Características del desarrollo en Frameworks multiplataforma para móviles*

Features Frameworks Development Platform for Mobile

Recibido: 1 de septiembre de 214 - Aprobado: 30 de septiembre de 2014

Para citar este artículo: C. Rodríguez, H. Enríquez, « Características del desarrollo en Frameworks multiplataforma para móviles », *Ingenium*, vol. 15, n.°30, pp. 101-117, octubre, 2014.

Camilo Rodríguez**
Héctor Enríquez***

Resumen

En el presente documento se hace una revisión sobre las características de diferentes frameworks para el desarrollo multiplataforma de aplicaciones móviles, partiendo desde el tamaño generado para una misma aplicación, utilizando alternativas básicas para su ejecución y analizando variables que generan dificultad en la programación y de esta manera determinar el entorno más conveniente en el momento de iniciar un proyecto de desarrollo para móviles.

Palabras clave

Framework, dispositivos móviles, aplicaciones nativas, aplicaciones web, aplicaciones híbridas.

Summary

This paper reviews the characteristics of different frameworks for cross-platform mobile application development, starting from the size generated for the same application, using basic alternatives for execution and analyzing variables that create difficulty in programming

* Artículo de investigación, producto derivado del proyecto de *investigación Análisis y comparación de Frameworks de desarrollo multiplataforma para dispositivos móviles*, realizado en el grupo ICT Research de la Universidad de San Buenaventura, llevado a cabo entre febrero y noviembre de 2013.

** Ingeniero de Sistemas. Magíster en Ingeniería de Sistemas de la Universidad Nacional de Colombia.

*** Ingeniero de Sistemas. Especialista en Gerencia de Proyectos de Información de la Universidad del Rosario.

and makes this way to determine the most suitable environment at the time of initiating a development project for mobile.

Keywords

Framework, mobile, native apps, web apps, hybrid apps.

1. Introducción

En la evolución tecnológica cada día toman más presencia en la cotidianidad los dispositivos móviles, evento por el cual se requiere de manera urgente una apropiación de herramientas que permitan agilizar procesos o facilitar actividades de desarrollo de aplicaciones para estas tecnologías. Una de las principales dificultades al desarrollar aplicaciones para móviles es la diversidad de dispositivos y sistemas operativos, teniéndose que construir una versión diferente para cada caso en un lenguaje y una herramienta diferente. En los últimos años han aparecido una serie de frameworks que permiten escribir la aplicación una sola vez y compilar para las diferentes plataformas. Es por esta razón que se realiza una comparativa sobre las características existentes de algunos de estos entornos de programación para móviles, pretendiendo acercar a los interesados en este tipo de desarrollo a las dificultades que encontrarán dentro de la experiencia de la programación en distintos sistemas operativos. Se ha programado una misma aplicación en las diferentes plataformas para establecer una comparación tanto en la etapa de desarrollo como en la etapa de ejecución.

2. Metodología

El desarrollo del presente estudio se ha realizado en cuatro fases, las cuales serán descritas a continuación:

2.1 Revisón de frameworks existentes

En el mes de febrero del año 2013, se han tenido en cuenta algunas herramientas útiles para el desarrollo de software multiplataforma para dispositivos móviles, las cuales reciben la denominación de Framework. Se eligieron los frameworks a estudiar basados en dos aspectos y los objetivos de este proyecto: que el framework permita generar aplicaciones para varias plataformas, y que la aplicación sea híbrida, de tal manera que se pueda generar una aplicación móvil como tal y no una aplicación web. Los frameworks seleccionados generan aplicaciones para al menos cuatro plataformas diferentes. Dentro de los frameworks seleccionados, se cuenta con Phonegap, Rhodes, Application Craft, Worklight, Mosync y Codename One.

2.1.1 Phonegap

PhoneGap se desarrolló originalmente por la empresa Nitobi en 2011 y fue adquirido posteriormente por Adobe, liberando el código bajo el proyecto Apache Cordova. En otras palabras, el producto que pertenece a Adobe se denomina actualmente PhoneGap y el

proyecto de código abierto se denomina Cordova. Varios de los framework descritos en este documento utilizan Cordova. El objetivo de este framework es permitir desarrollar aplicaciones embebidas en código nativo utilizando exclusivamente código html5 y Javascript (Ghatol, 2012). Phone Gap Build ofrece varios tipos de plan pagos. Debido a que el código se compila a través de gitHub, el derecho a que el código sea privado depende del precio del plan. La versión gratuita permite una sola aplicación privada y la paga va desde USD $120 a USD $900 por año, dependiendo del número de aplicaciones.

2.1.2 Rhodes

Rhodes es un framework de código abierto desarrollado por Rhomobile en diciembre de 2008 (Allen, 2010) y adquirido por Motorola en julio de 2011. La principal característica de este framework es que se desarrolla en Html5, Javascript y Ruby siguiendo la filosofía de la arquitectura MVC (Modelo-Vista-Controlador) utilizada en Ruby on Rails. Las vistas, es decir la interfaz de la aplicación, se generan utilizando html5, Javascript y ERB (Embedded Ruby), tal y como se generan páginas utilizando lenguajes del lado del servidor con PHP o JSP. Las solicitudes del cliente se procesan por medio de controladores y el acceso a datos por medio de los modelos (Allen, 2010). El código generado es RubyByte code el cual se ejecuta sobre una máquina virtual de Ruby. Si bien Rhodes se puede descargar de manera gratuita, Motorola también ofrece una versión empresarial con soporte que cuesta USD $500 y el servidor Rhode RhoSync, a un precio de USD $5000 por 100 usuarios conectados.

2.1.3 Application Craft

Este framework utiliza el código de PhoneGap-Cordova. Este framework solo ofrece una versión para compilar en línea, la cual cuesta USD $14 por mes, y una versión empresarial descargable. Sin embargo, la versión en línea ofrece un entorno gráfico de desarrollo para Html5 y javascript, no solo para el proceso de compilación como es el caso de PhoneGap Build.

2.1.4 Worklight

Es un framework para el desarrollo de aplicaciones para dispositivos móviles desarrollado por IBM, el cual se compone de cuatro herramientas: IBM Worklight Studio la cual corresponde a un entorno de desarrollo basado en Eclipse; IBM Worklight Device Runtime Components, que permite tener facilidad en tiempo de ejecución del aplicativo desarrollado con el componente de Hardware (dispositivo móvil); IBM Worklight Server que corresponde al servidor utilizado por el framework, y está basado en Java; finalmente, IBM Worklight Console, herramienta de interfaz de usuario que permite controlar la totalidad de las aplicaciones móviles.

2.1.5 Mosync

Este framework permite desarrollar aplicaciones que se ejecutan en dispositivos móviles el cual está separado por dos categorías a saber: el SDK y el Reload. La principal diferencia es que la primera permite el desarrollo de aplicaciones nativas e híbridas, mientras la segunda solo permite el desarrollo de aplicaciones híbridas.

2.1.6 Codename One

Codename One es un framework perteneciente a la empresa que lleva el mismo nombre. Este framework permite el desarrollo de aplicaciones multiplataforma y ofrece la posibilidad de crear aplicaciones nativas, similar a Mosync, y a diferencia de otros frameworks explicados en este documento, como Phonegap. Las aplicaciones son desarrolladas en lenguaje java utilizando un plugin para los IDE netbeans o eclipse, y convertidas a código para Dalvik en Android, MIDP+JSR's en J2ME, RIM API en Blackberry, Objective C en iOS y C# en Windows Phone 7.

2.2 Selección de entornos

Se analizaron los frameworks presentados con el objetivo de depurar acertadamente las plataformas a ser evaluadas. Para desarrollar de manera adecuada este procedimiento, se elaboró un análisis DOFA (Debilidades, Oportunidades, Fortalezas, Amenazas) para cada framework propuesto, teniendo como resultado las siguientes tablas:

PHONEGAP	
DEBILIDADES	El servicio de notificaciones está limitado a Andriod y a iOS. No permite desarrollo multiplataforma de aplicaciones nativas, solamente híbridas.
OPORTUNIDADES	Aprovecha de manera extensa los servicios y utilidades de los dispositivos móviles así como de las plataformas de desarrollo.
FORTALEZAS	Utiliza la arquitectura cordova, elemento en el que se basan otros frameworks, lo que lo hace pionero en arquitectura.
AMENAZAS	Precio variable de acuerdo con el número de aplicaciones, evento que puede disminuir el interés por el producto.

Tabla 1: DOFA para Phonegap

RHODES	
DEBILIDADES	El desarrollador debe conocer el lenguaje Ruby y estar familiarizado con el patrón de desarrollo MVC y el framework Ruby on Rails. Las aplicaciones producidas son híbridas y no nativas.
OPORTUNIDADES	Generar las aplicaciones con un framework similar a Ruby on rails bajo MVC hace que la tecnología de las mismas sea igual a la empleada en cualquier aplicación web cliente-servidor bajo MVC.
FORTALEZAS	Maneja un tipo de arquitectura (Modelo Vista Controlador), que es la arquitectura más empleada en la actualidad para el desarrollo de aplicaciones web.
AMENAZAS	Versión empresarial con costo, lo cual puede minimizar la cantidad de usuarios que se inclinen por el aprendizaje de este servicio.

Tabla 2: DOFA para Rhodes

APPLICATION CRAFT	
DEBILIDADES	La versión descargable no tiene entorno gráfico para el desarrollo con HTML 5 y JavaScript. Solo se pueden desarrollar aplicaciones híbridas.
OPORTUNIDADES	Usa la arquitectura cordova, evento que le puede generar respaldo.
FORTALEZAS	La versión en línea ofrece un entorno gráfico de desarrollo para HTML 5 y JavaScript.
AMENAZAS	Costo de compilación mensual, lo cual reduce el número de posibles interesados en este framework.

Tabla 3: DOFA para Application Craft

WORKLIGHT	
DEBILIDADES	La versión libre del framework es una versión beta que está en constante retroalimentación para mejorar las versiones pagas. Sólo permite el desarrollo de aplicaciones nativas multiplataforma.
OPORTUNIDADES	Respaldo de una casa de tecnología con mucha trayectoria (IBM), lo que genera confianza al desarrollador y al usuario.
FORTALEZAS	IBM ofrece un conjunto de herramientas no solo para el desarrollo de la aplicación, sino también para la integración con otras plataformas y servicios de software.
AMENAZAS	Alto costo de adquisición de las versiones más sofisticadas, reduce la cantidad de personas que utilizan el framework, por ende la masificación del mismo.

Tabla 4: DOFA para Worklight

MOSYNC	
DEBILIDADES	N/A
OPORTUNIDADES	Cobertura amplia en el desarrollo de aplicaciones debido a la clasificación de tecnologías dentro del SDK y el Reload.
FORTALEZAS	Manejo de lenguaje JavaScript, HTML 5, CSS y C/C++ permite más opciones en la programación de aplicativos, permitiendo desarrollar tanto aplicaciones nativas como aplicaciones híbridas.
AMENAZAS	Manejo de dos versiones de framework, puede crear confusión en los usuarios.

Tabla 5: DOFA para MoSync

CODENAME ONE	
DEBILIDADES	No maneja el lenguaje HTML 5, lo que reduce el tipo de aplicaciones a desarrollarse. Se desarrolla exclusivamente en JAVA.
OPORTUNIDADES	Compilación bajo servidores de la nube, lo que hace mucho más rápida la generación de ejecutables.
FORTALEZAS	Estandarización del lenguaje java, que facilita la portabilidad de los aplicativos. Crea aplicaciones nativas.
AMENAZAS	Costo para compilar aplicaciones, puede reducir el uso del framework.

Tabla 6. DOFA para Codename One

Con base en las variables relacionadas en las matrices DOFA para cada uno de los framework, se establece la utilización de tres de estos, los cuales son Phonegap, MoSync y Codename One. Se decide esto debido a que Application Craft cuenta con arquitectura Cordova, la cual es la misma arquitectura que utiliza Phonegap, por este motivo se selecciona este último, ya que el uso de esta arquitectura haría de la utilización de los dos framework una prueba redundante.

En el caso de Rhodes y Worklight se ha decidido suprimir su utilización debido a que demandan un costo para la compilación con garantía (las versiones libres son pruebas para ejecutar una retroalimentación, lo cual no forma parte del presente estudio).

2.3 Desarrollo de aplicaciones

Para analizar la funcionalidad de los frameworks seleccionados, se ha decidido desarrollar una aplicación básica, la cual ha sido una calculadora con las funciones elementales.

Se ha determinado que sea esta aplicación el elemento de prueba debido a que tendrá un comportamiento reducido en cada framework, lo cual facilitará analizar sus condiciones tanto para la etapa de desarrollo como para la etapa de compilación y pruebas de ejecución.

2.3.1 Phonegap

Dentro de las características comprobadas en Phonegap, está el hecho de trabajar con los lenguajes Html5 y javascript, evento que representa estandarización dentro de las características actuales de la programación con entorno gráfico.

Después de tener el código escrito desde cualquier editor, se debe subir a un repositorio GITHUB, donde se tendrá el código disponible para su compilación. Es preciso aclarar que se debe tener una cuenta bajo esta plataforma (GITHUB) para poder acceder a este servicio (figura 1) y también se debe ingresar al sitio web de Phonegap para poder compilar.

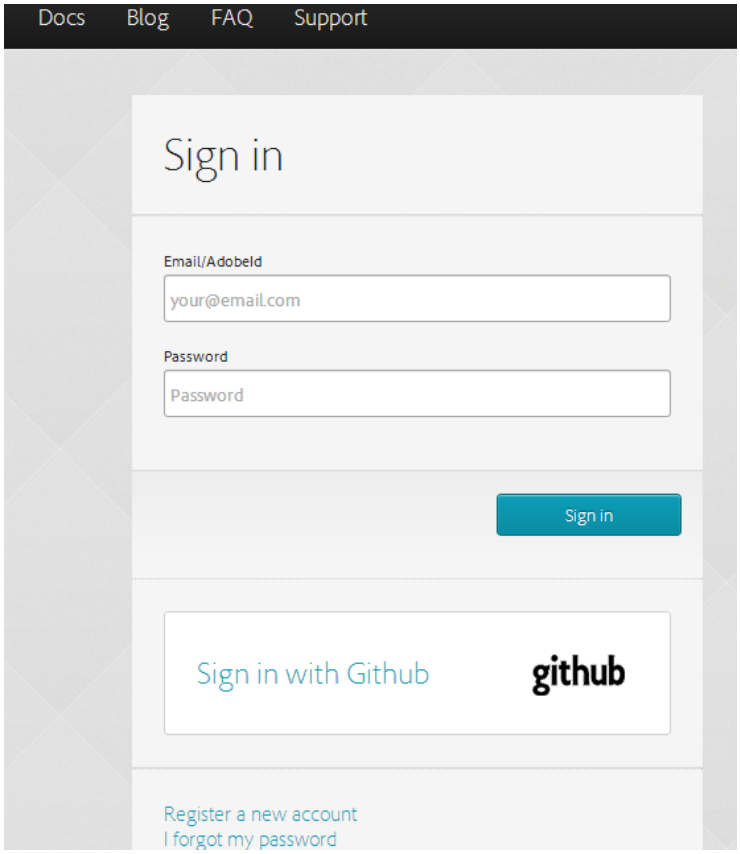

Figura 1. Acceso a una cuenta en GITHUB

Cuando el código se ha subido a dicho repositorio (GITHUB), se ingresa a la cuenta PhoneGap, desde donde se recupera para que pueda ser compilado para cualquier plataforma.

El proceso de compilación se hace en línea.

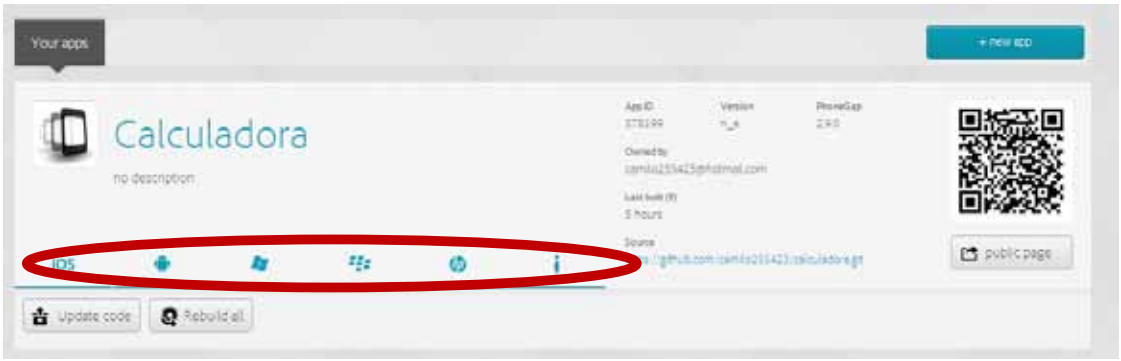

Figura 2. Compilación de un aplicativo en Phonegap multiplataforma

Para el caso de IOS se debe contar con los certificados de desarrollo o distribución.

2.3.2 MoSync

El framework llamado MoSync trabaja el lenguaje de programación C++, evento positivo teniendo en cuenta la amplia trayectoria de dicho lenguaje y el soporte que tiene, así como la documentación existente tanto de manera física como en Internet.

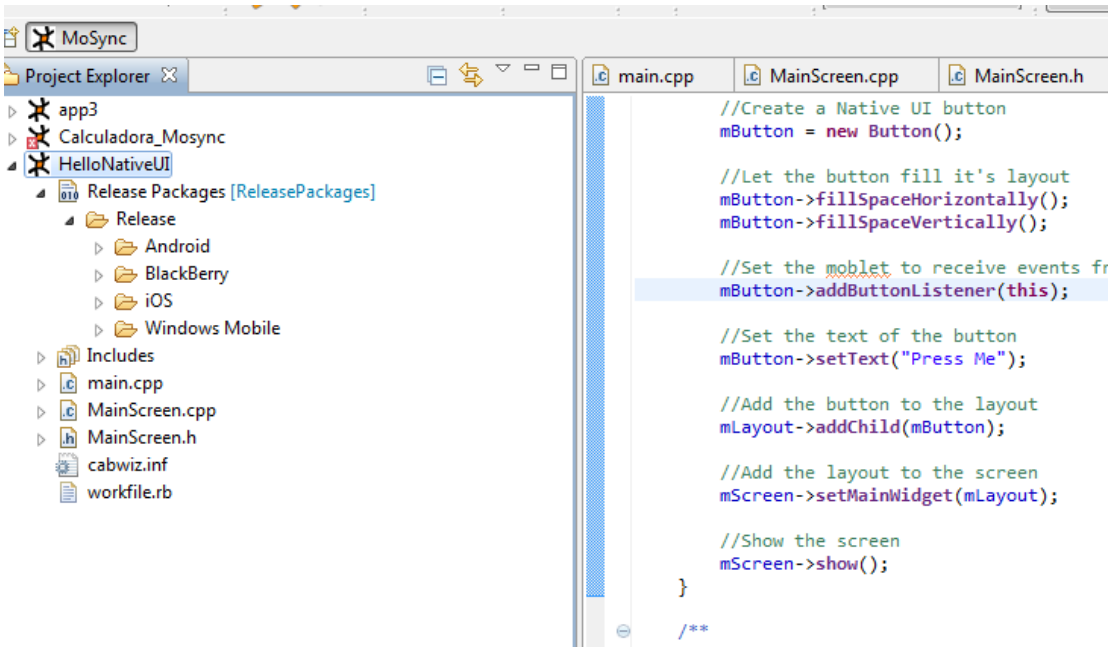

Figura 3. Interfaz de desarrollo de MoSync basada en C++

En el momento de compilar el código escrito desde un editor, este proceso se debe realizar para cada plataforma de manera local, seleccionando la misma. En la figura 4 se ve el proceso de compilación para Android.

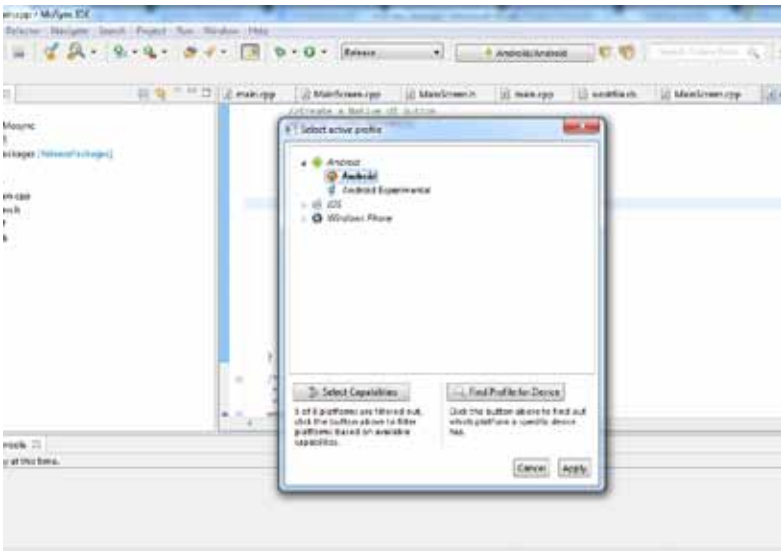

Figura 4. Compilar aplicativo en MoSync para Android

Dentro del desarrollo realizado se ha podido comprobar que no todas las bibliotecas de C y C++ funcionan en el framework, el cual trae adaptadas algunas de estas y son las únicas que funcionan de manera correcta para el desarrollo.

El plug in para eclipse no funciona correctamente. En algunas ocasiones el editor señala errres de sintaxis incluso habiéndolos ya corregido.

Otra de las características encontradas es que no cuenta con herramientas visuales, motivo por el que es necesario realizar la totalidad del desarrollo por medio de código.

2.3.3 Codename One

La programación de este framework se realiza en Java, para lo cual codename one ha creado paquetes propios para su desarrollo. En la prueba se ha trabajado con el plug in para Netbeans. Además del insalador para Netbeans o Eclipse se debe contar con una cuenta, la cual se puede registrar de manera gratuita, en el sitio de Codename one.

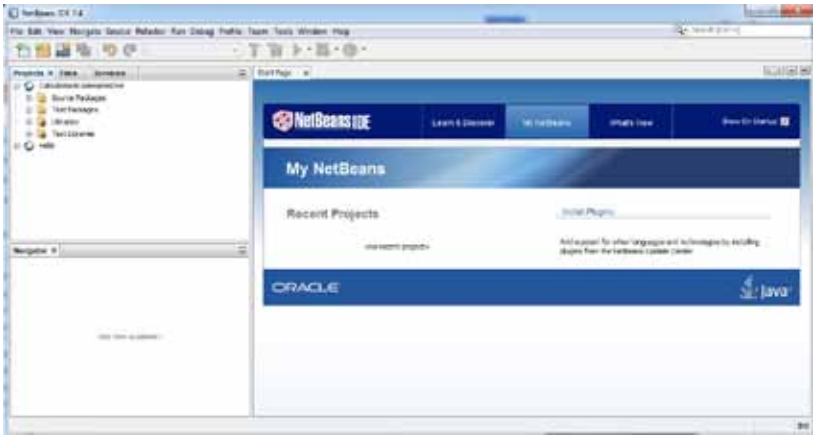

Figura 5. Editor de Netbeans utilizado para Codename One

Para el diseño de la interfaz, este framework presenta herramientas visuales, lo que permite tener un desarrollo más ágil. En la figura 6 se visualiza una de estas herramientas, donde se ha conseguido una interfaz para el aplicativo calculadora.

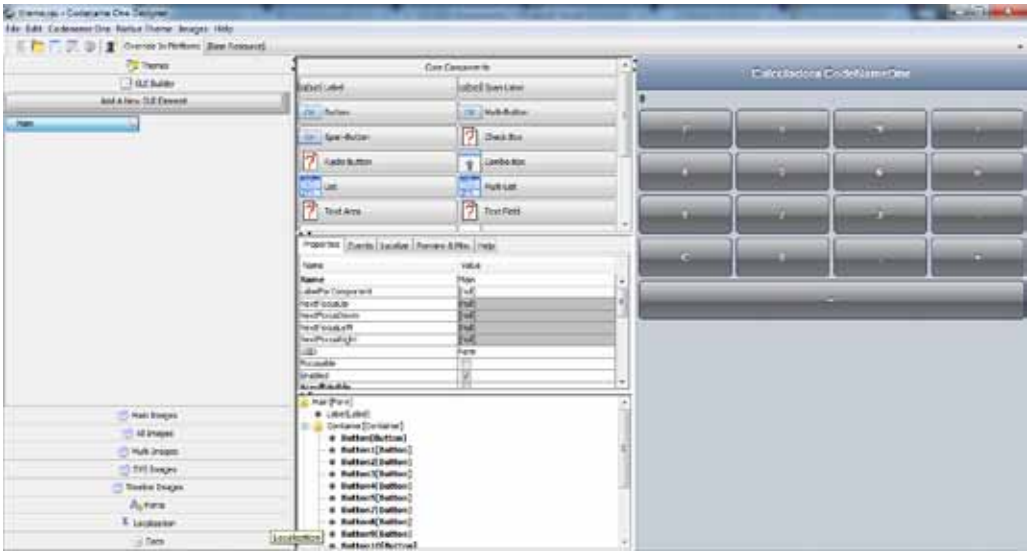

Figura 6. Herramientas para el diseño de la interfaz visual en Codename One

Codename One garantiza que las aplicaciones que van a ser compiladas tienen soporte de legalidad, para lo cual solicita sean especificados los certificados de desarrollo y distribución para cada plataforma.

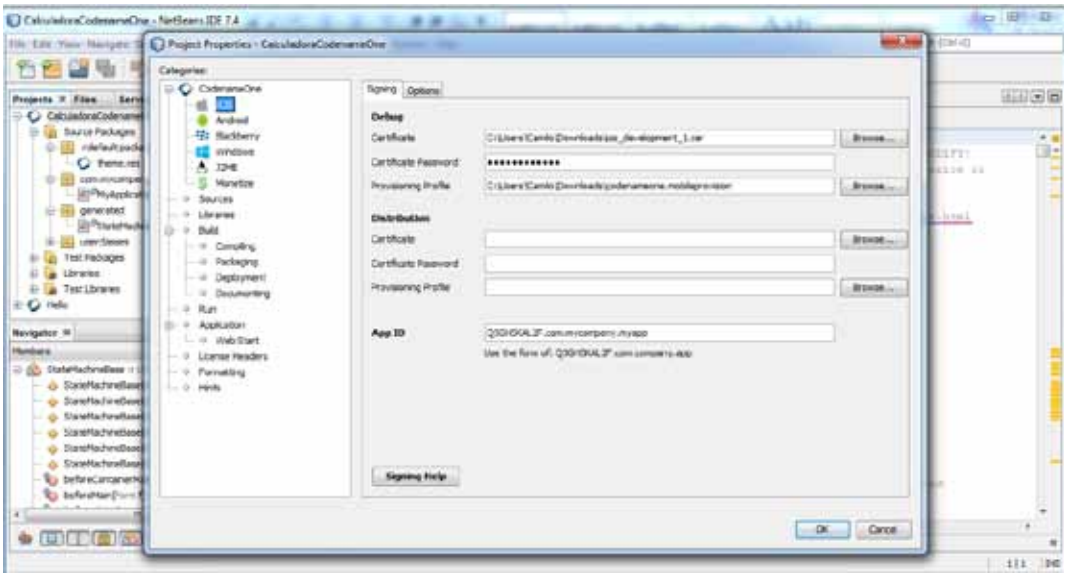

Figura 7. Especificación de certificados de desarrollo y distribución Codename One

Para compilar el aplicativo, se debe seleccionar la plataforma en la que se ha trabajado y posterior a esto se envía el código del mismo al servidor, donde se podrá descargar el archivo definitivo.

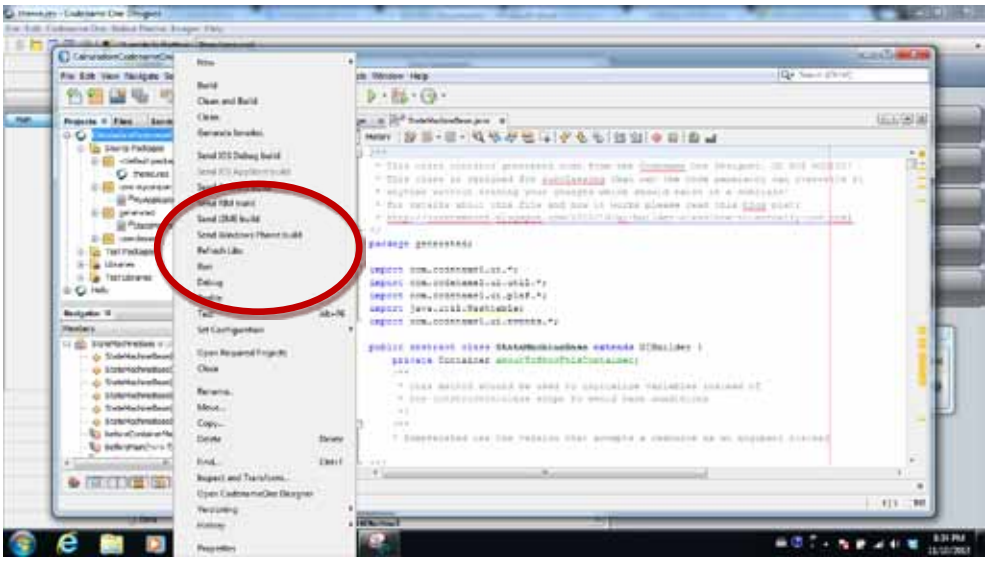

Figura 8. Selección de plataforma y envío al servidor para compilar en Codename One

El resultado final puede ser descargado desde el servidor, el cual ha generado el archivo con la aplicación para la plataforma que se haya creado.

Figura 9. Recepción de archivo compilado

Codename One presenta un desarrollo simple, que tiene un buen comportamiento en el editor de eclipse y netbeans, y como valor adicional cuenta con herramientas para crear la interfaz gráfica, las cuales son de mucha utilidad. Una de las ventajas es que debe compilarse directamente en el servidor.

2.4 Análisis y discusión de resultados

En esta etapa lo primero que se realizó fue la relación de peso de cada aplicativo para cada framework. Estos pesos se tomaron luego de tener el archivo final del proyecto Calculadora para cada plataforma. Dicha relación se presenta a continuación:

FRAMEWORK	iOS (ipa)	Android (apk)	Windows Phone (xap)	Blackberry (jad)
Phonegap	3436 KB	468 KB	345 KB	6 KB
MoSync	N/A	671 KB	N/A	1 KB
Codename One	N/A	1377 KB	N/A	3 KB

Tabla 7. Relación de peso de una calculadora en distintos frameworks para las plataformas analizadas

Como se puede apreciar, hay cuatro archivos dentro de la relación que no presentan peso asociado; esto se debe a que mostraron fallos a la hora de ser compilados, por lo que no se contó con archivo definitivo para estos frameworks. En el caso de MoSync para iOS, se ha generado un código para ser compilado en XCode, lo cual no está siendo validado dentro de la presente investigación. En MoSync para Windows Phone, se presentó fallo en la generación de archivo final. En Codename One para iOS, presenta fallos en el servidor por dificultades en la asociación de certificados de desarrollador, por lo que no se ha generado archivo definitivo. En Codename One para Windows Phone, presenta error de compilación, lo que hace que el desarrollo para esta plataforma sea diferente a las anteriormente validadas, tampoco se cuenta con archivo final.

Después de medir los pesos de cada uno de los archivos finales se ha procedido a la prueba de los mismos en dispositivos físicos, en blackberry se ha probado en una Blackberry curve 9320 y en los emuladores Blackberry 8520, Blackberry 9105 y Blackberry 9860.

Figura 10. Emulador de Blackberry 9105

Dentro de las dificultades para compilar en estos dispositivos se ha tenido que los archivos .jad y .jar no son reconocidos como archivos válidos, por ende el sistema no permite su descarga ni compilación. Para solventar dicho inconveniente, se ha generado un enlace en web directamente desde el sistema de compilación provisto por phonegap, al cual se ha accedido en el móvil a través de su navegador a Internet. Se ha descargado el archivo y ejecutado directamente, con un resultado exitoso, pero hay que mencionar que con relación a las otras pruebas su comportamiento en tiempo de respuesta ha sido mucho más tardío.

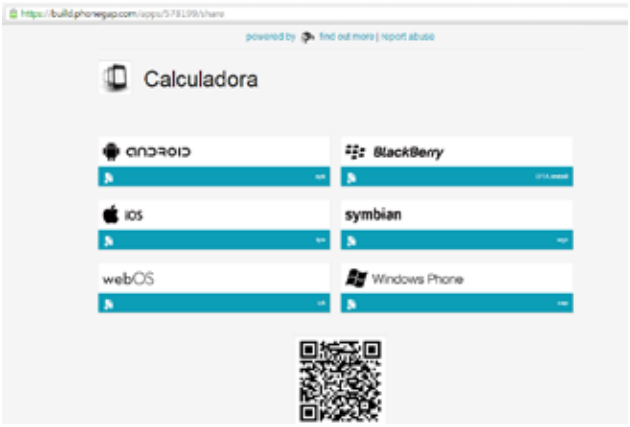

Figura 11. Enlace del servidor de phonegap Multiplataforma

Para android se ha probado en un dispositivo Samsung Galaxy S2, con sistema operativo 4.1.2, el cual ha presentado resultados satisfactorios para los tres frameworks.

Figura 12. Calculadora en Phonegap para Android

En cuanto al rendimiento del aplicativo la recepción de datos mediante la pantalla táctil ha tenido mejor respuesta por parte del aplicativo creado en MoSync, el cual responde casi de manera instantánea; el aplicativo creado en Phonegap tarda un poco pero si se realizan pulsaciones continuas no pierde información, mientras que la aplicación en Codename One al principio se demora en responder.

Figura 13. Calculadora en MoSync para Android

La instalación es sencilla, simplemente se ejecuta cada archivo, el sistema operativo valida la peligrosidad de los aplicativos y procede a instalarlos, en un proceso que tardó 15 segundos por cada uno (el mismo tiempo exacto para todos los frameworks).

Figura 14. Calculadora en Codename One para Android

Se ha probado el archivo en un iPhone 4, presentando dificultades en su proceso de instalación ya que la seguridad de dicho dispositivo lleva a que se deba trabajar directamente con herramientas como el iTunes y el iTools, lo cual genera dificultades en el proceso de ejecución de la aplicación.

2.4.1 Estructura de costos

Con base en la revisión realizada y con las pruebas y resultados obtenidos se realiza un breve análisis de los costos que genera el desarrollo en cada uno de los frameworks anteriormente descritos.

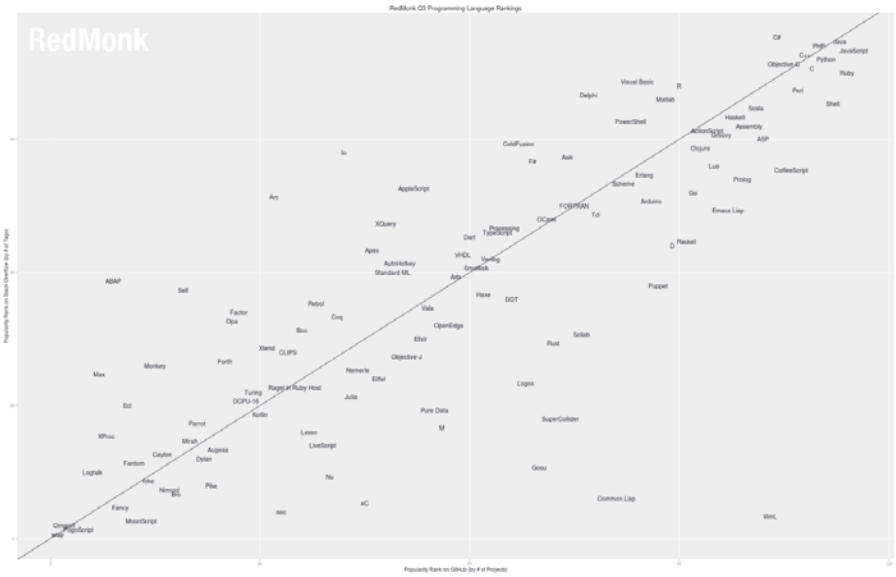

Figura 15. Índice de popularidad de lenguajes de programación

De acuerdo con el reporte que ha realizado la firma Redmonk[1] a junio de 2013, se clasifican los lenguajes de programación de acuerdo con su utilización. Para establecer esta métrica se han tenido en cuenta la cantidad de proyectos en el repositorio GitHub y el número de relacionados en foros de programación (dentro del estudio de esta firma).

Se han establecido los 10 primeros en la siguiente tabla:

POSICIÓN	LENGUAJE	FACTOR COSTO
1	Java	1,00000
2	JavaScript	1,05263
3	PHP	1,10526
4	Python	1,15789
5	Ruby	1,21053
6	C#	1,26316
7	C++	1,31579
8	C	1,36842
9	Objective-C	1,42105
10	Shell	1,47368
11	Perl	1,52632

1 RedMonk es una firma fundada en 2002, se enfoca en análisis industriales. Dentro de sus principales clientes están Adobe, Microsoft, SAP, entre otros.

POSICIÓN	LENGUAJE	FACTOR COSTO
12	Scala	1,57895
13	Assembly	1,63158
14	Haskell	1,68421
15	ASP	1,73684
16	R	1,78947
17	CoffeeScript	1,84211
18	Groovy	1,89474
19	Matlab	1,94737
20	Visual Basic	2,00000

Tabla 16. Listado de lenguajes de programación más comunes según RedMonk

Con base en la información anterior, es preciso aclarar que a mayor cantidad de programadores existentes para un lenguaje, los costos de programación se hacen más económicos. Dentro de los foros exclusivos para programadores se encuentra que el promedio de costo por hora de programación está entre los 15 y los 40 dólares (dependiendo del nivel de experiencia y de popularidad del lenguaje), así que se tomará como elemento referencial el valor mínimo en dólares (15) para establecer el criterio de costo a cada lenguaje.

El primer lenguaje tendrá el factor multiplicador 1, en adelante se hace un fraccionamiento hasta llegar al factor 2, por lo que será incremental lineal para establecer el precio de hora por cada lenguaje.

A esto hay que sumarle la cantidad de horas a contratar al programador y los costos fijos del proyecto.

De las plataformas utilizadas, iOS exige un certificado anual para su compilación, el cual tiene un costo de 99 dólares y es necesario trabajar con un equipo Mac, por lo que se hace de estricto cumplimiento su adquisición.

En cuanto a las otras plataformas se establece un costo por dispositivo, el cual es variable (de acuerdo al equipo adquirido para cada proyecto).

De esta forma se tendría para establecer los costos de este tipo de proyectos como se muestra a continuación:

$$TP = CF + \frac{CL}{2880} + \left(15 * FC\right) * TH$$

La ecuación se interpreta de la siguiente manera

TP = Total Proyecto

CF = Costos Fijos

CL = Costo Licencia[2]

FC = Factor Costo

TH = Total Horas

La anterior ecuación presenta la fórmula para establecer el costo de un proyecto tomando en cuenta el lenguaje y por lo tanto del framework.

Sin embargo, la selección del framework depende del tipo de proyecto. Si se requiere desarrollar una aplicación móvil, donde no se necesite usar características propias de las aplicaciones nativas, se puede usar un framework de aplicaciones híbridas y sencillo, como Phone Gap, cuyo costo adicional sería si requiere que el código del repositiorio GItHub sea privado. El costo actual de este tipo de cuenta se muestra a continuación:

Gratis	Bronce	Plata	Oro	Platino	
Precio mensusal	$0	$25	$50	$100	$200
Miembros	Sin límite	Sin límite	Sin límite	Sin límite	Sin límite
Repositorios Públicos	Sin límite	Sin límite	Sin límite	Sin límite	Sin límite
Repositorios Privados	0	10	20	50	125

Tabla 1. Precios en dólares Github, tomado de www.github.com

En el caso de Rhodes, los costos adicionales dependen del tipo de licencia de la aplicación móvil desarrollada. Bajo licencia de código abierto GPLv3 el uso del framework no tiene ningún costo.

Para aplicaciones comerciales tiene un costo de $500 dólares por aplicación. Adicionalmente puede utilizar los servicios como RhoSync y RhoConnect para sincronizar las aplicaciones desarrolladas con aplicaciones Backend. El costo de estos servicios depende del plan empresarial.

Conclusiones

De los frameworks revisados, desde el punto de vista de facilidad de programación se recomienda codename One, ya que se programa en java estandar, trae herramientas de entorno gráfico para el desarrollo de la interfaz y el plugin para netbeans funciona correctamente. Sin embargo, a la hora de generar desde el servidor se presentan fallas para la generación en algunas plataformas, como es el caso de windows phone y iphone.

2 Se divide el costo de licencia entre 2880 pues es la cantidad de horas útiles que se cuentan por año para trabajar (30 días calendario, 12 meses, 8 horas diarias).

Las otras opciones recomendadas son:

- PhoneGap
- WorkLight
- Rhodes

Sin embargo, hay que aclarar que estas opciones generan aplicaciones híbridas, es decir html5-javascript embebidas en una aplicación nativa.

Phonegap es la más sencilla de utilizar, y genera aplicaciones que no requieran gran complejidad en su arquitectura (por ejemplo elaboración por capas, sincronización con servidores).

Por el contrario WorkLight y Rhodes no son solo frameworks para la construcción de la aplicación del lado del cliente (el celular) sino también del lado del servidor contando con varios servicios de sincronización.

En el caso de Rhodes, es el único framework que ofrece el desarrollo de la aplicación móvil bajo el patron de desarrollo MVC, usando un framework basado en Ruby on Rails.

Dentro de los problemas que se tuvieron a la hora de probar las aplicaciones en los teléfonos, el principal fue la dificultad para instalar las aplicaciones en el dispositivo. En algunos casos se tuvo problemas con los certificados a través de los framework, como por ejemplo IOS en codename one y en otras ocasiones.

En términos de costos, basado en el lenguaje de programación es más económico desarrollar en phonegap, worklight, o codename one ya que requieren java o html-Javascript que frameworks como Rhodes que requieren programadores en Ruby.

Referencias

Codename one architecture, http://codenameone.com/.

IBM worklight architecture, www-01.ibm.com/software/mobile-solutions/worklight/features/phonegap/

Mosync architecture, www.mosync.com/documentation/manualpages/runtime-architecture

Sarah Allen, Vidal Graupera, and Lee Lundrigan. Pro Smartphone Cross-Platform Development: iPhone, Blackberry, Windows Mobile and Android Development and Distribution. Apress, Berkely, CA, USA, 1st edition, 2010.

Andre Charland and Brian LeRoux. Mobile application development: Web vs. native. Queue, 9(4):20:20 20:28, April 2011.

Mohamed E. Fayad, Douglas C. Schmidt, and Ralph E. Johnson. Building application frameworks: object-oriented foundations of framework design. John Wiley, & Sons, Inc., New York, NY, USA, 1999.

Rohit Ghatol and Yogesh Patel. Beginning PhoneGap: Mobile Web Framework for JavaScript and HTML5. Apress, Berkely, CA, USA, 1st edition, 2012.

Motorola. Device capabilities, http://docs.rhomobile.com/rhodes/devicecaps, 3 2013

Motorola. Framework architecture, http://docs.rhomobile.com/rhodes/introduction, 3 2013.

Bakker Richard. Mobile Application Development: Strategies for Accelerated Delivery and Deployment. In Innovate Comes to You 2011, The Rational Software Conference, 2011.

Dirk Rlehle. Framework Design: A Role Modeling Approach. Ph.D. thesis, ETH Zürich, Zürich, Switzerland, 2000.