ENFOQUE MDA PARA EL DESARROLLO DE APLICACIONES MÓVILES CON FUNCIONALIDADES ENRIQUECIDAS

Alumno: Emanuel A. Sanchiz F.
Tutores: Ing. Magalí González, Dr. Luca Cernuzzi

1 Introducción

En estos últimos años el número de usuarios de teléfonos inteligentes y tabletas ha ido creciendo rápidamente, tanto que ha superado al de usuarios de computadores personales, ya sean de escritorio o portátiles[1][2][3]. Estos dispositivos móviles están ocupando un espacio muy importante en la cotidianidad de las personas. Hoy en día, mediante ellos, el acceso a internet lo tenemos literalmente en nuestras manos, y con ello accedemos a un abanico de posibilidades y servicios de comunicación, educación, comercio y más. La evolución de los teléfonos inteligentes y las tabletas, los cuales brindan una interacción innovadora y cada vez más cómoda, es uno de los factores responsables de una inminente era móvil.

Consecuentemente se da una explosión de demanda de aplicaciones móviles y con ello aumenta el interés por su desarrollo. Sin embargo, el cambio de ambiente de escritorio a móvil tiene sus complicaciones. Entre una PC y un teléfono inteligente hay diferencias significativas en cuanto a recursos y se podrían citar los de computación, almacenamiento, provisión de energía, seguridad entre otros. En este sentido, haciendo referencia a las aplicaciones que requieran de funcionalidades que superen las posibilidades de un dispositivo móvil, se abre la necesidad de recurrir a distintas estrategias, donde una de las cuales es la extensión de las capacidades locales a través del acceso a funcionalidades y servicios remotos (implementados en la nube).

En relación con lo anterior, en Abolfazli et al. [1], se menciona lo siguiente: *"desde la invención del computador uno de los principales puntos de motivación para los avances de la tecnología de computación es el mejoramiento de la experiencia de usuario en cuanto a interacción y a computación, lo cual se ha traducido, por ejemplo, en las invenciones de la web y de las Rich Internet Applications (RIA)"*. En base a esto último, y ante el advenimiento y constante crecimiento del ambiente móvil nuevas soluciones necesitan ser propuestas e implementadas. Teniendo en cuenta dicha situación se establecen las Aplicaciones Móviles Enriquecidas (RMAs - Rich Mobile Applications). Una de las características de este tipo de aplicaciones móviles es brindar funcionalidades enriquecidas. Para posibilitar dichas funcionalidades la integración del ambiente móvil a la nube se considera fundamental [2] [3]. Existen investigaciones que consideran un afianzamiento cada vez mayor de la integración entre dispositivos móviles y la nube a razón de la superación de las limitaciones presentes. En tal sentido, existen diversas propuestas que buscan solucionar las limitaciones en el área, como por ejemplo el mejoramiento de una red poco confiable como lo es la inalámbrica, protocolos y procesos de comunicación más eficientes entre otras cuestiones [1].

Más allá de lo hasta aquí mencionado, a continuación hacemos referencia a un problema crítico. El mismo es la dificultad de portabilidad entre distintas plataformas. Existe una cantidad considerable de plataformas, las cuales engloban a su vez sus propios sistemas operativos, librerías, lenguajes de programación, métodos y protocolos de comunicación remota y otras características. Todo ello acarrea mayor tiempo, costo y esfuerzo para el desarrollo de distintas versiones (que se adapten a cada plataforma y a cada proveedor) de una misma aplicación. La dificultad de portabilidad se da tanto en el ambiente móvil [1] [2] [4] [5] [6], como en la nube, donde el problema es agudizado por el *vendor lock-in*[4] [4] [7] [8].

En ese sentido, el Desarrollo Dirigido por Modelos (MDD - Model Driven Development), y más específicamente la Arquitectura Dirigida por Modelos (MDA - Model Driven Architecture) se presenta como una alternativa adecuada para tratar dicha problemática, en vista de que es un enfoque de desarrollo de software que tiene como una de sus motivaciones principales el mejoramiento de la portabilidad, lo cual se traduce en la consideración de modelos independientes de plataforma y reglas de transformación que permiten generar distintas implementaciones a partir de los mismos modelos, para consecuentemente, obtener beneficios como el incremento de productividad y la adaptación a cambios tecnológicos entre otros [9] [10]. Existen varias propuestas dirigidas por modelos como WebRatio Mobile Platform [11], MD^2 [12], y MobiCloud [13]. Sin embargo, en base al estudio del estado del arte, donde hallamos cuestiones poco consideradas o directamente no consideradas, proponemos encarar el desarrollo de aplicaciones móviles con funcionalidades enriquecidas a partir del Enfoque Web Orientado por Modelos (MoWebA - Model Oriented Web Approach), el cual está basado en MDA, y que a diferencia de varios otros enfoques y propuestas de desarrollo presenta propiedades de modelado, como la navegación jerárquica y el establecimiento de un nivel de modelado de arquitectura específica [14], las cuales podrían constituirse en características adecuadas y oportunas que generen mayores beneficios en el desarrollo de las aplicaciones móviles con funcionalidades enriquecidas. En adición a dichas propiedades contemplamos otras cuestiones, poco consideradas en el estado del arte, como el modelado unificado y generación de código para los ambientes móvil y de la nube.

[1] Gartner, enlace: `http://goo.gl/O2OtSI`
[2] Canalys, enlace: `http://goo.gl/7vU8V1`
[3] Morgan Stanley, enlace: `https://goo.gl/Ifznvz`; eMarketer, enlace: `http://goo.gl/TzfNaj`
[4] Detallamos este problema en la sección 2.2.

Complementariamente al enfoque MDA, considerando las dificultades de portabilidad del lado de la nube, concretamente el problema del *vendor lock-in*, optamos por el seguimiento de una implementación de código abierto considerando que las propiedades de disponibilidad de código fuente y la adopción de estándares abiertos brindan mayor flexibilidad para tratar dicho problema.[5]

Los aspectos de implementación de la presente propuesta se centrarán en las aplicaciones móviles colectoras de datos, las cuales son un tipo específico de aplicaciones que se enmarcan dentro del esquema de aplicaciones móviles con funcionalidades enriquecidas.

Es oportuno mencionar que nuestra propuesta se enmarca dentro de un proyecto de investigación, de nuestra universidad, sobre el desarrollo de software dirigido por modelos, cuyo objetivo principal es explorar los beneficios de la aplicación de dicho paradigma. En adición, mencionamos igualmente que hemos remitido el estado del arte como artículo a una conferencia, cuya aprobación sigue pendiente al momento de la redacción del presente documento.

El resto del contenido del presente documento lo disponemos de la siguiente manera: en la sección 2 presentamos el contexto y los conceptos más relevantes para la comprensión del presente proyecto. En la sección 3 describimos el estado del arte del área de interés a través de un conjunto de propuestas actuales de solución. Hemos llevado a cabo dicho estado del arte a través de un mapeo sistemático de la literatura. En la sección 4 resumimos la problemática y la propuesta de solución, para finalmente en la sección 5, describir los avances de las actividades para llevarla a cabo.

2 Contexto de la Propuesta

Esta sección contiene un conjunto de conceptos y consideraciones que permiten el establecimiento del contexto de nuestra propuesta. Primero presentamos a las aplicaciones móviles enriquecidas como un marco contextual para luego explicar convenientemente qué involucran las funcionalidades enriquecidas. Después llevamos a cabo una breve radiografía de conceptos de la nube, ya que esta es un elemento esencial en el esquema de aplicaciones presentado. A continuación describimos de manera sintética algunas cuestiones relacionadas a la portabilidad, tanto en el ambiente móvil como de la nube, y los desafíos que presenta haciendo referencia a la problemática a encarar. Luego detallamos aspectos sobre el estilo de arquitectura REST para el establecimiento de comunicación y acceso a la nube. Posteriormente, presentamos a las aplicaciones móviles colectoras de datos como un tipo de aplicaciones más específico dentro del esquema aludido en la primera subsección, ya que sobre dicho tipo de aplicación centraremos la implementación del presente trabajo. Finalmente, describimos el paradigma de desarrollo de software a seguir, especificando a MoWebA como el enfoque del cual parte nuestra propuesta.

2.1 Aplicaciones Móviles con Funcionalidades Enriquecidas

Adoptamos el término funcionalidades enriquecidas a partir del concepto de las RMAs, el cual engloba ciertas características que debe cumplir una aplicación móvil para proveer al usuario una experiencia enriquecida superando o lidiando adecuadamente con las limitaciones del ambiente móvil. Una aplicación con funcionalidades enriquecidas basa las mismas en las capacidades de la nube.

Aplicaciones Móviles Enriquecidas

La definición de las RMAs propuesta por Abolfazli et al. [1], es la siguiente:

"Las aplicaciones móviles enriquecidas son aplicaciones móviles energéticamente eficientes, de diseño multi-capa, con funcionalidades en línea originadas de la convergencia de la computación en la nube, la web futura y las inminentes tecnologías de comunicación destinadas a la provisión de una experiencia de usuario enriquecida a través de una alta funcionalidad, una interacción inmersiva, un bajo tiempo de respuesta a usuario sobre un ambiente inalámbrico confiable permitiendo reconocimiento de contexto, uso sin conexión, portabilidad entre distintas plataformas y ubicuidad de datos".

El área de las aplicaciones móviles enriquecidas se encuentra sujeta a diversas investigaciones que buscan justamente encontrar soluciones que permitan ofrecer una interfaz de usuario amigable y atractiva, una amplia gama de funcionalidades, una alta capacidad de interacción, la portabilidad entre distintas plataformas sin mayores modificaciones y más, todo ello superponiéndose a las limitaciones presentadas como el reducido tamaño del dispositivo en sí y por ende de la pantalla, las diferencias críticas entre plataformas, la dependencia sobre una red de comunicación poco confiable como lo es la inalámbrica, la acotación de recursos computacionales, de almacenamiento, de energía y las dificultades propias de la movilidad como la disminución de la concentración del usuario, el cambio de los puntos de conexión, la alta exposición en cuanto a seguridad.

Las Funcionalidades Enriquecidas y la Nube

Un aspecto clave y fundamental para encarar varios de los desafíos que implican las aplicaciones móviles enriquecidas es la nube [2] [3]. Y sobre este punto existen investigaciones y proyectos que proponen una adaptación de la nube

[5] Más información en: `https://goo.gl/mOCr8P`

tradicional a una adecuada al entorno móvil, denominada *Mobile Cloud Computing*, donde esta contemple aspectos propios como la movilidad, la baja calidad de conexión y la limitación de recursos [15] [4] [16] [17] [18]. En concreto, la nube se constituye en un extensor de las capacidades móviles. Citando como ejemplos se comprende el acceso a más capacidad de: i) *almacenamiento*, considerando además el hecho de mantener datos delicados almacenados remotamente y no en forma local añadiendo un factor de seguridad. ii) *computación*, lo cual a su vez podría constituirse en un ahorro de energía significativo si se tratan con procesos complejos.

En Abolfazli et al. se menciona que desde el punto de vista del usuario, las funcionalidades enriquecidas son aquellas proporcionadas independientemente a las restricciones de recursos del dispositivo en sí. En ese sentido, la nube se constituye en una opción que colabora con la provisión de funcionalidades enriquecidas [1].

Consideramos entonces a la nube como un componente clave en el esquema de las aplicaciones móviles con funcionalidades enriquecidas, por lo cual es necesario hacer una breve radiografía de los conceptos asociados a la misma. Nos enfocamos en conceptos referidos a los tipos de nube que sirven como base para la comprensión de los aspectos relacionados a la portabilidad en dicho ambiente.

Existen dos tipos de modelos de despliegue de nube [7]: i) *nube pública*: es la infraestructura de nube ofrecida por una organización (proveedor) como un conjunto de servicios al público en general; ii) *nube privada*: es la infraestructura de nube sobre la cual opera una sola organización. Dicha infraestructura puede pertenecer a la misma organización o a un tercero (proveedor).

Un término muy ligado a la nube es *cloud computing* o *computación en la nube*, el cual es un modelo de acceso ubicuo, conveniente y bajo demanda a un conjunto de recursos de computación compartidos y configurables (ejemplos: redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser proveídos rápidamente y puestos en marcha con un mínimo esfuerzo de gestión o de interacción con el proveedor de dichos recursos [19]. Ejemplos de proveedores: *Amazon Web Services, Microsoft Azure Platform, Google App Engine, Salesforce* [20]. La computación en la nube es suministrada por terceros y contempla la provisión tanto de modelos de nube pública como privada [15].

Existen tres niveles de servicios proporcionados por la computación en la nube [7]: i) *infraestructura* o *IaaS*: es la capacidad proveída al consumidor que contempla procesamiento, almacenamiento, redes, y otros recursos de computación fundamentales sobre los cuales el consumidor es capaz de desplegar y correr software arbitrario, lo cual puede incluir sistemas operativos y aplicaciones; ii) *plataforma* o *PaaS*: es la capacidad proveída al consumidor para desplegar sobre la infraestructura de nube aplicaciones creadas o adquiridas desarrolladas a partir de lenguajes de programación y herramientas soportadas por el proveedor; iii) *software* o *SaaS*: es la capacidad proveída al consumidor para la utilización de aplicaciones del proveedor que corren sobre la infraestructura de nube.

Según lo descrito, el interés del presente trabajo se enfoca en el desarrollo de aplicaciones móviles con funcionalidades enriquecidas portables a nivel de plataforma.

2.2 Los Desafíos de la Portabilidad

Hemos mencionado implícitamente más arriba la necesidad de mejorar la portabilidad entre plataformas tanto del lado móvil como de la nube. La portabilidad se define en términos de la facilidad de migración de software de un ambiente a otro [8]. Para nuestro trabajo, los ambientes serían las plataformas tanto del lado móvil como de la nube. A continuación especificamos cuestiones relativas a ambos casos (móvil y nube) y a la interfaz de comunicación entre los mismos.

Del Lado Móvil

Las plataformas móviles populares difieren en aspectos arquitecturales de software y hardware, por lo cual la portabilidad se hace complicada [2] [5]. Existen diferencias a nivel de sistemas operativos, lenguajes de programación, librerías, métodos o funcionalidades. Consecuentemente los desarrolladores deben escribir diferentes versiones de código para una sola aplicación lo cual aumenta el costo y el tiempo de desarrollo [1] [4] [6].

Del Lado de la Nube

A nivel de plataforma, se debe seguir un estilo de programación específico y hacer uso de las librerías y tecnologías ofrecidas por los proveedores. La migración ente dos plataformas (propietarias) requiere un total proceso de re-ingeniería. El desarrollador debe conocer los detalles de cada plataforma y sus librerías, lo cual hace que la portabilidad (entre nubes públicas, entre nubes públicas y privadas, entre proveedores) sea costosa y complicada [8]. Dicho problema se denomina *vendor lock-in* o *proprietary lock-in*. Donde cada proveedor de computación en la nube tiene sus propias políticas, sus propios servicios y entornos soportados por lo cual la migración (entre proveedores y entre nubes públicas y privadas) se hace complicada [4] [7] [8].

Por otra parte, al contemplar el lado móvil y la nube se debe de considerar la interfaz de comunicación entre ambos. La misma se implementa a través de *APIs*, las cuales pueden variar según las plataformas.[6]

[6] Backendless, enlace: `https://goo.gl/FHjIxX`

2.3 El Estilo de Arquitectura REST

Teniendo en cuenta la contemplación de la nube en el esquema de aplicaciones móviles que hemos presentado, son necesarios el establecimiento de la comunicación entre los dos ambientes (móvil y nube) y la consideración del diseño en la nube. Para dichas cuestiones una opción que consideramos conveniente es el estilo de arquitectura REST.

REST (REpresentational State Tranfer) *"es un conjunto coordinado de restricciones arquitecturales, considerado como un modelo abstracto de la arquitectura Web, utilizado como guía para el rediseño y la definición del protocolo http y de los identificadores uniformes de recursos* (*URIs*). *Dichas restricciones arquitecturales buscan minimizar la latencia y la comunicación de red, y al mismo tiempo maximizar la independencia y la escalabilidad de las implementaciones de los componentes"* [21].

Para una mayor comprensión de REST, a continuación describimos algunos de sus conceptos fundamentales, definiciones según Richardson et al. [22]: i) *recurso*: es cualquier cosa que es lo suficientemente importante o útil como para ser referenciada o nombrada. Usualmente un recurso es un documento, una base de datos o el resultado de la ejecución de un algoritmo. La referencia o nombramiento de un recurso se debe dar a través de un URL; ii) *URL*: Uniform Resource Location, dirección global de un recurso. Si se le da un URL a algo, ese algo se convierte en un recurso. Se hace posible la comunicación entre el cliente y el servidor a través de la URL; iii) *representación*: nunca se transfiere el recurso en sí sino representaciones, y estas pueden referirse al estado actual del recurso o cambios a aplicarse sobre un recurso. La representación es una descripción útil y se puede dar en distintos formatos como *JSON*,[7] *XML*[8] y otros. La transferencia de representaciones se da en ambos sentidos, del servidor al cliente y del cliente al servidor; iv) *métodos o verbos HTTP*: sólo se puede interactuar a través del protocolo HTTP, es decir a través de mensajes estándares HTTP. Los cuatro más utilizados son: GET, PUT, POST, DELETE. Estos métodos operan sobre la URL; v) *idempotencia*: se refiere a que el envío de dos o más veces de una petición tiene el mismo efecto sobre el estado de un recurso. Las operaciones idempotentes son: GET, PUT, DELETE; vi) *hipermedia*: es la manera en la cual el servidor presenta al cliente qué operaciones HTTP podría efectuar en el futuro. Es como un menú proveído por el servidor y desde el cual el cliente tiene la libre elección.

En cuanto a las restricciones de estilos arquitecturales, REST contempla las siguientes [21]: i) *arquitectura cliente-servidor*: donde se resalta el principio de Separación de Intereses (*Separation of Concerns*), separando los aspectos de la interfaz de usuario de la gestión y almacenamiento de datos, lo cual permite una evolución independiente de componentes; ii) *comunicación sin estado*: donde cada petición del cliente contiene toda la información necesaria para entenderla y no se depende de ningún contexto almacenado en el servidor. El estado de sesión es mantenido enteramente en el cliente, es decir al servidor no le importa en cual estado se encuentra el cliente. Esta restricción induce a tres propiedades: a) *Visibilidad*; b) *Confiabilidad*; c) *Escalabilidad*; iii) *caché*: que el dato dentro de una respuesta correspondiente a una petición está etiquetado explícita o implícitamente como *cacheable* o no *cacheable*. La ventaja es la capacidad potencial de eliminar parcial o completamente algunas interacciones mejorando la eficiencia, la escalabilidad, y la percepción de usuario con relación al desempeño reduciendo la latencia promedio de una serie de interacciones. Una limitación importante es que un dato *cacheable* y considerado viejo necesita ser obtenido nuevamente desde el servidor; iv) *interfaz uniforme entre componentes*: es el principal rasgo que diferencia a REST de otros estilos arquitecturales basados en redes. Aplicando el principio de Generalidad (*Generality*) de la Ingeniería del Software a la interfaz de componente, la arquitectura global del sistema es simplificada y la visibilidad de las interacciones es mejorada. Las implementaciones son desacopladas de los servicios que proveen lo cual alienta la independencia en la capacidad de evolución. La desventaja es que una interfaz uniforme degrada la eficiencia en el sentido de que la información es transferida en una manera estandarizada en lugar de una específica que se adecue a las necesidades de la aplicación. La interfaz REST fue diseñada para ser eficiente con transferencias de datos hipermedia de *granularidad* gruesa, optimizando el caso común de la web pero resulta en una interfaz no óptima para otras formas de interacción arquitectural. A fin de obtener una interfaz uniforme, REST es definido por cuatro restricciones de interfaz: a) identificación de recursos; b) manipulación de recursos a través de representaciones; c) mensajes auto-descriptivos; d) hipermedia como el motor de estado de la aplicación.

Otras dos restricciones son un sistema basado en capas y código sobre demanda; aunque esta última es opcional.

Por otra parte, es importante destacar que REST junto a *SOAP* o *Simple Object Access Protocol*[9] son los paradigmas más predominantes para la comunicación web [23]. SOAP es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Ahora bien, REST toma ventaja sobre SOAP. Algunas de las razones son las siguientes [24]: i) *complejidad*: la *serialización* o *des-serialización* de lenguajes nativos a mensajes SOAP toma un tiempo considerable; ii) *redundancia*: existe mucha información en los mensajes SOAP y su descripción puede llegar a ser redundante. Esto incrementa el volumen de comunicación de red.

Inclusive un simple mensaje SOAP contiene una porción grande de datos en XML, lo cual consume mucho ancho de banda y puede causar un mayor tiempo de latencia de red [25]. En [26], una comparación de desempeño concluye que una implementación guiada por el estilo REST toma ventaja sobre una basada en SOAP.

[7] JavaScript Object Notation, enlace: `http://goo.gl/ErtWK1`
[8] eXtensible Markup Language, enlace: `http://goo.gl/QwUt1r`
[9] *Simple Object Access Protocol*. Especificaciones de *SOAP*: `http://goo.gl/P2JxZu`

En resumen, resaltamos las siguientes propiedades del estilo de arquitectura REST teniendo en cuenta el diseño en la nube y la comunicación de ésta con el ambiente móvil: i) diseño en la nube basado en *recursos*; ii) interfaz de comunicación *uniforme*; iii) *simplicidad* y *eficiencia* en la interacción cliente-servidor.

2.4 Aplicaciones Móviles Colectoras de Datos

Centraremos la implementación de este proyecto final de carrera en un tipo de aplicaciones móviles más específico: las aplicaciones Móviles Colectoras de Datos, del inglés *Mobile Data Collection* (*MDC*). Las mismas siguen el esquema híbrido móvil/nube, donde el principal recurso remoto es el almacenamiento. Presentamos esta sub-sección para describir en qué consisten las aplicaciones sobre las cuales centraremos la fase de implementación.

Las MDC son aplicaciones móviles que llevan a cabo una recopilación selectiva de información estructurada utilizando dispositivos como telfonos inteligentes o tabletas.

La información recopilada es transferida a una base de datos centralizada para su almacenamiento. Existen dos tipos de comunicación para la transferencia de datos:[10] i) la primera no contempla retardo; ii) la segunda puede llegar a tener un retardo y se concreta a través de un método de sincronización.

Los datos recolectados en los centros de almacenamiento se someten posteriormente a una serie de análisis y estudios de interés. Se citan algunos tipos de aplicaciones:[11] i) inspecciones, auditorías e informes; ii) encuestas; iii) programas de control de salud o *mHealth*.

Algunas de las ventajas de estas aplicaciones son las siguientes:[12] [13] i) El registro de datos se lleva cabo a través de formularios electrónicos por lo cual se ahorran *papel* y procedimientos manuales, lo cual en adición a la conexión directa con la base de datos centralizada aumenta la eficiencia del proceso; ii) Disponibilidad *instantánea* de datos por medio de internet. Se tiene un acceso a los datos tanto inmediato como global; iii) Mejoramiento de la calidad de datos mediante filtros y validaciones de formulario.

2.5 Enfoque Dirigido por Modelos

El enfoque de desarrollo de software que seguiremos en este proyecto final de carrera es el dirigido por modelos. Los modelos pueden ser utilizados para distintos propósitos, uno de ellos es el modelado para el desarrollo de artefactos de software. Este enfoque es conocido como Ingeniería de Software Dirigida por modelos (*MDSE* - Model Driven Software Engineering) o directamente Ingeniería Dirigida por Modelos (*MDE* - Model Driven Engineering) .

MDE puede ser definida como un enfoque para la aplicación de los beneficios del modelado a las actividades de la Ingeniería del Software donde los modelos se transforman en elementos fundamentales del proceso de desarrollo. Esta metodología se especifica de la siguiente manera [10]: i) *conceptos*: los conceptos fundamentales son *modelos* y *tranformaciones*; ii) *notaciones*: los modelos y las transformaciones se expresan en una notación denominada *lenguaje de modelado*, el cual a su vez es definido por el proceso de *metamodelado* a través de *metamodelos*; iii) *procesos y reglas*: los tipos de modelos, el orden y los niveles de abstracción son definidos dependiendo del tipo de software a producir; iv) *herramientas*: se refieren a los entornos de desarrollo que permitan definir los modelos y las transformaciones como también a compiladores e intérpretes para ejecutarlos y producir los artefactos finales de software. Detallamos a continuación, paradigmas más específicos de MDE: el Desarrollo Dirigido por Modelos y la Arquitectura Dirigida por Modelos.

Desarrollo Dirigido por Modelos (MDD - Model Driven Development).

MDD es un paradigma, el cual se refiere netamente a las actividades de desarrollo y que utiliza a los modelos como artefactos primarios del proceso de desarrollo, donde este es implementado a través de generaciones (semi)automáticas a partir de los modelos [10]. Algunos de los beneficios de la aplicación de MDD son los siguientes[9]: i) *incremento de la productividad*; ii) *adaptación a los cambios tecnológicos*; iii) *adaptación a los cambios de requisitos*; iv) *consistencia*; v) *re-utilización*; vi) *mejoras en la comunicación con el usuario*; vii) *mejoras en la comunicación con los desarrolladores*; viii) *captura de experiencia*; ix) *los modelos son productos de larga duración*; x) *posibilidad de demorar decisiones tecnológicas*;

Arquitectura Dirigida por Modelos - (MDA - Model Driven Architecture).

MDA es la visión particular de MDD propuesta por el Object Management Group (OMG) que se basa en el uso de estándares. MDA puede ser considerada como un subconjunto de MDD [10].

MDA es una familia de estándares, de los cuales *Meta Object Facility* (*MOF*) y *Unified Modelling Language* (*UML*) son universalmente aceptados[14]. A continuación una breve descripción de algunos de los estándares:

[10] Nomad, `http://goo.gl/n32wUh`
[11] Poimapper, `http://goo.gl/FmLEqr`
[12] Captura Forms, `http://goo.gl/CIHwLy`
[13] U.S. Agency for USAID, `https://goo.gl/XozMWs`
[14] OMG-MOF-UML, enlace: `http://goo.gl/utxGiQ`

- Estándares relacionados a la transformación y a la gestión fundacional de modelos: i) *MOF*: provee la base fundamental para la Arquitectura Dirigida por Modelos. Usa los *metamodelos* especificados en UML para describir lenguajes de modelado como objetos interrelacionados. Es un *subconjunto* de UML para la construcción de *metamodelos*; ii) *XML Metadata Interchange* (*XMI*): especifica como intercambiar modelos usando una estructura *XML* específica; iii) *Query View Transform* (*QVT*): estándar para la transformación entre modelos basado en patrones.

- Lenguaje de Modelado Unificado y Perfiles, *UML*: es una familia de notaciones de modelado unificadas por un metamodelo común cubriendo múltiples aspectos de modelado de sistemas y negocios. Una característica clave de *UML* es extender *UML* mediante perfiles o *profiles*. Estos pueden estar adaptados a requerimientos específicos, extendiendo y adaptando las capacidades del núcleo de *UML*.

Enfoque Web Orientado por Modelos - (MoWebA - Model Oriented Web Approach)

La propuesta dirigida por modelos específica sobre la cual enfocamos este trabajo es MoWebA. La misma está basada en la Arquitectura Dirigida por Modelos.

MoWebA es una propuesta centrada en roles de navegación para aplicaciones web, que define aspectos metodológicos y los complementa con todo un entorno que incluye modelado, herramientas de transformación, generación automática de código, uso de estándares y una robusta arquitectura [14].

Las consideraciones más importantes sobre las cuales se apoya MoWebA son: i) el modelado orientado a la navegación podría ayudar a simplificar los modelos de las aplicaciones web; ii) la automatización puede ser un instrumento muy útil para simplificar el proceso de desarrollo; iii) la adopción de estándares facilitaría la interoperabilidad entre modelos, métodos, reglas de transformación y herramientas; iv) teniendo en cuenta la evolución del entorno web es muy importante el mejoramiento del desarrollo de las aplicaciones web actuales.

MoWebA propone un modelado centrado en la navegación a diferencia de varias metodologías del área de desarrollo de aplicaciones web, las cuales diseñan sus modelos navegacionales partiendo de los modelos conceptuales (por ejemplo modelos estructurales). Sin embargo, MoWebA considera que la manera en la cual la información es organizada y estructurada dentro del sistema no es necesariamente la manera en la que los usuarios acceden a ella. Por esa razón propone una navegación más *function-oriented* que *data-oriented*, lo cual previene una conexión directa sobre un modelo de datos específico. Esta característica proporciona un esquema más adecuado para el diseño de una navegación basada en la interacción del usuario o del contexto. Esto último a su vez, podría ayudar a simplificar los modelos de las aplicaciones web [14]. Sobre el mismo punto, la gran mayoría de los métodos web plantean grafos genéricos como mapas navegacionales. En este caso se propone definir la estructura navegacional como una jerarquía de unidades funcionales. Con este aporte es posible generar varios niveles de exploración a través de menús y sub-menús, y mantener ubicado al usuario a través del historial de navegación.

MoWebA adopta el enfoque MDA identificando tres niveles de abstracciones para el modelado: i) El espacio del problema $->$ Modelos Independientes de Computación (CIM - Computation Independent Model) + Modelos Independientes de Plataforma (PIM - Platform Independent Model); ii) El espacio de modelado de la solución $->$ Modelos Específicos de Arquitectura + Modelos Específicos de Plataforma; iii) La definición de código fuente $->$ Modelos Específicos de Implementación y Código Manual.

MoWebA sugiere el soporte de la evolución de las aplicaciones a través del *Modelado Específico de Arquitectura* por sus siglas en inglés *ASM*, permitiendo extender su PIM de acuerdo a nuevos requerimientos específicos.

MoWebA define dos procesos principales complementarios: uno relacionado a las actividades de modelado y otro relacionado a actividades de transformación. El proceso de modelado incluye todas aquellas actividades necesarias para obtener todos los diagramas para la especificación completa del sistema a generar. El proceso de transformación se refiere a los pasos, técnicas y herramientas que permiten transformaciones *M2M* (*Modelo a Modelo*) y *M2T* (*Modelo a Texto*), basado en el enfoque MDA.

3 Estado del Arte

Presentamos a continuación un conjunto de propuestas MDD destinadas al desarrollo de aplicaciones móviles. Para obtener dicho conjunto de propuestas hemos llevado a cabo el estado del arte a través de un *mapeo sistemático de la literatura*.

Dicho mapeo sistemático es un estudio de la literatura que comprende una etapa de planificación previa a la búsqueda y recopilación de información. En dicha etapa previa se definen en primer lugar las preguntas de investigación, las cuales guían el resto del mapeo, y se diseña un protocolo de búsqueda y selección de estudios, en base a dicha planificación se efectúa la ejecución del proceso, lo cual se constituye en la siguiente etapa del mapeo. El principal objetivo de este es brindar una visión global sobre un tema de interés (con evidencias empíricas o no), identificando cantidades, tipos de investigación y resultados disponibles a través de un proceso riguroso y sistemático [27] [28].

Buscamos entonces, a partir del estudio de mapeo sistemático, clasificar las propuestas que abordan el tema de interés

e identificar las cuestiones cubiertas y no cubiertas, para posteriormente, contestar las preguntas de investigación definidas.

Preguntas de Investigación

Hemos confeccionado seis preguntas de investigación, estas expresan los puntos de inters sobre los cuales enfocamos el estudio de mapeo. El resto del trabajo se desarrolla en base a estas preguntas:

1. ¿Cuáles son las soluciones dirigidas por modelos para el desarrollo de software relacionadas al problema de la portabilidad entre plataformas que comprenden el ambiente móvil y/o de la nube?

   En las propuestas clasificadas:

2. ¿Qué aportes y limitantes presentan estas soluciones con relación a la dificultad de portabilidad entre plataformas?
3. ¿Se identifican evidencias que muestren la adopción del modelado navegacional jerárquico?
4. ¿Se identifica el establecimiento de una distinción entre los niveles de modelado: PIM y ASM?
5. ¿Se identifica una clara separación entre los detalles de la capa de presentación y comportamiento para el diseño de las aplicaciones?
6. ¿Presentan las soluciones clasificadas algún tipo de evaluación?, en caso afirmativo, ¿son positivos los resultados de las evaluaciones de sus propuestas?

Cadenas de Búsqueda

Posteriormente, derivamos términos a partir de la primera pregunta de investigación para establecer una cadena inicial de búsqueda. Luego, modificamos dicha cadena gradualmente por sinónimos, abreviaciones y supresiones para obtener las restantes. El cuadro 1 muestra las cadenas definidas a partir de las cuales hemos obtenido las propuestas.

Cuadro 1: Cadenas de búsqueda

Cadenas de Bsqueda
Model Driven Mobile Cloud Application Software Portability[15]
MDD Mobile Cloud Application Software Portability
Model Driven Mobile Application Portability
Model Driven Mobile
Model Driven Mobile Application Cross Platform
Model Driven Android
Architecture Driven Portability
Model Driven Windows Phone

Criterios de Inclusión

Incluimos estudios y propuestas que: i) consideran el problema de la portabilidad a nivel de plataforma; ii) contemplan algunas de las siguientes propiedades: una navegación jerárquica, una separación entre los niveles de modelados PIM y ASM, una clara separación del diseño de la capa de presentación de la de comportamiento; iii) presentan algunos de los siguientes elementos: meta-modelos, perfiles, reglas de transformación, marcos de trabajo, herramientas.

Criterios de exclusión

Excluimos estudios y propuestas que: i) no están en lengua inglesa; ii) no están relacionados a la Ingeniería del Software; iii) no incluyan a los modelos como artefactos importantes del desarrollo de software; iv) no tengan relación con el desarrollo de aplicaciones móviles y/o de la nube; v) sean muy resumidos, sin muchos detalles de la implementación de sus propuestas, que no presenten aspectos concretos relacionados a meta-modelos, marcos de trabajo, herramientas, reglas de transformación.

No se ha establecido una restricción explícita relacionada a los años de publicación, pero consideramos con menos prioridad a los trabajos de más de 5 años.

Protocolo del Mapeo

Describimos a continuación el protocolo del mapeo en base al cual llevamos a cabo el proceso de búsqueda y la selección de estudios.

1. Derivar términos de las preguntas de investigación para elaborar una cadena de búsqueda inicial:
   - Utilizar sinónimos, abreviaciones, composiciones y supresiones para obtener otras cadenas.
2. Consultar las fuentes con los términos de búsqueda elaborados (tener en cuenta como máximo los 100 primeros resultados[16]):

[16] Los 100 primeros como máximo porque entre los resultados arrojados por las búsquedas pocos coincidían con nuestra área de estudio.

- Fuentes a consultar: bibliotecas digitales (IEEEXplore, ACM y Science Direct) y servicios de indexación (Google Scholar y DBLP) comprendiendo Revistas y Conferencias.

3. Aplicación del procedimiento de Selección de Estudios. Dicho procedimiento consiste en la aplicación de los criterios de selección en tres etapas:

   a) Primera Etapa: concentrada sobre los títulos, las palabras claves y los resúmenes de los resultados arrojados por los términos de búsqueda. Iteración entre los puntos 1, 2 y 3(a) hasta la finalización de las búsquedas en las fuentes de consulta.

   b) Segunda Etapa: incorporación de las secciones de "Introducción" y de "Conclusiones" en el análisis de los estudios seleccionados en la etapa previa.

   c) Tercera Etapa: análisis de texto completo sobre los estudios seleccionados en 3(b).

4. Sobre el conjunto de estudios obtenidos por los pasos 1, 2 y 3 del protocolo, aplicación de la estrategia de búsqueda *Snowballing Search* con el objetivo de recabar mayor información sobre los mismos y obtener, consecuentemente, mayor comprensión de sus propuestas. Las pasos claves del *Snowballing Search* son: i) identificar un conjunto inicial A de estudios (identificación de este conjunto a través de los pasos 1, 2 y 3); ii) identificar más estudios usando las listas de referencias de los estudios pertenecientes al conjunto A; iii) identificar más estudios a través de las listas de estudios, proveídas por las bibliotecas y fuentes de indexación digitales, que citan a los estudios pertenecientes al conjunto A; iv) repetir los pasos *ii)* y *iii)* hasta no encontrar más estudios de interés.

Análisis de las Propuestas y Resultados

En el cuadro 2 detallamos y comparamos las propuestas recopiladas en base a varios aspectos. Aclaramos a continuación la correspondencia de estos aspectos con las preguntas de investigación. En base a los mencionados aspectos y sus correspondientes valores, elaboramos las respuestas a las preguntas de investigación: i) pregunta 1:{son las propuestas identificadas, de P1 hasta P11}; ii) pregunta 2:{componentes remotos abarcados, diseño unificado de módulos locales y remotos, tipo de comunicación remota, método de sincronización inteligente, generación de código nativo(móvil), plataformas destino (móvil y nube), integración con otros servicios en la nube}; iii) pregunta 3:{consideración de modelado de navegación}; iv) pregunta 4:{diferenciación entre PIM y ASM}; v) pregunta 5:{clara separación de capas de presentación y comportamiento}; vi) pregunta 6:{presenta evaluación de propuesta, son positivos los resultados de sus evaluaciones}; vii) consideramos además aspectos complementarios y adicionales, que tienen que ver con el lenguaje de modelado (teniendo en cuenta que existen dos tipos de modelos: textuales y gráficos), editores de modelos y generadores de código, con el objetivo de identificar una tendencia o preferencia en cuanto a tipo de modelado, entornos de trabajo y acceso a la herramienta o entorno de trabajo:{tipo de lenguaje de modelado, lenguaje de modelado, herramienta de desarrollo, costo/licencia}.

A continuación, clasificamos y analizamos los resultados en base a las preguntas de investigación.

1. ¿Cuáles son las soluciones dirigidas por modelos para el desarrollo de software relacionadas al problema de la portabilidad entre plataformas que comprenden el ambiente móvil y/o de la nube?

Identificamos finalmente once estudios, los cuales se listan en el cuadro 3 verificando el cumplimiento de los criterios de selección. Estos constituyeron la respuesta a la primera pregunta de investigación. Sobre ellos llevamos a cabo un análisis para responder a las preguntas restantes. Dichos estudios describen el planteamiento de la propuesta, presentan información acerca de la implementación del lenguaje de modelado, muestran la utilización del mismo mediante ejemplos sobre los cuales llevan a cabo una evaluación y/o emiten un juicio al respecto. Dentro de los documentos recopilados, la P1 (conforme el ID del cuadro 3) se presenta como la propuesta más completa y con más peso en la industria. A ésta, le siguen las propuestas P6 y P3, las cuales contemplan además experimentos con socios de la industria.

2. ¿Qué aportes y limitantes presentan estas soluciones con relación a la dificultad de portabilidad entre plataformas?

En cuanto a los aportes que consideramos más relevantes, identificamos tres propuestas que abstraen al desarrollador de los detalles específicos de implementación de los ambientes móvil y de la nube: la P1, la P6 y la P10. De estas tres solo dos propuestas, la P6 y la P10, comprenden un modelado unificado (móvil + nube), y son lenguajes textuales. En cuanto a métodos complementarios al enfoque dirigido por modelos, la propuesta P1 contempla la generación de código móvil basado en el Apache Cordova[17] de enfoque híbrido o *native wrapper*, además de la generación de código Java estándar para el lado de la nube basando su portabilidad en el Java Server. Esta generación de aplicaciones Java es adoptada igualmente por la propuesta P6. El lenguaje de programación predominante para la generación de código del lado de la nube es Java. Solo la propuesta P1 especifica la generación de código que contemple una adaptación tanto a la nube privada como a la nube pública.

[17] Apache Cordova, enlace: https://goo.gl/QocB2x

Cuadro 2: Cuadro de resumen del estado del arte

Proyectos Dirigidos por Modelos											
Aspectos	**Proyectos**										
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**
Costo/Licencia	Comudidad Gratuito Profesional Pago	No especificado	No especificado	No especificado	No especificado	Apache License 2.0	No especificado	No especificado	No especificado	No especificado	No especificado
Componentes Remotos Abarcados	Backend	No	No	No	Backend	Backend	No	No[18]	No	Backend	No[19]
Diseño Unificado de Módulos Locales y Remotos	No.*PSIS*[20]	No.*PSIS*[20]	No se aplica	No se aplica	No se aplica	Sí	No se aplica	No se aplica	No se aplica	Sí	No se aplica
Tipo de Comunicación Remota	REST	Servicios	No se aplica	No se aplica	REST	REST	No se aplica	No se aplica	No se aplica	REST	No se aplica
Método de Sincronización Inteligente	Sí	No	No se aplica	No se aplica	No	No	No se aplica	No se aplica	No se aplica	No	No se aplica
Tipo de Lenguaje de Modelado	Gráfico	Gráfico	Gráfico	Gráfico	Textual	Textual	Gráfico	Textual	Gráfico	Textual	Gráfico
Lenguaje de Modelado	IFML-Mobile. UML + BPMN	XIS-Mobile, Basado en UML	Basado en EMF	Basado en UML	DSL	DSL-MD[2]	Basado en UML	UIML	Basado en UML	DSL basado en Ruby	Basado en UML
Diferenciación entre PIM y ASM	No	No	No	No	No	No	No	No	No	No	No
Consideración de Modelado de Navegación	Sí.[21]	Sí.[22]	No	No	No	Sí. No jerárquica	Sí (*SDE*). No jerárquico.	Sí. No jerárquico	No	No	No
Clara Separación de Capas de Presentación y Comportamiento	No[23]	No[24]	Sí(MVC)	No se aplica ,NMCP[25]	Sí(MVC)[26]	Sí(MVC)	Sí	No se aplica, CCP[27]	Sí	Sí(MVC)[28]	No se aplica, CCC[29]
Herramienta de Desarrollo	WebRatio Web y Mobile Platforms	Sparx+EMF[30]	GMF Xtend,*BNE*[31]	MAG[32]	Editor generado por Xtext,*BNE*[31]	Xcode,BNE[31]	IBM Rational, Artisan Studio, Stateflow, Sparx	MODIF	Papyrus	Textual y Gráfica(básica).	*BNE*[31],(Ecore y ATL)
Generación de Código Nativo (Móvil)	No.[33]	Sí	Sí	Sí	No se aplica	Sí	Sí	Sí	Sí	Sí	Sí
Plataforma(s) Destino (Móvil)	A. Cordova, Ionic AngularJS, SQLite	Android, Windows P.	Android	Android, Windows P.	No se aplica	Android, iOS	Android, Windows P.	Android, iOS	Android	Android, Blackberry	Andorid, Windows P.
Plataforma(s) Destino (Nube)	Estándar Java-Backend services	No se aplica	No se aplica	No se aplica	GAE and Windows Azure (Java)	JEE App-JPA (Java)	No se aplica	No se aplica	No se aplica	AWS Cloud, GAE	No se aplica
Integración con otros Servicios en la Nube	Sí	No especifica	No especifica	No	No especifica	No especifica	No	No especifica	No	S, Salesforce	Sí 34
Presenta Evaluación de Propuesta	Sí	No[35]	Sí	Sí	Sí	Sí	Sí	Sí	No[36]	Sí	No
Son Positivos los Resultados de sus Evaluaciones	Sí	No se aplica	Sí	Sí	Sí	Sí	Sí[37]	Sí	No se aplica[38]	Sí	No se aplica

[18]Utiliza servidores, pero para almacenar información del perfiles. Dichos servidores están relacionados al frontend.
[19]Lo que sí utiliza son las interfaces de servicios de cada plataforma(como el GPS).
[20]PSIS: Por proyectos separados, conexin a travs de interfaces de servicios.
[21]Modelado mediante diagramas de flujo. No jerárquica.
[22]Considera un diagrama exclusivo para la navegación. No jerárquica.
[23]Se mezclan detalles de la capa de presentación y flujo/navegación.
[24]Se mezclan detalles de interfaz de usuario y acciones.
[25] NMCP: No contempla el modelado de la capa de presentación.
[26]El modelado se limita a operaciones básicas CRUD.
[27]CCP: Se centra en el modelado de la capa de presentación
[28]Enfoque básico para el modelado de la interfaz.
[29]CCC: Centrado en la capa de comportamiento.
[30]Basado en Sparx Enterprise Architect Model-Driven Generation Technologies and EMF.
[31]BNE: Basado en Eclipse.
[32]MAG: Mobile App Generator. No especifica herramienta de modelado.
[33]Híbrido Native Wrapper, generación de código no nativo.
[34]Utiliza servicios s/ plataforma móvil destino.
[35]Sólo se especifica un caso de estudio simple (To-Do list App) para mostrar la implementación del modelado.
[36]Se implementa un caso de estudio para mostrar la utilización del modelado.
[37]En general, son positivos los resultados, aunque se aclara una generación de variables y métodos redundantes.
[38]Aunque no presentó una evaluacón propiamente, en su generación de código a modo de ejemplo aclara que el código generado no es eficiente.

Cuadro 3: Propuestas identificadas de soluciones dirigidas por modelos

ID	Propuesta de Solución	Título del Documento	Autores	Tipo	Año
P1	WebRatio Mobile Platform	Extending the Interaction Flow Modeling Language (IFML) for Model Driven Development of Mobile Applications Front End.	Brambilla, M.; Mauri, A.; Umuhoza, E.	INPROCEEDINGS	2014
P2	XIS-Mobile	XIS-mobile: a DSL for mobile applications	Ribeiro, A.; da Silva, A. R.	INPROCEEDINGS	2014
P3	Propuesta MDD para Variantes de Apps según roles	Model-Driven Development of Mobile Applications Allowing Role-Driven Variants	Vaupel, S.; Taentzer,G.; Harries, J. P.; Stroh, R.; Gerlach, R.; Guckert, M.	INPROCEEDINGS	2014
P4	Model Driven Approach for mobile Apps	A Model-driven Approach to Generate Mobile Applications for Multiple Platforms	Usman, N; Iqbal, M. Z.; Khan, M. U.	INPROCEEDINGS	2014
P5	Propuesta MDE aplicada a la Nube	Towards a model-driven approach for promoting cloud PaaS portability	da Silva, E.A. N.; Fortes, R P.; Lucredio, D.	INPROCEEDINGS	2013
P6	MD^2	Cross-platform model-driven development of mobile applications with md2	Heitktter,H.; Majchrzak, T. A.; Kuchen, H.	INPROCEEDINGS	2013
P7	Propuesta de Diseño Dirigido por Modelos para plataformas móviles	Model-driven design for the development of multi-platform smartphone applications	Botturi, G.; Ebeid, E.; Fummi, F.; Quaglia, D.	INPROCEEDINGS	2013
P8	MODIF	An efficient model-based methodology for developing device-independent mobile applications	Cimino M. G.C.A.; Marcelloni F.	ARTICLE	2012
P9	Propuesta MDE para Modelado de PSM	A Model Driven Approach for Android Applications Development	Parada, A.G.; de Brisolara, L.B.	INPROCEEDINGS	2012
P10	MobiCloud	A domain specific language for enterprise grade cloud-mobile hybrid applications	Ranabahu, A. H.; Maximilien, E. M.; Sheth, A. P.; Thirunarayan, K.	INPROCEEDINGS	2011
P11	CC for Mobile Apps on MDA	A Development of Process-Based Composite Contexts for Mobile Device Platforms Based on Model Driven Architecture	Gultawatvichai, E.; Senivongse T.	INPROCEEDINGS	2011

De entre las plataformas móviles más populares, Android es la mś generada (en nueve propuestas) y iOS la menos (en dos propuestas). Estos y otros aportes se describen cuantitativamente en el cuadro 4. En cuanto a las limitantes, identificamos que de las once propuestas ocho no contemplan el modelado y la generación de código para ambos ambientes: la P5 sólo para la nube; las P2, P3, P4, P7, P8, P9, y la P11 sólo para el lado móvil.

Cuadro 4: Aportes con relación a la problemática de la portabilidad entre plataformas

Aportes	Num	%	Observaciones
Generación de código para los dos ambientes: móvil y nube.	3/11	27.3	Dos de las tres propuestas generan una aplicación Java del lado del servidor.
Generación de código nativo para al menos dos plataformas (lado móvil).	7/11	63.6	Las siete propuestas contemplan la plataforma Android.
Generación de código para al menos dos plataformas o proveedores (lado de la nube).	2/11	18.2	Dos propuestas adicionales utilizan una implementación Java basando su portabilidad en el Java Server.
Consideración de enfoques complementarios a MDE para la problemática.	2/11	18.2	De lado móvil: Generación de aplicaciones híbridas (Native Wrapper). Del lado de la nube: Código Java estándar (Java Server).
Interfaz de comunicación estándar.	4/5	80	Interfaz de comunicación considerada: REST. De entre las propuestas que consideran una comunicación con módulos remotos.

Además algunas propuestas no contemplan ciertos aspectos o bien podrían ser enriquecidas, como las que se citan a continuación: la P7 y la P9 no muestran evidencias de la consideración del modelado de la capa de datos, además en sus conclusiones aclaran que su generación de código podría ser mejorada y/o enriquecida, por ejemplo en cuanto a eficiencia y legibilidad; la P4 y la P5 no contemplan el modelado de la capa de presentación; mientras que en la P10 el modelado de la capa de presentación podría ser enriquecido; en la P8 se presenta un esquema enfocado exclusivamente en el modelado de interfaz gráfica independiente de plataforma basado en perfiles de usuario, sin embargo no se identifica un esquema de modelado para las capas de comportamiento y de datos; en la P11 se

propone un modelado independiente de plataforma centrado en la integración de funcionalidades de reconocimiento de contexto (por ejemplo GPS), no contemplando los demás aspectos de diseño.

Podríamos decir entonces que pocas propuestas consideran el modelado y generación de código para ambos ambientes: el móvil y el de la nube. Por otra parte, rescatamos el hecho de incorporar mecanismos complementarios al desarrollo dirigido por modelos para mejorar la portabilidad como el enfoque de código abierto. Así mismo, según identificamos, existe una cantidad interesante de propuestas que no abarcan todos los aspectos de una aplicación y que podrían ser enriquecidas.

Cuadro 5: Limitantes con relación a la problemática de la portabilidad entre plataformas

Limitantes	Num	%	Observaciones
Consideración de modelado y generación de código para un solo ambiente (móvil o nube).	8/11	72.7	Una propuesta considera solo el ambiente de la nube. El resto considera únicamente el ambiente móvil.
Generación para menos de dos plataformas móviles.	2/11	18.2	Estas dos propuestas sólo generan para la plataforma Android.
No consideración del modelado de la capa de datos.	4/11	36.4	Sólo consideran el modelado (y generación) de la capa de presentación o de comportamiento.
No consideración o enfoque básico de la capa de presentación (móvil)	4/11	36.4	Tres propuestas no la consideran y la última la considera de una manera básica.
No consideración de un método de sincronización inteligente	2/3	75	Considerando solo las propuestas que contemplan un modelado en los dos ambientes: móvil y nube.

3. ¿Se identifican evidencias que muestren la adopción del modelado navegacional jerárquico?

Si bien hemos hallado estudios que contemplan el modelado de la navegación, no identificamos la consideración del modelado de navegación jerárquica y menos que la misma se constituya en el punto de partida del resto del diseño de una aplicación. Distinguimos entonces una tendencia de modelar la navegación como grafos sin tener en cuenta una jerarquía.

4. ¿Se identifica el establecimiento de una distinción entre los niveles de modelado: PIM y ASM?

Hemos encontrado estudios que definen una extensión de los modelos independientes de plataforma: la P1 y la P2. Sin embargo, no hemos identificado que dichos estudios consideren o reconozcan la diferenciación entre el PIM y el ASM. Por lo cual la aplicación de dichas extensiones al PIM lo modifican directamente y posiblemente disminuyen su reusabilidad y portabilidad.

5. ¿Se identifica una clara separación entre los detalles de la capa de presentación y comportamiento para el diseño de las aplicaciones?

La separación clara entre los detalles de la capa de presentación y el diseño de la capa de comportamiento, la identificamos solo en la P6 y la P10 que proponen lenguajes textuales. En propuestas como las P4, P8 y P11 existen una separaciones, aunque en el primero no se contempla la capa de presentación, en el segundo sólo se modela la capa de presentación, y en el tercero sólo se abarca la capa de comportamiento. En el resto de las propuestas no hemos identificado dicha separación, por lo cual se podría distinguir una tendencia a detalles de dichas capas en el modelado. En el cuadro 6 especificamos de forma cuantitativa la adopción de susodicha característica.

Cuadro 6: Separación de capas de presentación y comportamiento

Característica	Num	%	Observaciones
Separación de los detalles de la capa de presentación del diseño de la capa de comportamiento	5/11	45.5	Tres de las cinco propuestas son lenguajes textuales.

6. ¿Presentan las soluciones clasificadas algún tipo de evaluación?, en caso afirmativo, ¿son positivos los resultados de las evaluaciones de sus propuestas?

Todas las evaluaciones hechas (en ocho de los once trabajos) presentaron resultados alentadores. Un método de evaluación bastante utilizado es la comparación entre cantidad de líneas de código, o elementos de modelado y cantidad de líneas de código generada para cada plataforma específica. Las propuestas P3, P6 y P10 utilizaron dicho método y presentaron diferencias entre dichas cantidades en al menos un orden de magnitud. En la P1 se llevó a cabo una evaluación en base a la métrica días-hombre y se determinaron los siguientes beneficios: ahorro de esfuerzo 48%; y reducción de costo 21%. En la P5 se llevó a cabo un experimento sobre la experiencia de usuario en cuanto al uso de las funcionalidades CRUD generadas para distintas plataformas de la nube (GAE y Windows Azure); en dicho experimento los usuarios proveyeron evidencias de la equivalencia de las funcionalidades. En la

P4 se llevaron a cabo casos de pruebas sobre una aplicación específica para la evaluación de la equivalencia de sus implementaciones. Por un lado, la aplicación Android generada se comparó con una versión previamente desarrollada (sin la propuesta). Por el otro lado, se comparó la aplicación generada para Windows Phone con la aplicación generada para Android. En ambos casos se dieron evidencias positivas sobre la equivalencia de las aplicaciones. En la P8 se presenta una valoración cualitativa en dos fases sobre el diseño y desarrollo de dos aplicaciones, resaltando las propiedades de expresividad y legibilidad en la representación del lenguaje de modelado, eficiencia en la compilación/interpretación, curva de aprendizaje y previsibilidad, entre otras. Cabe destacar que algunas propuestas pueden ser enriquecidas para abarcar más aspectos: unos de modelado de la capa de datos y otros de la capa de presentación o de comportamiento. Por otra parte, la P9 menciona que no contemplan una generación de código eficiente. Una cuestión sujeta a mejora sobre la generación de código automático mencionada por la P7, fue la utilización de una cantidad redundante de variables y métodos, lo que hacía que el código fuera menos compacto en comparación a una implementación desarrollada en forma manual.

Cuadro 7: Evaluaciones de las propuestas

Aspecto	Num	%	Observaciones
Trabajos que presentaron una evaluación sobre sus propuestas	8/11	72.7	Uno de los métodos más utilizados es la comparación líneas de código.[39] Los tres trabajos restantes presentan sólo un ejemplo para mostrar la implementación de la propuesta.
De los trabajos que presentaron una evaluación, los que utilizaron un caso de estudio (en la industria)	4/8	50	Comparación contra el desarrollo manual tradicional en cuanto a costo y esfuerzo de desarrollo. [40]

Discusión de Resultados

En resumen, dentro del conjunto de los estudios identificados, considerando además las respuestas a las preguntas de investigación y los cuadros resúmenes resaltamos los siguientes puntos: i) para la comunicación remota, la implementacíon más utilizada (por las propuestas que consideran un modelado en la nube) es del estilo REST; ii) el tipo de modelado más utilizado es el gráfico; iii) solo cuatro de las once propuestas comprenden el modelado y la generación de código para la nube. Y de esas cuatro propuestas solamente dos presentan un esquema de modelado unificado y generación de código para ambos ambientes: móvil y de la nube; iv) solo una propuesta genera una implementación de sincronización inteligente; v) no identificamos una consideración de una diferenciación explícita conceptual entre PIM y ASM; vi) no identificamos la consideración de un modelado navegacional jerárquico; vii) identificamos una tendencia a mezclar detalles de la capa de presentación con ciertos aspectos de la capa de comportamiento; viii) la mayoría de las propuestas generan código nativo del lado móvil; ix) el lenguaje de programación más utilizado para desarrollar los módulos en la nube es Java; x) la mayoría de los estudios presentan evaluaciones sobre sus respectivas propuestas, con resultados positivos.

4 Problemática y Propuesta de Solución

Considerando lo expuesto en las secciones anteriores describimos aquí la problemática y la propuesta de solución de este proyecto final de carrera. Primero especificamos la problemática considerada en base a los resultados del estado del arte. Luego presentamos el marco global de la solución sugiriendo y justificando la extensión de MoWebA para el desarrollo de aplicaciones móviles. Finalmente determinamos el alcance (del desarrollo e implementación) del presente trabajo.

4.1 Problemática

En base a lo adelantado en la *Introducción* (sección 1) y lo descripto en el *Contexto de la Propuesta* (sección 2), la problemática a tratar consiste en la dificultad de la portabilidad a nivel de plataforma contemplada en: i) el ambiente móvil: debido principalmente a las diferencias de sistemas operativos, lenguajes de programación y librerías; ii) el ambiente de la nube: a causa de las diferencias a nivel de plataforma agudizadas por el inconveniente del *vendor lock-in*; iii) en las aplicaciones móviles con funcionalidades enriquecidas (con implementaciones en la nube): para una misma aplicación se implementan módulos tanto para el ambiente móvil como para la nube, lo cual se traduce igualmente en la necesidad de trabajar con distintas plataformas, lenguajes de programación y configuraciones.

Las consecuencias de esta problemática se traducen en un mayor tiempo, costo y esfuerzo para el desarrollo de distintas versiones (que se adapten a cada plataforma y a cada proveedor) de una misma aplicación. A través del mapeo sistemático de la literatura que hemos llevado a cabo para el *Estado del Arte* (sección 3), identificamos que existen varias propuestas que encaran dicha problemática con el enfoque dirigido por modelos, sin embargo, hallamos que las mismas no cubren ciertos aspectos que consideramos oportunos para el modelado de las aplicaciones

[39] Comparación entre líneas de código (en el caso de modelado textual) o elementos de modelado (en el caso de modelado gráfico) contra líneas de código de implementación para las distintas plataformas específicas.

[40] El primer trabajo (P1) utilizó la métrica días-hombre. El segundo (P6) realizó dos tipos de análisis: cualitativo y cuantitativo. El tercero (P3) presentó un resumen cuantitativo. A su vez, el cuarto (P8) elaboró un análisis de cualitativo sobre el diseño y generación de dos aplicaciones.

móviles. Haciendo referencia a la sección de *Discusiones de Resultados* del estado del arte, identificamos aspectos o cuestiones que se encuentran pobremente cubiertos como el modelado unificado y generación de código para el ambiente móvil y de la nube, la generación de un método de sincronización inteligente, la consideración de una diferenciación entre PIM y ASM, la consideración de un modelado de navegación jerárquica, una clara separación entre los detalles de la capa de presentación y ciertos aspectos de la capa de comportamiento. Así también hemos encontrado que prácticamente todas las implementaciones del lado de la nube son llevadas a cabo mediante Java, dejando de lado otras alternativas como Node.js, AngularJS, Django, que también se podrían utilizar.

Nuestro objetivo es entonces proponer un enfoque MDA para el desarrollo de ciertos aspectos del diseño de las aplicaciones móviles con funcionalidades enriquecidas que busque mejorar la portabilidad entre las distintas plataformas móviles y/o de la nube, teniendo en cuenta además las cuestiones poco cubiertas en el estado del arte que mencionamos previamente.

En la siguiente subsección, describimos nuestra propuesta de solución.

4.2 Marco Global de la Solución

El marco global de la propuesta de solución consiste en adoptar un enfoque MDA para el desarrollo de aplicaciones móviles con funcionalidades enriquecidas ya que una de las motivaciones principales de dicho paradigma es justamente el mejoramiento de la portabilidad, lo cual podría traducirse en beneficios como una adaptación adecuada a los cambios tecnológicos, un incremento de la productividad en el desarrollo de dichas aplicaciones, la reutilización de implementaciones entre otros beneficios citados en la sub-sección 2.5. Proponemos entonces a MoWebA como enfoque de desarrollo, basado en MDA, para aplicaciones móviles ya que comprende ciertos aspectos de modelado que consideramos importantes tanto para facilitar el modelado mismo como para mejorar la portabilidad, dichos aspectos son los siguientes:

- La Navegación, como perspectiva fundamental del diseño de aplicaciones, constituyéndose en:
  - El punto de partida para el modelado de las capas de comportamiento y presentación; con el objetivo de conseguir independencia de las estructuras de almacenamiento y de acceso de datos.
  - Una forma de simplificación del modelado mediante la inclusión de la propiedad jerárquica.

  Por consiguiente, y en el caso de las aplicaciones móviles, la navegación debe estar sujeta a la interacción del usuario y del contexto, por ello la misma se constituye en un aspecto fundamental para el éxito de dichas aplicaciones [29] [30]. MoWebA al considerar a la navegación como un aspecto fundamental podría constituirse en una opción adecuada para el ambiente móvil.
- Un claro aislamiento de detalles de la capa de presentación de las capas de comportamiento:
  - Este planteamiento se apoya en el principio de Separación de Intereses (*Separation of Concerns*). El objetivo es proveer un marco que permita al diseñador concentrarse en un aspecto de modelado a la vez, abogando por las ventajas que pueda comprender este esquema, considerando sobre todo que una de las capas que presenta mayor dificultad de portabilidad es la de presentación. Entonces, un aislamiento claro de esta capa de los detalles de las demás podría redituar en beneficios para la portabilidad del diseño.
- El establecimiento de un nivel de arquitectura en la definición del modelado denominado Modelos Específicos de Arquitectura (ASM):
  - El ASM se constituye en una extensión de los Modelos Independientes de Plataforma (PIM) donde se incorporan conceptos y propiedades de arquitecturas emergentes, como la móvil, manteniendo siempre la independencia de plataforma para de esa manera posibilitar la portabilidad de dicho nivel de modelado entre las distintas plataformas destino. Todo ello con el objetivo de proveer un mecanismo de adaptación a la dinámica evolución de las aplicaciones donde se comprende igualmente la reusabilidad del PIM y también a su vez la portabilidad del mismo entre las distintas arquitecturas.

Así mismo, se propone un diseño integrado de módulos locales y remotos, es decir se plantea un diseño unificado. Las razones son las siguientes:

- Los módulos del lado de la nube tienen una implementación distinta a los módulos del lado móvil. La pretensión detrás del diseño unificado es brindar un mismo esquema de modelado para ambos ambientes y de esa manera proveer al diseñador una abstracción adicional sobre dichas implementaciones [13].
- Por otra parte, existen varios estudios e investigaciones que consideran un afianzamiento creciente de la integración entre dispositivos móviles y la nube a razón de la superación de las limitaciones presentes. En ese sentido, existen diversas propuestas que buscan solucionar las limitaciones en el área, como por ejemplo el mejoramiento de una red poco confiable como lo es la inalámbrica, protocolos y procesos de comunicación más eficientes. Se considera entonces a dicha integración potencial como soporte al esquema de modelado unificado.

Como hemos identificado en el mapeo sistemático de la literatura, una opción conveniente sería la de adoptar una solución complementaria al enfoque dirigido por modelos para el problema de la portabilidad. En este sentido, considerando la nube y el problema del *vendor lock-in*, proponemos un esquema de desarrollo *open source*,[41] el cual brinda una mayor flexibilidad para tratar las mencionadas dificultades de dicho problema ya que facilita: la disponibilidad del código fuente y el seguimiento de estándares abiertos. Además, existen proyectos open source como Openshift[42] que trabajan en el desarrollo de métodos y técnicas para mejorar la portabilidad de aplicaciones y consecuentemente aliviar el problema del *vendor lock-in*. Así mismo, la implementación del lado de la nube en el estado del arte, se lleva a cabo mayoritariamente en Java, en ese sentido proponemos utilizar una plataforma alternativa a la misma. En cuanto al diseño de los módulos de la nube, nos basamos en el estilo de arquitectura REST, por lo cual la interfaz de comunicación entre el lado móvil y la nube corresponde a una implementación de interfaz de programación de aplicaciones genérica (*API* REST) que pueda colaborar con la portabilidad.

A partir de este marco global, a continuación especificamos el alcance (implementación de modelado, reglas de transformación) del presente proyecto final de carrera.

4.3 Alcance del Proyecto Final de Carrera

La implementación a llevar cabo comprende los siguientes aspectos de una aplicación móvil con funcionalidades enriquecidas: i) comunicación entre cliente (móvil) y servidor (nube); ii) modelado del backend en la nube.

La comunicación y el modelado en la nube se basan en el estilo de arquitectura REST. El backend en la nube comprende: i) el modelado y generación de la base de datos remota; ii) la generación de operaciones y consultas sobre la base de datos.

Tendríamos en cuenta tres diagramas del *PIM* de MoWebA: i) diagrama lógico; ii) diagrama de entidades; iii) diagrama de servicios.

La extensión de los diagramas comprende la adopción de conceptos relacionados a: i) La diferenciación de módulos locales (móvil) y remotos (nube); ii) El Estilo de Arquitectura REST.

A partir de los diagramas, el propósito es construir reglas de transformación que permitan generar código funcional para las plataformas destino. Las consideraciones relacionadas a dichas plataformas son las siguientes: i) generación de código nativo para al menos dos plataformas: Android, iOS; ii) generación de código y configuraciones necesarias para facilitar la portabilidad entre nubes públicas y privadas y entre proveedores de open cloud. Plataformas y frameworks destino: *Openshift - Red Hat, Node.js*.

Para la validación de la propuesta consideramos dos alternativas, por una parte una experimentación que incluya un grupo de desarrolladores o bien una ilustración, llevada a cabo por nosotros. Sobre la alternativa elegida consideraríamos, como una posibilidad, ciertas características y sub-características del estándar ISO 9126[43] o de la ISO/IEC 25000 SQuaRE[44], a partir de las cuáles determinaríamos las métricas pertinentes para nuestra evaluación. Además en la problemática ya hemos resaltado cuestiones como el tiempo, costo y esfuerzo las cuales podemos adoptarlas igualmente como métricas.

5 Avances

Describimos brevemente a continuación, las actividades que hemos llevado a cabo, y las que aún quedan pendientes para culminar el proyecto final de carrera.

Actividades hechas

La primer actividad que hemos realizado, fue un estudio preliminar sobre el área de las aplicaciones móviles con el objetivo de hallar un enfoque similar a RIA. A partir de ello, identificamos a las aplicaciones móviles enriquecidas, para luego centrarnos en una de sus características: las funcionalidades enriquecidas. Luego, encontramos que el mejoramiento de la portabilidad es una necesidad crítica, tanto para el ambiente móvil como de la nube. En base a ello, consideramos al enfoque desarrollo dirigido por modelos como solución a dicho problema.

Teniendo en cuenta toda la información recabada, elaboramos la propuesta del proyecto final de carrera y posteriormente, llevamos a cabo el estado del arte a través de un estudio sistemático de la literatura (sobre el desarrollo de aplicaciones móviles enriquecidas bajo el enfoque dirigido por modelos). Hemos resumido dicho mapeo sistemático en un artículo, el cual, a su vez, hemos postulado a una conferencia para su publicación.

Por otra parte, iniciamos un estudio del estilo de arquitectura REST y de las plataformas destino a considerar. Además hemos elaborado una versin inicial del metamodelo.

Actividades pendientes

[42] Openshift, enlace: `https://goo.gl/UPRU9E`
[43] ISO 9126, enlace: `http://goo.gl/Cin7ku`
[44] ISO/IEC 25000 SQuaRE, enlace: `http://goo.gl/uaFa00`

Nos resta completar y pulir los metamodelos y perfiles, además de verificar de la extensión de MoWebA, y estudiar las herramientas de elaboración de reglas transformación. Igualmente resta la implementación y revisión de las reglas.

Un punto importante a definir y llevar a cabo es la validación del producto generado, con el objetivo de evaluar los resultados que arroja la propuesta. A su vez, nos queda continuar y finalizar el libro de proyecto final de carrera.

Referencias

[1] Abolfazli, S., Sanaei, Z., Gani, A., Xia, F., Yang, L.T.: Rich mobile applications: genesis, taxonomy, and open issues. Journal of Network and Computer Applications. 40, 345-362 (2014)

[2] March, V., Gu, Y., Leonardi, E., Goh, G., Kirchberg, M., Lee, B.S.: Mcloud: Towards a new paradigm of rich mobile applications. Procedia CS. 5, 618-624 (2011)

[3] Sahu, D., Sharma, S., Dubey, V., Tripathi, A.: Cloud computing in mobile applications. International Journal of Scientific and Research Publications. 2(8), 1-9 (2012)

[4] Sanaei, Z., Abolfazli, S., Member, Gani, A., Buyyag, R.: Heterogeneity in mobile cloud computing: taxonomy and open challenges. Communications Surveys & Tutorials, IEEE. 16(1), 369-392 (2014)

[5] Vaupel, S., Taentzer, G., Harries, J.P., Stroh, R., Gerlach, R., Guckert, M.: Model-driven development of mobile applications allowing role-driven variants. In: Model-Driven Engineering Languages and Systems. pp. 1-17. Springer (2014)

[6] Brambilla, M., Mauri, A., Umuhoza, E.: Extending the interaction flow modeling language (IFML) for model driven development of mobile applications front end. In: Mobile Web Information Systems - 11th International Conference, MobiWIS 2014, Barcelona, Spain, August 27-29. pp. 176-191. Springer International Publishing (2014)

[7] Gupta, P., Gupta, S.: Mobile cloud computing: The future of cloud. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering. 1(3), 134-145 (2012)

[8] da Silva, E.A.N., Fortes, R.P., Lucrédio, D.: A model-driven approach for promoting cloud paas portability. In: CASCON. pp. 92-105. (2013)

[9] Pons, C., Giandini, R., Pérez, G.: Desarrollo de Software Dirigido por Modelos. Editorial de la Universidad Nacional de La Plata (EDULP)/McGraw-Hill Educación, La Plata - Buenos Aires (2010)

[10] Brambilla, M., Cabot, J., Wimmer, M.: Model-Driven Software Engineering in Practice. Morgan & Claypool (2012)

[11] Acerbis, R., Bongio, A., Brambilla, M., Butti, S.: Modeldriven development based on omg's ifml with webratio web and mobile platform. In: Engineering the Web in the Big Data Era. pp. 605-608. Springer (2015)

[12] Heitkötter, H., Majchrzak, T.A., Kuchen, H.: Cross-platform model-driven development of mobile applications with md^2. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing, ACM. pp. 526-533. (2013)

[13] Ranabahu, A.H., Maximilien, E.M., Sheth, A.P., Thirunarayan, K.: A domain specific language for enterprise grade cloud-mobile hybrid applications. In: Proceedings of the compilation of the co-located workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11, ACM. pp. 77-84. (2011)

[14] González, M., Cernuzzi, L., Pastor, O.: A navigational role-centric model oriented web approach - MoWebA. International Journal of Web Engineering and Technology. (2016) (Obs: Publicación pendiente)

[15] Fernando, N., Loke, S.W., Rahayu, W.: Mobile cloud computing: A survey. Future Generation Computer Systems. 29(1), 84-106 (2013)

[16] Shiraz, M., Gani, A., Khokhar, R.H., Buyya, R.: A review on distributed application processing frameworks in smart mobile devices for mobile cloud computing. Communications Surveys & Tutorials, IEEE. 15(3), 1294-1313 (2013)

[17] ur Rehman Khan, A., Othman, M., Madani, S.A., Khan, S.U.: A survey of mobile cloud computing application models. Communications Surveys & Tutorials, IEEE. 16(1), 393-413 (2014)

[18] Wang, Y., Chen, R., Wang, D.C.: A survey of mobile cloud computing applications: Perspectives and challenges. Wireless Personal Communications. 80(4), 1607-1623 (2015)

[19] Mell, P., Grance, T.: The nist definition of cloud computing. (2011)

[20] Dinh, H.T., Lee, C., Niyato, D., Wang, P.: A survey of mobile cloud computing: architecture, applications, and approaches. Wireless communications and mobile computing. 13(18), 1587-1611 (2013)

[21] Fielding, R.T., Taylor, R.N.: Principled design of the modern web architecture. ACM Transactions on Internet Technology (TOIT). 2(2), 115-150 (2002)

[22] Richardson, L., Amundsen, M., Ruby, S.: RESTful Web APIs. O'Reilly Media, Inc. (2013)

[23] Riva, C., Laitkorpi, M.: Designing web-based mobile services with rest. In: Service-Oriented Computing-ICSOC 2007 Workshops. pp. 439-450. Springer (2009)

[24] Mohamed, K., Wijesekera, D.: Performance analysis of web services on mobile devices. Procedia Computer Science. 10, 744-751 (2012)

[25] Upadhyaya, B., Zou, Y., Xiao, H., Ng, J., Lau, A.: Migration of soap-based services to restful services. In: Web Systems Evolution (WSE), 2011 13th IEEE International Symposium on. pp. 105-114. IEEE (2011)

[26] Kumari, S., Rath, S.K.: Performance comparison of soap and rest based web services for enterprise application integration. In: Advances in Computing, Communications and Informatics (ICACCI), 2015 International Conference on. pp. 1656-1660. IEEE (2015)

[27] Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Technical report, Keele University and University of Durham (2007) Version 2.3.

[28] Genero, M., Cruz-Lemus, J., Piattini, M.: Métodos de Investigación en Ingeniería del Software. RA-MA, Madrid (2014)

[29] Neil, T.: Mobile Design Pattern Gallery: UI Patterns for Smartphone Apps. 2nd edn. O'Reilly Media, Inc. (2014)

[30] Fling, B.: Mobile Design and Development: Practical concepts and techniques for creating mobile sites and web apps. O'Reilly Media, Inc. (2009)