Instituto Tecnológico de Costa Rica

Sede Regional San Carlos

Carrera de Ingeniería en Computación

Mobile Testing Framework: Una plataforma de Automatización de pruebas en dispositivos móviles.

Informe 3 de Práctica de especialidad para optar por el Título de Ingeniería en Computación, con el grado académico de Bachillerato

Uriel Vindas Agüero
200736900

San Carlos, Junio 2013

Resumen Ejecutivo

Hoy en día el proceso de automatización de pruebas unitarias, está tomando un papel muy importante en el control de calidad de software, ahorran tiempo, esfuerzo y disminuye considerablemente la ocurrencia de algún error debido a las pruebas manuales. Esto hace que Avantica Technologies demuestre interés en desarrollar un framework que permita llevar a cabo el proceso de automatización para casos de prueba en aplicaciones nativas en dispositivos móviles.

Para llevar a cabo dicha herramienta se desarrolló una investigación previa sobre algunas herramientas de automatización existen para nuestro objetivo. Con base a la investigación se decidió trabajar con MonkeyTalk. Esta es una herramienta de libre uso (open source), para pruebas funcionales en aplicaciones Mobile desarrolladas los entornos Android y iOS.

Dicho framework se desarrolló utilizando Java conjuntamente con Javascreipt como lenguaje de programación y Eclipse como entorno de desarrollo, además de incluir librerías (monkeytalk-runner.jar y MonkeyTalk.API) que nos permiten aprovechar tanto las funcionalidades de automatización así como las de ejecución de casos de prueba.

Durante el desarrollo del proyecto fue necesario realizar algunos cambios en la metodología y por ende se tomaron las medidas apropiadas basados en los riesgos que se plantearon.

Tabla de contenido

Capítulo 1 Descripción del problema

Contexto del proyecto.

La empresa

El proyecto se llevó a cabo en la empresa llamada Avantica Technologies, ubicada justamente en la sede San Carlos, Alajuela, Costa Rica. El proyecto será desarrollado propiamente en área de aseguramiento de la calidad (QA).

Antecedentes del proyecto

Anteriormente no se han realizado grandes trabajos sobre este proyecto, se conocen las herramientas existentes para realizar lo requerido pero solo se han probado un par de ellas, por lo cual se califica como un proyecto nuevo y con poca experiencia.

Descripción del Problema

La situación que se presenta es en el momento de realizar pruebas de aseguramiento de calidad en las aplicaciones móviles. Como bien se conoce, el control de calidad en un proyecto de software es de vital importancia y es la última validación y verificación antes de la entrega al cliente. Este proceso está sujeto a cambios, resolución de pequeños errores por ejemplo y realizar constantemente las mismas pruebas requiere mucho tiempo. Hoy en día existen muchas herramientas que permiten realizar de forma automática casos de pruebas para aplicaciones en dispositivos móviles.

El proyecto consiste en desarrollar primeramente una investigación sobre las opciones o herramientas disponibles para la automatización, para luego poder identificar y definir la que más sea de conveniencia y de esta forma desarrollar una plataforma de automatización, la cual se convertirá en parte del framework de automatización de Avantica en dispositivos móviles para aplicaciones nativas.

El proyecto viene a mejorar la forma en cómo se realizan los servicios de pruebas a nivel de dispositivos móvil dentro de la empresa, ya que este framework traerá beneficios de eficiencia al automatizar scripts de casos de pruebas y reducir considerablemente los tiempos de trabajo. Los participantes en el proyecto se describen a continuación:

Tabla 1.1 Stackeholders

Responsable	Rol	Relación con el producto	% Asignación
Uriel Vindas Agüero	Practicante	Encargado del desarrollo del proyecto.	% 100
Mario Núñez	Líder de QA	Seguimiento al proyecto	% 5
Maickol Chinchilla.	Encargado de Avantica sobre las practicas	Supervisar el desarrollo del proyecto	% 5

Cabe destacar como nota aclarativa, que el cliente potencial de este proyecto es la misma empresa, Avantica Technologies.

Justificación

La diferencia entre este producto y los que ya existen radica en que va ser el propio framework para la empresa. En general sus usuarios potenciales serán el equipo especializado en automatización. Ayudará a agilizar el proceso de aseguramiento de calidad de aplicaciones móviles nativas, a su vez que va a contribuir a inducir conocimiento sobre herramientas poco comunes como lo son MonkeyTalkl, MonkeyRunner, UIAutomation, Calabash, y a su vez mejorar los conocimientos en Java, Junit, etc.

Riesgos

Riesgos ocurridos y de más importancia.

1- Modificaciones inesperadas y que son deseadas por parte del cliente del proyecto.

- **Tipo alcance.**
- **Impacto.**

a) Este riesgo genera un impacto en la estipulación y cumplimiento del cronograma establecido.

b) Genera un a nivel de tareas, ya que se debe rediseñar labores, y mantener el orden para no alterar mucho las fechas.

- **Probabilidad de ocurrencia.**

10% resultado basado en el juicio de experto y experiencias vividas en proyectos anteriores.

- **Estrategia de mitigación.**

Para evitar este tipo de inconvenientes lo mejor será tener una semana de prevención, ante estos eventos, puede que la modificación no sea muy grande, pero en otros casos si, e incluso se tenga que reiniciar trabajo que se necesitó varias horas.

- **Estrategia de Concurrencia.**

En caso de que esto ocurra, se modifica un poco el cronograma, y de ser necesario se trabaja un poco fuera de las horas establecidas.

2- Retraso en algún requerimiento del proyecto

- Personas**:**

- **Impacto.**

a) Este riesgo afecta el cumplimiento de los requerimientos de la aplicación.

b) Ausencia de desarrollo de algún requerimiento importante, a causa de no poder implementarlo en la construcción del proyecto.

- **Probabilidad de ocurrencia.**

10% resultado basado en el juicio de experto, basado en la revisión sobre el estado actual de nuestros conocimientos.

- **Estrategia de mitigación.**

Analizar las distintas maneras de poder realizar lo que se solicita, esto mediante la investigación, estudio y pequeñas pruebas basadas en alguna experiencia tenida anteriormente relacionada con lo que se quiere realizar. Si algo no es solucionado en el tiempo establecido, se recomienda hacérselo saber lo más pronto posible al supervisor para tomar las medidas necesarias

- **Estrategia de Concurrencia.**

En caso de que esto ocurra, se optará por recurrir a personas que nos guíen o nos den ideas de cómo se debe realizar, y como implementarlo de una mejor manera.

3- Cambios inesperados en la metodología.

- **Tipo tecnológico.**
    - Por ejemplo una manera más simple o eficiente para realizar alguna función necesaria.
    - Replanteamiento de algún requerimiento, o bien la sustitución de alguno por otro menos complicado.

- **Impacto.**

a) Este riesgo atrasaría el planeamiento de trabajo llevado hasta el momento de su ocurrencia.

b) Puede que al cambiar alguna manera de realizar las cosas, estas afecten el funcionamiento del proyecto.

- **Probabilidad de ocurrencia.**

20% resultado basado en el juicio de experto, basado en la revisión sobre el estado actual de lo requerido o solicitado.

- **Estrategia de mitigación.**

Para evitar este tipo de inconvenientes o bien que no generen tanto impacto y atraso, lo que se plantea es mantener informado al supervisor sobre qué y cómo se están haciendo las tareas.

- **Estrategia de Concurrencia.**

En caso de que esto ocurra, se optará por adaptar los cambios realizados lo más rápido posible, teniendo en cuenta que pueden surgir otros inconvenientes. Esto con la aceptación del supervisor y posterior revisión.

Riesgos de menor importancia

4- Inconveniente de inexperiencia con la tecnología

- Personas**:**

- **Impacto.**

a) La falta de experiencia en algunos puntos del desarrollo del proyecto puede tener inconvenientes como el atraso en la elaboración del mismo.

b) Ausencia de desarrollo de algún requerimiento importante, a causa de no saber cómo implementarlo en la construcción del proyecto.

- **Probabilidad de ocurrencia.**

10% resultado basado en el juicio de experto, basado en la revisión sobre el estado actual de nuestros conocimientos.

- **Estrategia de mitigación.**

Dedicar una semana a la investigación y desarrollo de ejemplos que nos permitan ya sea conocer la tecnología o herramienta, en caso que sea nueva, o bien recordar y mejorar los conocimientos tenidos en ella.

- **Estrategia de Concurrencia.**

En caso de que esto ocurra, se optará por recurrir a personas que nos guíen o nos den ideas de cómo se debe realizar, y como implementarlo de una mejor manera.

Objetivos y Alcances del sistema:

Objetivo General:

A. Implementar una plataforma para casos de prueba, utilizando herramientas tales como Robotium, Monkey Talk, Calabash, KIF, que permiten automatizar casos de pruebas a nivel móvil, para aplicaciones nativas; esto con el fin de desarrollar una estructura y ambiente de trabajo dentro de la empresa relacionado con automatización de pruebas unitarias a nivel mobile.

Objetivos Específicos:

A. Investigar sobre las herramientas existentes para la automatización de casos de pruebas en aplicaciones nativas en dispositivos móviles.
B. Definir la herramienta a utilizar como base para realizar el proyecto.
C. Implementar y documentar el ambiente y la estructura para crear el script de automatización.
D. Desarrollar el repositorio de scrips (clases y objetos).
E. Documentar el proceso de configuración de la herramienta así como su funcionamiento.

Alcance de proyecto:

Requerimientos del proyecto

Requerimientos funcionales

La solución propuesta debe implementar los siguientes requerimientos funcionales:

A. Investigación, documentación y definición sobre la herramienta a utilizar como base para desarrollar el proyecto.
B. Documentación sobre la configuración del entorno de desarrollo así como la estructura creada.
C. Desarrollo de scripts genéricos (clases y objetos).
D. Documentación del Framework a desarrollar.

Requerimientos no funcionales

La solución del proyecto debe implementar los siguientes requerimientos no funcionales:

A. La solución debe considerar una buena separación de los servicios y la funcionalidad, para facilitar la creación de nuevos scripts automáticos aplicaciones en el futuro. Esta separación también facilita el mantenimiento del sistema.

Capítulo 2 Descripción de la Solución Implementada

La herramienta seleccionada (MonkeTalk) como base de referencia para el proyecto, en sí, utiliza su propia interfaz de desarrollo (IDE) para la programación de sus pruebas y métodos. Sus principales partes se describen a continuación.

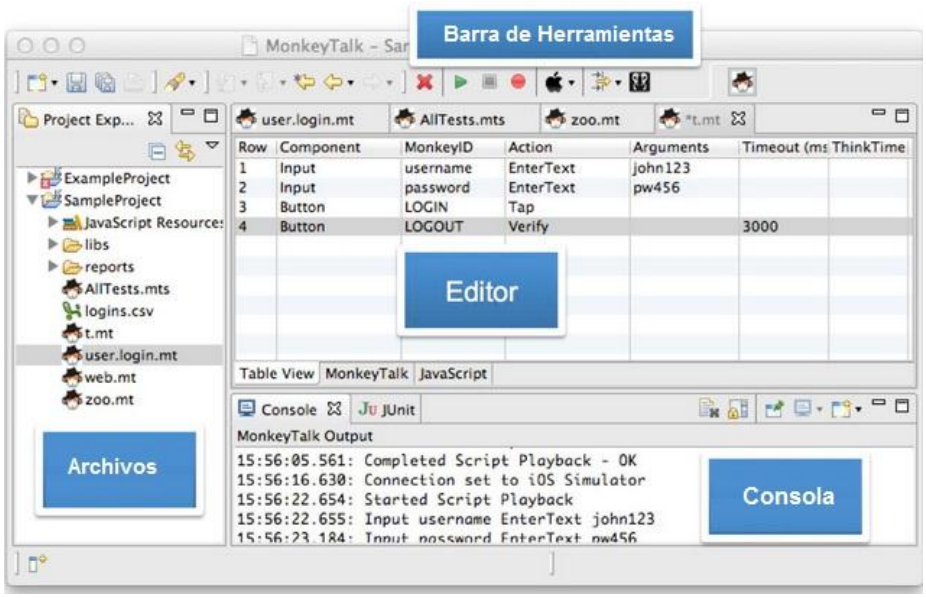

Figura 2.1 Ilustración del IDE de MonkeyTalk.

Como se puede apreciar en la figura anterior, en la barra de herramientas se encuentran las opciones para la conexión al dispositivo o emulador (si fuera el caso), grabar y ejecutar los casos de prueba. El editor es donde se muestra y editan los scripts con los casos de prueba. A la izquierda se muestran los archivos del proyecto (Librerías, Test, TestSuites) y también existe una consola para visualizar el estatus y los resultados al ejecutar una prueba.

A partir de esta base se planteó una solución para implementar el funcionamiento básico de la aplicación en una propia área de trabajo.

Perspectiva del proyecto

A continuación se presenta un diagrama de la perspectiva del proyecto:

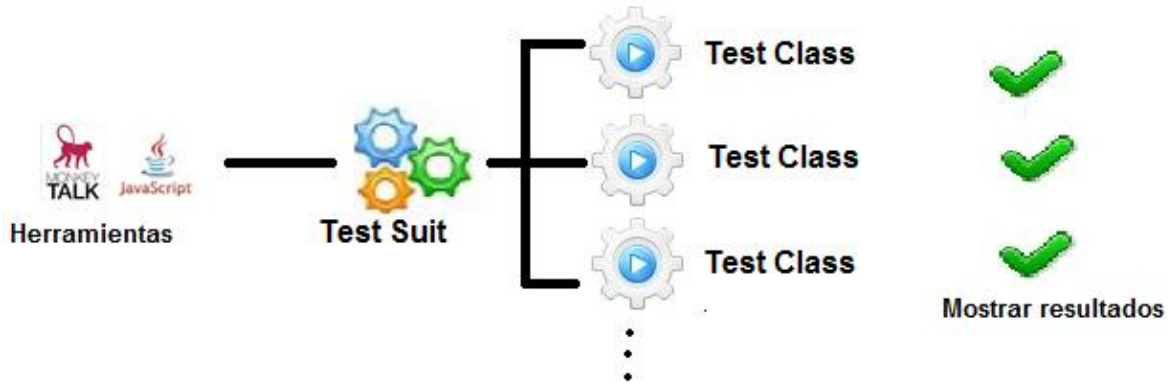

Figura 2.2 Diagrama de la perspectiva del proyecto

Como se puede apreciar en el diagrama anterior, a partir de las herramientas tomadas se crea un TestSuit, los cuales contendrán la llamada a los casos de prueba deseados. Un TestSuit es una colección de TestCases (casos de prueba) que comparten validaciones similares por ejemplo pruebas de inicio de sesión. Un TestCase consiste en una serie de instrucciones que simulan la interacción del usuario con el dispositivo al usar una aplicación.

Podrán existir tantas Test Class como se deseen y cada una contendrá la definición de varios casos de prueba. De este modo desde el Test Suite se podrá ejecutar uno a uno cada caso de prueba y posterior a ello, mostrar y almacenar el resultado de las pruebas realizadas, ya sea que funcionen o fallen.

Modelo de diseño del proyecto

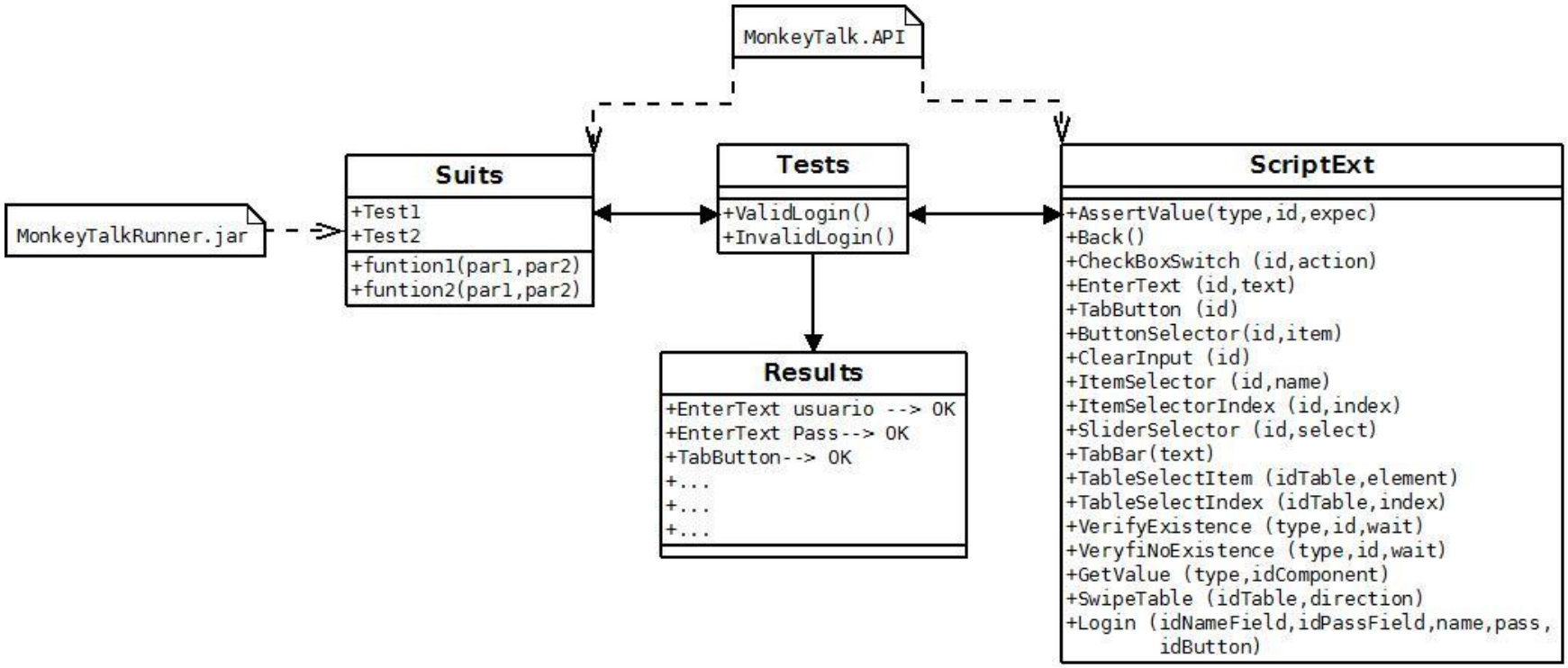

Figura 2.3 Modelo de diseño del Proyecto

Descripción de la aplicación

Componentes y librerías utilizadas

Basándose en el del hecho de que se quiere eliminar la dependencia propiamente del IDE de Monkeyalk, y hacer nuestras propias pruebas fuera de él, se tomaron únicamente los recursos necesarios para ello. Tal es el caso del API de Monkey Talk (el cual viene dentro del paquete de descarga) y lenguaje JavaScript. En el API se encuentran todas las funcionalidades de automatización que brinda, es su librería principal. JavaScript se utiliza para editar casos de prueba, ya que MonkeyTalk tiene la posibilidad de reconocerlo en sus casos de prueba.

MonkeyTalk provee una librería llamada monkeytalk-runner.jar, y como se aprecia en el diagrama anterior, es utilizada para ejecutar un TestSuit y con ello los casos de prueba, esto sin usar o depender de la herramienta como tal, sino, utilizando la consola de Windows.

Esta librería es utilizada al momento de iniciar el proyecto, y se activa desde código escrito en Java. Sus parámetros fueron personalizados de tal forma que inicie la ejecución del TestSuit deseado.

Descripción

El Framework como tal, no posee una interfaz gráfica de usuario, sino que hace uso de las funcionalidades para la visualización del comportamiento de cada uno de los casos de prueba, todo esto dentro del entorno de desarrollo Eclipse.

Como se puede apreciar en la figura 2.3 o diseño del proyecto, la estructura conceptual para la implementación de proyecto, consiste básicamente en cuatro áreas o clases.

Como se explicó anteriormente, el archivo de Suits es el encargado de hacer las llamadas a los distintos casos de prueba que se desean ejecutar.

Utilizando el API de MonkeyTalk, acá se hace la instancia de las TestClass para la ejecución de los casos de prueba deseados. A continuación se presenta un ejemplo:

Figura 2.4 Ejemplo de un Test Suit.

En este caso, podemos observar en la lista, que primeramente se ejecutará el caso de prueba llamado validLogin, una vez concluido se procederá a realizar el siguiente el cual es invalidLogin_wrongPwd, se necesita mínimo uno para que funcione.

Por otro lado tenemos el área de TestClass. Como se explicó anteriormente, una lista con varios casos de prueba que se pueden llevar a cabo. Un ejemplo de un caso de prueba en particular seria el siguiente:

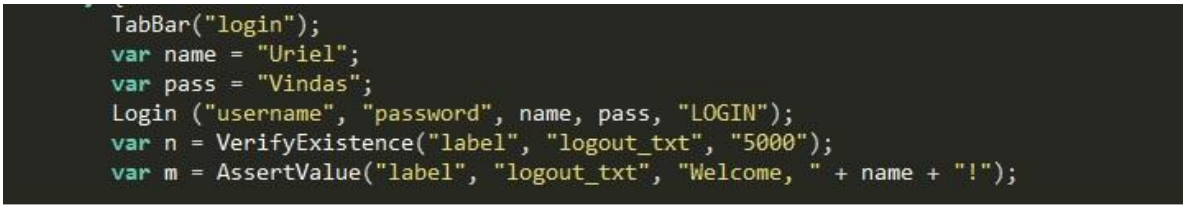

Figura 2.6 Ejemplo de un Caso de Prueba (Test Case).

Como se puede apreciar en la figura anterior, para este caso de prueba se realizará una simple validación para un login. Primeramente se definen tanto de nombre como contraseña deseados y luego se ejecuta una función genérica “Login” enviándole los parámetros definidos. Luego de esto se realiza una verificación de existencia de un componente “label”, y si existe se valida que su valor sea el deseado, de ser así, el caso de prueba se acepta y pasa, de lo contrario falla.

Siguiendo con el modelo, el ScriptExt, es el archivo que contiene todos los métodos genéricos que se están desarrollando e implementando en el framework. De este modo, cada caso de prueba puede llamar y utilizar los métodos de la herramienta, de una forma más sencilla y personalizada. Cabe decir que al ser un archivo con formato javascript, se trata básicamente de una lista de definición de funciones con sus respectivos parámetros, las cuales son llamadas por las clases con casos de prueba. Como en el ejemplo anterior se utilizaron las funciones de Login y verificaciones de existencia, los cuales están definidos en este archivo.

Por último, para poder observar los resultados de cada caso de prueba, se almacena el proceso de ejecución en un archivo de texto, dentro de una carpeta en el mismo proyecto, en el nombre del archivo es especifica la fecha y hora de la ejecución de la prueba, tal y como se muestra a continuación.

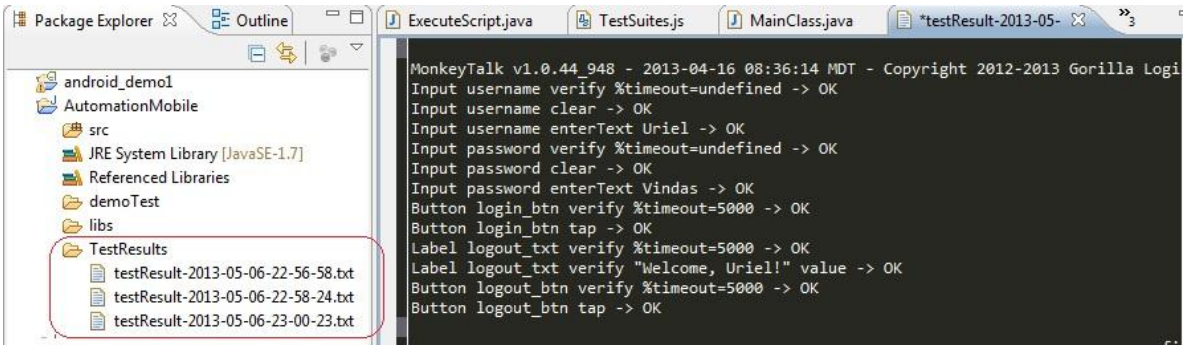

Figura 2.6 Ejemplo de un resultado de prueba.

Se puede observar en la figura anterior, cada paso de la ejecución de un caso de prueba está representado por un renglón, si la instrucción se ejecuta correctamente finaliza con un "OK". Podemos ver como se verifican los campos de texto, luego se inserta el texto tanto de usuario como de contraseña, se presiona el botón Login y por último se hace una verificación del valor del componente deseado. Si ocurre algún error, verificación no valida o inexistencia de algún componente, mostrará un fallo o "FAILURE" junto con su justificación o descripción del problema encontrado.

Conclusiones y comentarios

A. Al terminar el proyecto se lograron cumplir con los objetivos previstos, se entregó un proyecto funcional tanto en plataforma Android como en iOS. El framework como tal servirá de base continuar con futuras integraciones de funcionalidades, así como mejoras y mantenimiento.

B. Toda la documentación de configuración y explicación del framework quedó terminada y entregada. Se realizaron pruebas en otra máquina y el conocimiento del proyecto fue transmitido a otras personas para en un futuro continuar trabajando en él.

C. Se recomienda hacer pruebas en dispositivos reales ya que durante el proyecto solo se desarrollaron desde el emulador.