación, incluso hay algunas características que se pueden implementar de manera programada sin necesidad de que haya interacción del usuario, mientras que otras se pueden colocar en cualquier evento dentro de la aplicación para así determinar, por ejemplo, si el usuario ha pasado por algún punto específico como lo son los puntos de control, posibles de implementar gracias al SDK de TestFlight.

En este proyecto, los eventos que se utilizaron son los botones de “finalizar”, “guardar” y “cancelar” que se encuentran en la vista de auditorías, cada uno de ellos se utilizó como prueba para la implementación de algunas características del SDK, en donde únicamente se debe presionar el botón y el sistema automáticamente realizará las llamadas del plugin siguiendo el ciclo que se mencionó anteriormente.

Conclusiones y comentarios

El uso de plataformas como TestFlight permite la distribución de aplicaciones, tanto de Android como iOS beta e internas, a miembros de un grupo establecido. Una vez que se han distribuido aplicaciones mediante esta plataforma, los desarrolladores pueden administrar las pruebas y recibir información valiosa sobre la aplicación y las condiciones en que se está utilizando, esto gracias a características que ofrece TestFlight con su SDK.

Tener conocimientos sobre la posibilidad de utilizar una plataforma como PhoneGap, posibilita a los desarrolladores crear aplicaciones para dispositivos móviles mediante herramientas como JavaScript, HTML5 y CSS3; en lugar de lenguajes nativos de desarrollo para móviles, esto simplifica en gran medida la posibilidad de implementar la aplicación de manera multiplataforma, ya que, gracias a PhoneGap, el desarrollo es el mismo para cualquier lenguaje móvil que acepte trabajar con esta plataforma, como por ejemplo Android y iOS.

El desarrollo de plugins como complemento en un programa ayuda a simplificar el trabajo al programador, ya que de esta manera se puede ejecutar una tarea de manera más práctica, así como dar solución a problemas específicos como el de poder implementar la comunicación entre un lenguaje nativo con un lenguaje híbrido.

Cuando se desarrollan proyectos dentro de un grupo de dos o más personas, hacer uso de repositorios es de suma importancia para poder controlar las versiones del proyecto en que se está trabajando; ya que cada integrante puede trabajar en distintas tareas del proyecto y luego, gracias al repositorio, se puede integrar cada uno de estos desarrollos a la versión final de manera automática, para así avanzar más rápido el desarrollo.

Glosario

TestFlight: TestFlight[2] es una de plataforma que se utiliza para la distribución de aplicaciones de iOS beta e internas a miembros del grupo. Después, los desarrolladores pueden administrar las pruebas y recibir información de su equipo gracias a características que ofrece TestFlight[3].

PhoneGap[4]: PhoneGap[5] es una plataforma de desarrollo móvil que permite a los programadores de software crear aplicaciones para dispositivos móviles mediante herramientas como JavaScript, HTML5, CSS3, en lugar de lenguajes específicos de dispositivos como Objective-C.

iOS: Es un sistema operativo móvil desarrollado y distribuido por Apple Inc.

Android: Android es un sistema operativo basado en Linux diseñado principalmente para dispositivos móviles como smartphones y tablet pc. Desarrollado inicialmente por Android, donde luego Google Inc. Brindó un apoyo económico y posteriormente lo compró siendo el actual dueño.

SDK: Kit de desarrollo de software; consiste en un conjunto de herramientas de desarrollo de software que permite la creación de aplicaciones para un cierto paquete de software, framework, plataforma de hardware, sistema informático, consola de video juegos, el sistema operativo o plataforma de desarrollo similares.

Plugin: Es un componente de software que añade una característica específica de una aplicación de software existente permitiendo una personalización.

JavaScript: es un lenguaje de programación interpretado. Como parte de los navegadores de la web, permite implementaciones de secuencias de comandos del lado del cliente para interactuar con el usuario, también para controlar el navegador, comunicarse asincrónicamente, y modificar el contenido dela información mostrada.

HTML5: Es un lenguaje de marcado que se utiliza para estructurar y presentar el contenido de la World Wide Web y la tecnología del internet.

CSS3: Es un lenguaje de hoja de estilo utilizado para describir la presentación semántica, el aspecto y el formato de un documento escrito en un lenguaje de marcado.

Objective-C: Es el principal lenguaje de programación utilizado por apple para el os x y el iOS sistemas operativos.

Referencias Bibliográficas

[1] Esquivel, Gaudy (2013). *Presentación de un proyecto-PRACTICA-ESPECIALIDAD-COMPUTACIÓN*. Consultado en junio 10, 2013 en https://www.dropbox.com/s/8xg6brrup3ewp8q/Presentaci%C3%B3n%20de%20un%20proyecto-PRACTICA-ESPECIALIDAD-COMPUTACI%C3%93N.doc

[2] TestFlight. *TestFlight. Beta Testing On The Fly, How it works*. Consultado en julio 26, 2013 en https://testflightapp.com/

[3] TestFlight. *TestFlight. Beta Testing On The Fly, Features*. Consultado en julio 26, 2013 en https://testflightapp.com/sdk/

[4] PhoneGap. *Easily create apps using the web technologies you know and love: HTML, CSS, and JavaScript*. Consultado en julio 26, 2013 en http://phonegap.com/

[5] PhoneGap. *Developer Portal*. Consultado en julio 26, 2013 en http://phonegap.com/developer/

[6] Wikipedia. *Conceptos varios.* Consultado en agosto 28, 2013 enhttp://en.wikipedia.org/wiki/Main_Page