UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

TIA
Tecnología, Investigación y Academia

Market Cart App: aplicación móvil para la gestión de compra de víveres en línea

Market Cart App: Mobile Application for the Management of Food Purchasing Online

Iván González[1]

Para citar este artículo: González, I. (2018). Market Cart App: aplicación móvil para la gestión de compra de víveres en línea. *TIA*, 6(1), pp. 3-17.

Artículo de investigación

Fecha de recepción:
24-05-2015
Fecha de aceptación:
14-07-2017

ISSN: 2344-8288
Vol. 6 No. 1
Enero - junio 2018
Bogotá-Colombia

Resumen

Este artículo describe claramente el desarrollo del prototipo funcional de una aplicación móvil enfocada en la gestión de compra de víveres, lo anterior para el uso de las personas en el día a día de sus vidas. Esta aplicación se enfocada en facilitar la vida de las personas haciendo que la compra de víveres se realice desde la comodidad de sus casas. En este artículo se describe el proceso de desarrollo de la aplicación, las fases de desarrollo y las interfaces del prototipo funcional en Android.

Palabras clave: aplicación Android, compra, Kruchten, víveres, XP.

Abstract

This article describes clearly the development of the functional prototype of a mobile application focused on managing the purchase of supplies for use by the people in their everyday lifes. This application is focused on making life easier for people doing that their grocery shopping can be done from the comfort of their homes. This article describes the process of the application's development, the stages of development and the interfaces of the functional prototype in Android.

Keywords: Android application, purchase, Kruchten, groceries, XP.

[1] Ingeniero de Sistemas. Especialización en Ingeniería de Software, Universidad Distrital Francisco José de Caldas. Support Engineer, Bizagi LATAM. Correo electrónico: ianlee9204@hotmail.com

INTRODUCCIÓN

Con el resplandor de la era tecnológica y el *boom* de los dispositivos inteligentes, las personas ahora pueden realizar casi cualquier tarea que antes requería tiempo, estrés y dificultad en sus dispositivos; un ejemplo claro de esto son las diferentes acciones que debían realizarse en los bancos, el hecho de tener que disponerse a realizar una transacción en un cajero electrónico era dispendioso, difícil y estresante, sin embargo, con las aplicaciones móviles para hacer transacciones como la aplicación de BBVA o Bancolombia ahora es posible realizar varias transacciones desde el teléfono inteligente que facilitan el día a día de cualquier persona.

Otras de las tareas que dificultan el quehacer diario es la compra de víveres para el hogar. Esta tarea no solo conlleva el hecho de tener que trasladarse a un supermercado sino también el hecho de demorarse una gran cantidad de tiempo haciendo filas, estresándose por la cantidad de personas en el supermercado y el mal servicio de las cajas registradoras a la hora de realizar el pago por los víveres.

IMPORTANCIA DE LAS APLICACIONES MÓVILES EN LA ACTUALIDAD

Hoy en día se vive en un mundo donde la tecnología ha facilitado la vida del ser humano, al punto de permitirle delegar tareas o actividades indeseadas, pero totalmente necesarias, a una aplicación. Adicional a esto, el acceso a la tecnología móvil facilita las interacciones, comunicaciones y transacciones diarias.

El día a día está hecho de un rango amplio de prácticas sociales que recaen en la rutina de las experiencias de las personas, estas incluyen trabajo, familia, sociabilidad, consumo, salud, servicios sociales, seguridad, entretenimiento [2]. Las experiencias pueden generar gusto o disgusto dependiendo de las actividades en las que se basan, si estas actividades dificultan la vida de las personas y generan experiencias negativas, ¿qué se puede hacer para generar un cambio significativo en la experiencia?

Así, la pregunta que se plantea es, ¿cómo aprovechar al máximo estas tecnologías para remover las actividades indeseadas por las personas en su diario vivir?

Importancia de las aplicaciones móviles en el diario vivir de las personas

Actualmente se pueden observar ejemplos claros que proponen soluciones interesantes a ciertas actividades tediosas de realizar como lo son las aplicaciones bancarias. ¿Quién hubiese pensado unos años atrás que se podrían realizar transacciones de dinero desde un celular?, ahora esto es posible y demuestra cómo un simple celular puede ser un dispositivo potente y práctico para actividades del día a día.

Pero esto no es suficiente. Con el avance tecnológico exponencial que se experimenta actualmente, cualquier cosa que se hubiese soñado años o décadas atrás se está convirtiendo cada vez más en una realidad y menos en un sueño.

Otro ejemplo de lo anterior son las aplicaciones de transporte. Con el auge de las aplicaciones de Taxis como Tappsi o SmartTaxi se cambió totalmente la forma en que las personas usaban este servicio. Anteriormente era necesario realizar una llamada que podía demorar quince, veinte o hasta 30 minutos en ser respondida por una persona, además, se solicitaba el servicio que podía demorar dependiendo de la cantidad de taxis libres en ese momento y de la distancia a la que se encontraba el taxi seleccionado del lugar en que se encontraba la persona.

Importancia de las aplicaciones móviles en las organizaciones

Es importante recalcar la importancia de las aplicaciones móviles en las personas, pero también es importante recalcar los beneficios del desarrollo y el uso de las aplicaciones móviles en las organizaciones.

Los dispositivos móviles están permitiendo a las organizaciones y a las personas manejar sus negocios y vidas de manera más eficaz y efectiva. Las aplicaciones móviles pueden ser usadas para soportar el comercio electrónico con clientes y proveedores y conducir el comercio electrónico dentro y fuera de los límites de las organizaciones. A pesar de estos beneficios, las organizaciones y las personas son faltas de entendimiento en el valor de las aplicaciones móviles [3]. El valor es definido aquí como los principios para evaluar las consecuencias de acción, falta de acción, o decisión [4]. El valor de la proposición de las aplicaciones móviles puede ser definido como el valor neto de los beneficios y costos asociados con la adopción y adaptación de las aplicaciones móviles [5].

Con estas aplicaciones a las personas se les facilitó el uso del servicio en su diario vivir. Se eliminó al intermediario que comunicaba al taxista con el cliente y se creó una aplicación donde el mismo taxista se comunicaba directamente con el cliente a través de una aplicación móvil. Por razón de esta aplicación cambió el modo en que solicitaba un servicio de taxi y esto cambió la vida de las personas. ¿Cuál es el fin de la tecnología sino la de facilitar la vida del ser humano?

Teniendo en cuenta lo anterior, se propone una aplicación para facilitar la vida de las personas, una aplicación para realizar las compras de víveres que pueden llegar a ser difíciles, estresantes, y agotadoras para las personas diariamente, esta aplicación es Market Cart App.

MARCO TEÓRICO

Técnica de requerimientos de casos de uso

El diagrama de casos de uso representa la forma en que un cliente (actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en que los elementos interactúan (operaciones o casos de uso).

Un diagrama de casos de uso consta de los siguientes elementos:

- Actor.
- Casos de uso.
- Relaciones de uso, herencia y comunicación.
- Elementos.

Actor

Es un rol que un usuario juega con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema.

Como ejemplo a la definición anterior, se tiene el caso de un sistema de ventas en que el rol de vendedor con respecto al sistema puede ser realizado por un vendedor o bien por el jefe de local.

Caso de uso

Es una operación específica que se realiza tras una orden de algún agente. Relación que indica la invocación desde un actor o caso de uso a otra operación (caso de uso). Dicha relación se denota con una flecha simple.

- Dependencia o instanciación.

Relación en la cual una clase depende de otra, es decir, se instancia (se crea). Dicha relación se denota con una flecha punteada.

- Generalización.

Esta relación cumple una doble función dependiendo de su estereotipo, que puede ser de uso ("*uses*") o de herencia ("*extends*"). Se recomienda utilizar *extends* cuando un caso de uso es similar a otro (características). Se recomienda utilizar *uses* cuando se tiene un conjunto de características que son similares en más de un caso de uso y no se desea mantener copiada la descripción de la característica [6].

EXTREME PROGRAMMING

La metodología XP define cuatro variables para cualquier proyecto de *software*: costo, tiempo, calidad y alcance. Además, se especifica que de estas cuatro variables sólo tres podrán ser fijadas arbitrariamente por actores externos al grupo de desarrolladores (clientes y jefes de proyecto).

El ciclo de vida de un proyecto XP incluye, al igual que las otras metodologías, entender lo que el cliente necesita, estimar el esfuerzo, crear la solución y entregar el producto final al cliente; sin embargo, XP propone un ciclo de vida dinámico donde se admite expresamente que, en muchos casos, los clientes no son capaces de especificar sus requerimientos al comienzo de un proyecto.

Por esto, se trata de realizar ciclos de desarrollo cortos (llamados iteraciones), con entregables funcionales al finalizar cada ciclo. En cada iteración se realiza un ciclo completo de análisis, diseño, desarrollo y pruebas, pero utilizando un conjunto de reglas y prácticas que caracterizan a XP (que serán detalladas más adelante). Típicamente, un proyecto con XP lleva diez a quince ciclos o iteraciones. La Figura 1 esquematiza los ciclos de desarrollo en cascada e iterativos tradicionales (por ejemplo, incremental o espiral), comparados con el de XP.

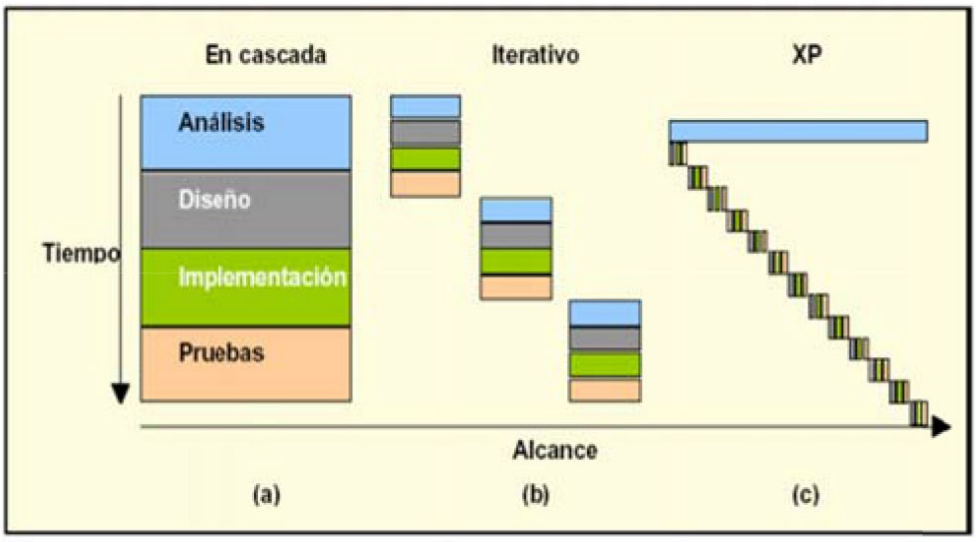

Figura 1. Reglas y prácticas en *eXtreme Programming*

Fuente: [7].

Fase de exploración

Es la fase en la que se define el alcance general del proyecto. En esta fase, el cliente define lo que necesita mediante la redacción de sencillas historias de usuarios. Los programadores estiman los tiempos de desarrollo con base en esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias, ya que estarán basadas en datos de muy alto nivel y podrían variar cuando se analicen más en detalle en cada iteración.

Esta fase dura típicamente un par de semanas, el resultado es una visión general del sistema y un plazo total estimado.

Fase de planificación

La planificación es una fase corta en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a estas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un plan de entregas o *release plan*, como se detallará en la sección reglas y prácticas.

Fase de iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración.

Como las historias de usuario no tienen suficiente detalle para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

Fase de puesta en producción

Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta no tener la funcionalidad completa. En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste (*fine tuning*) [7].

ARQUITECTURA SOA

Es un paradigma para organizar y utilizar capacidades distribuidas que puedan estar bajo el control de diferentes dominios de propiedad. La idea que subyace a esta arquitectura es que siempre conviene ordenar la forma en la que se comunican las distintas partes de un sistema; para conseguir este objetivo, se define una entidad o *middleware* con el que todos los sistemas intercambian información mediante conectores.

Este *middleware* actuará como una capa de adaptación cuya finalidad es aislar la forma de comunicarse o cambiar información con cada sistema particular. De esta forma, si un sistema *A* necesitara datos de otro sistema *B*, *A* no tendría que tener información sobre cómo es la base de datos de *B*, sino que *A* se lo pediría al *middleware*, este le solicitaría la información a *B* que es el que conoce la estructura de su base de datos. *B* accedería a la base de datos y devolvería el valor al *middleware* para que finalmente le llegara la información a *A*.

Si en algún momento se deseara sustituir *B* por otro sistema *B*, el cambio sería transparente para *A*, ya que en ningún momento se ha realizado en *A* un desarrollo dependiente de la estructura de *B*, sino únicamente dependiente de los datos que *B* contiene [8].

En la Figura 2 se puede apreciar lo mencionado anteriormente.

ARQUITECTURA

La arquitectura del *software* se trata de abstracciones, de descomposición y composición, de estilos y estética. También tiene relación con el diseño y la implementación de la estructura de alto nivel del *software*. Los diseñadores construyen la arquitectura usando varios elementos arquitectónicos elegidos apropiadamente, estos elementos satisfacen la mayor parte de los requisitos de funcionalidad y performance del sistema, así como también otros requisitos no funcionales como confiabilidad, escalabilidad, portabilidad y disponibilidad del sistema, Figura 3.

Vista lógica

La arquitectura lógica apoya principalmente los requisitos funcionales, lo que el sistema debe brindar en términos de servicios a sus usuarios. El sistema se descompone en una serie de abstracciones clave, tomadas (principalmente) del dominio del problema en la forma de objetos o clases de objetos; aquí se aplican los principios de abstracción, encapsulamiento y herencia. Esta descomposición no solo se hace para potenciar el análisis funcional, sino que también sirve para identificar mecanismos y elementos de diseño comunes a diversas partes del sistema.

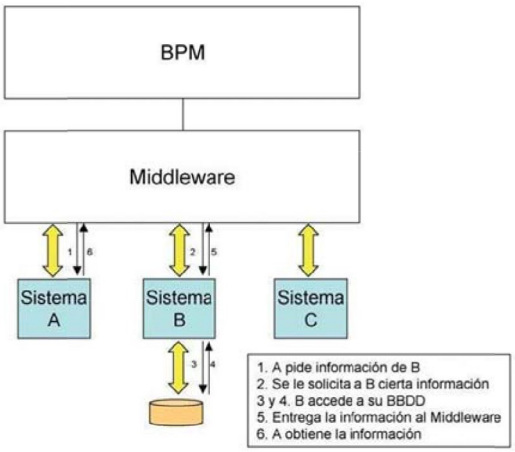

Figura 2. SOA (arquitectura orientada al servicio)
Fuente: Julio Alba. Consultor/director de proyectos de SATEC.

Figura 3. Planos arquitectónicos. Vistas de la arquitectura del *software*
Fuente: [9].

Se usa el enfoque de *Booch/Rational* para representar la arquitectura lógica, mediante diagramas de clases [9].

Vista de procesos

La arquitectura de procesos toma en cuenta algunos requisitos no funcionales tales como la performance y la disponibilidad. Se enfoca en asuntos de concurrencia y distribución, integridad del sistema, de tolerancia a fallas. La vista de procesos también específica en cuál hilo de control se ejecuta efectivamente una operación de una clase identificada en la vista lógica. El flujo de mensajes y la carga de procesos pueden estimarse con base en el diagrama de procesos [10].

Vista de desarrollo

La vista de desarrollo se centra en la organización real de los módulos de *software* en el ambiente de desarrollo del *software*. El *software* se empaqueta en partes pequeñas bibliotecas de programas o subsistemas que pueden ser desarrollados por uno o un grupo pequeño de desarrolladores. Los subsistemas se organizan en una jerarquía de capas, cada una de las cuales brinda una interfaz estrecha y bien definida hacia las capas superiores.

La vista de desarrollo de un sistema se representa en diagramas de módulos o subsistemas que muestran las relaciones exporta e importa. La arquitectura de desarrollo solo puede describirse completamente cuando todos los elementos del *software* han sido identificados; sin embargo, es posible listar las reglas que rigen la arquitectura de desarrollo partición, agrupamiento, visibilidad antes de conocer todos los elementos.

Se puede dibujar la arquitectura de desarrollo en *Rational Rose* en cuanto a módulos y subsistemas, en ingeniería hacia adelante y reversa a partir de código fuente Ada y C++ [9].

Vista física

La arquitectura física toma en cuenta primeramente los requisitos no funcionales del sistema tales como la disponibilidad, confiabilidad (tolerancia a fallas), performance (*throughput*) y escalabilidad. El *software* ejecuta sobre una red de computadores o nodos de procesamiento (o tan solo nodos). Los variados elementos identificados redes, procesos, tareas y objetos requieren ser mapeados sobre los variados nodos [9].

Vista de escenarios

Los elementos de las cuatro vistas trabajan conjuntamente en forma natural mediante el uso de un conjunto pequeño de escenarios relevantes —instancias de casos de uso más generales— para los cuales son descritos los scripts correspondientes (secuencias de interacciones entre objetos y entre procesos) tal como lo describen Rubin y Goldberg[1]. Los escenarios son de alguna manera una abstracción de los requisitos más importantes.

Su diseño se expresa mediante el uso de diagramas de escenarios y diagramas de interacción de objetos [3].

PROCESO DE DESARROLLO

En esta aplicación se va a trabajar con la metodología ágil XP (*eXtreme Programming*) ya que se requiere que la aplicación se implemente en poco tiempo.

Los métodos ágiles suelen ser muy adecuados para el desarrollo de aplicaciones móviles por las razones mencionadas a continuación.

- Alta volatilidad del entorno: con cambios en entornos de desarrollo, nuevos terminales y nuevas tecnologías a un ritmo mucho más elevado que en otros entornos de desarrollo.

- Equipos de desarrollo pequeños: dado que los desarrollos móviles suelen ser proyectos relativamente pequeños, los equipos no son con frecuencia muy grandes. Generalmente son llevados a cabo por desarrolladores individuales o por PYME.
- *Software* no crítico: no suelen ser aplicaciones de alto nivel de criticidad, dado que suelen ser aplicaciones para entretenimiento o gestión empresarial no crítica.
- Ciclos de desarrollo cortos: dada la evolución constante de la industria, se requieren ciclos de vida realmente cortos para poder dar salida a las aplicaciones a tiempo [11].

El proyecto de la aplicación contiene la siguiente estructura:

- Exploración.
- Planificación de entregables.
- Plan de iteraciones.
- Desarrollo.
- Mantenimiento y pruebas.

Es importante describir que el proceso de desarrollo es incremental, lo que facilita el desarrollo través de la planificación de iteraciones cortas que aseguran el desarrollo progresivo del prototipo. En este proyecto se va a manejar para el análisis de requerimientos, la técnica de casos de uso y los diagramas de casos de uso de UML. Para el envío de la lista de víveres al supermercado se debe realizar un servicio web con una arquitectura SOA que facilite el manejo de la información desde la aplicación móvil.

VISTAS DE ARQUITECTURA —4+1‖ KRUCHTEN

Para facilitar el desarrollo de la aplicación se describe la arquitectura de la aplicación a través de las vistas —4+1‖ de Kruchten.

En primer lugar, se describe la vista de escenarios a través de los diagramas de casos de uso que especifican las funcionalidades principales de los requerimientos analizados.

Vista de escenarios

En la Figura 4 se presenta el diagrama de casos de uso de Market Cart App.

Se plantea inicialmente el caso de uso "comprar víveres" que detalla la funcionalidad principal de la aplicación móvil (Figura 5).

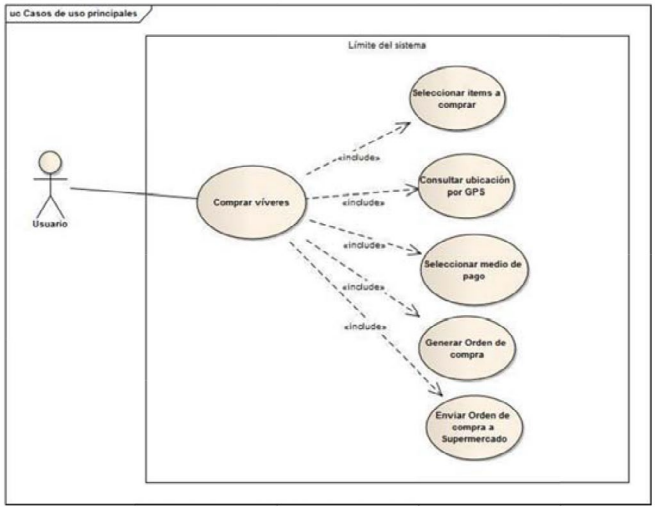

Figura 4. Diagrama de casos de uso

Fuente: elaboración propia.

Figura 5. Caso de uso comprar víveres

Fuente: elaboración propia.

A continuación, se describen los siguientes casos de uso que proporcionan una funcionalidad específica que constituye la funcionalidad del caso de uso de comprar víveres (Figura 6, Figura. 7, Figura 8, Figura 9, Figura 10).

Figura 6. Caso de uso seleccionar ítems a comprar.

Fuente: elaboración propia.

Figura 7. Caso de uso consultar ubicación por GPS

Fuente: elaboración propia.

Figura 8. Caso de uso seleccionar medio de pago

Fuente: elaboración propia.

Figura 9. Caso de uso generar orden de compra

Fuente: elaboración propia.

Figura 10. Caso de uso enviar orden de compra al supermercado

Fuente: elaboración propia.

PROTOTIPO FUNCIONAL

A continuación, se describe el prototipo funcional junto con unas imágenes que detallan su funcionamiento.

Inicialmente se despliega una interfaz gráfica con el nombre de la aplicación y un botón que dice "comprar víveres" como se muestra en la Figura 11.

Figura 11. Pantalla de inicio

Fuente: elaboración propia.

Al oprimir el botón anterior se despliega el siguiente menú mostrando las diferentes funcionalidades del aplicativo (Figura 12).

Por otro lado, es necesario mencionar que cada botón tiene una funcionalidad diferente. En el primer caso "mi lista de compras" al acceder se observa la pantalla presentada en la Figura 13.

En la sección "buscar ítem" se escribe el nombre del producto necesitado. Cuando se encuentra el ítem se oprime el botón agregar indicado con el signo +.

En la descripción del producto se puede observar el nombre, la descripción o tipo del producto, la cantidad se escribe y el precio

Figura 12. Menú de opciones

Fuente: elaboración propia.

Figura 13. Pantalla "mi lista de compras"

Fuente: elaboración propia.

se calcula automáticamente con base en la cantidad. En el campo de texto "total" se calcula automáticamente el precio total de la orden con base en los precios de los productos seleccionados. Si se quiere eliminar un producto de la lista se oprime el botón - y el producto se elimina.

Cuando se oprime el botón "seleccionar ítem" se despliega la pantalla mostrada en la Figura 14.

En la pantalla se observa un campo de filtro llamado "todos", el cual permite modificar los productos que se visualizan. Si se escribe una categoría específica como "pescados" o "gaseosas" se muestran los productos relacionados con esa categoría.

Adicional a eso, si se quiere ver específicamente un producto se selecciona y se despliega en la pantalla el producto (Figura 15).

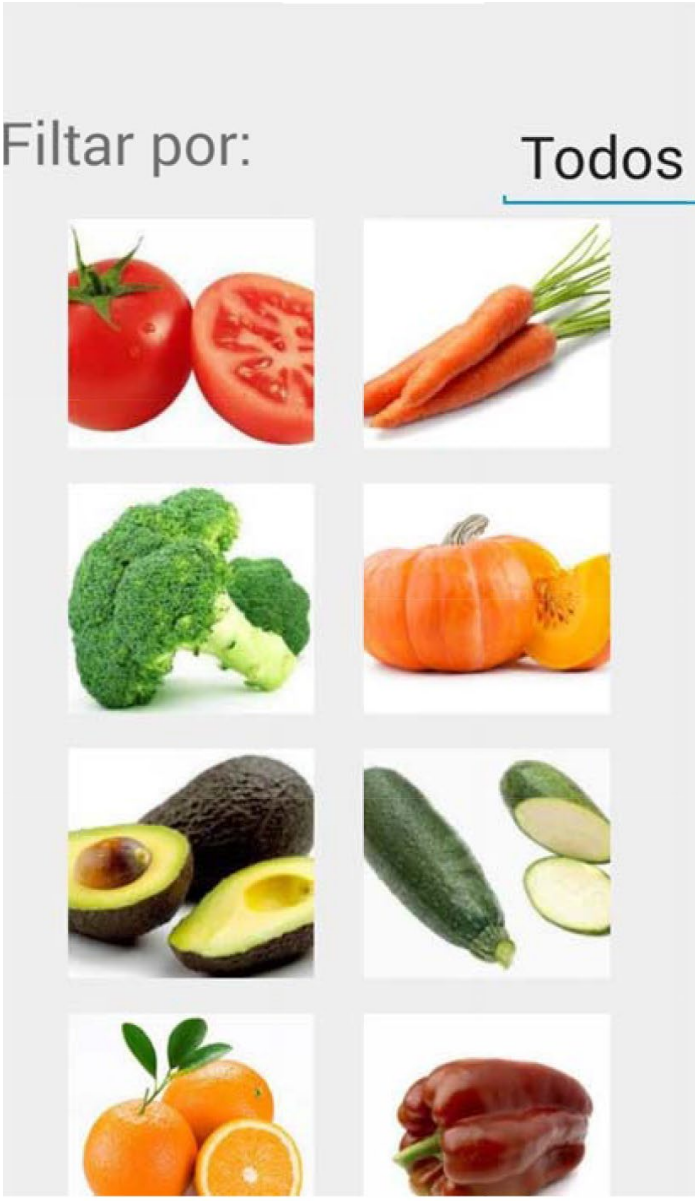

Figura 14. Pantalla "seleccionar ítem"

Fuente: elaboración propia.

Figura 15. Pantalla "detalle ítem"

Fuente: elaboración propia.

Cuando se oprime el botón "mi ubicación" se despliega una pantalla que permite al usuario detallar su ubicación de domicilio con la integración de Google Maps (Figura 16).

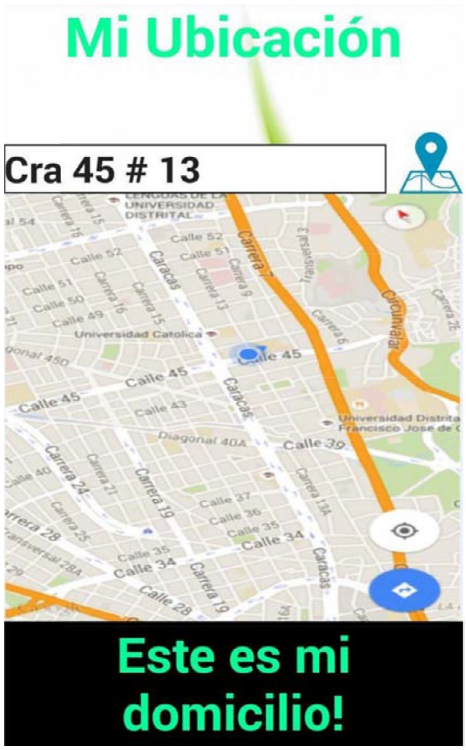

Figura 16. Pantalla "mi ubicación"

Fuente: elaboración propia.

El usuario, al acceder a esta pantalla, tiene dos opciones. Inicialmente puede escribir su dirección en el campo superior y oprimir el botón "este es mi domicilio"; al realizar eso, la aplicación determina su dirección de domicilio para enviar los productos a esa dirección.

La segunda opción que tiene el usuario es oprimir el botón de "ubicación" al lado del campo de dirección; este botón automáticamente detecta la dirección del usuario con la integración de Google Maps. Después de esto el usuario debe oprimir el botón "este es mi domicilio" para confirmar y enviar la dirección de domicilio.

Se pasa entonces a los medios de pago, en esta parte el usuario elige la opción de pago a realizar: efectivo o tarjeta de crédito/débito (Figura 17).

Si se decide por la opción de efectivo el usuario realiza el pago cuando el domicilio llega a su hogar. De lo contrario con la integración de un servicio externo de pagos online se realiza el pago en línea.

CONCLUSIONES

Las aplicaciones móviles brindan oportunidades inmensas para facilitar el día a día de las personas, permitiéndoles realizar actividades tediosas a través de aplicaciones desde su dispositivo móvil para dedicar a ellas el mínimo tiempo posible con el fin de generar experiencias positivas en sus vidas.

En el desarrollo del prototipo funcional de la aplicación móvil Market Cart App se pudo observar la practicidad y facilidad de implementar un proceso de desarrollo basado en metodologías ágiles XP. Este proceso de desarrollo permitió la implementación del prototipo funcional en un mes con su respectivo ciclo de vida del *software*.

Con el desarrollo de la arquitectura del aplicativo móvil basado en las vistas —4+1‖ de Kruchten se permitió plantear una arquitectura completa que pudiera definir y llevar como base para el desarrollo del prototipo funcional.

Figura 17. Pantalla "medio de pago"

Fuente: elaboración propia.

La arquitectura SOA permitió tener los conocimientos necesarios para iniciar la segunda etapa en desarrollo de la aplicación móvil Market Cart App, ya que facilitó la integración de los servicios web a la aplicación.

REFERENCIAS

[1] Rubin, K. S. & Goldberg, A. (1992). Object behavior analysis. *Communications of the ACM, 35*(9), 48-62.

[2] Castells, M., Fernández, M., Linchuan, J. & Sey, A. (2006). *Mobile Communication and Society: A Global Perspective*. London. The MIT Press.

[3] Nah, F., Siau, K. & Sheng, H. (2005). *The Value of Mobile Applications: A Utility Company Study*. Nebraska: University of Nebraska-Lincoln.

[4] Keeney, R. (1992). *Value-Focused Thinking*. Cambridge, Massachusetts: Harvard University Press.

[5] Marsden, J., Tung, Y. & Keeney, R. (1999) The Value of Internet Commerce to the Customer. *Management Science*, *45*(4), 533-542.

[6] Casos de Uso. (s.f.). Universidad de Chile. Recuperado de http://users.dcc.uchile.cl/~psalinas/uml/casosuso.html

[7] Joskowicz, J. (2008) Reglas y prácticas en eXtreme Programming. Recuperado de https://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf

[8] Alba, J. (2008). ¿Qué es SOA (Arquitectura Orientada al Servicio)? *Bit*, *167*, 52-53.

[9] Kruchten, P. (1995). Planos arquitectónicos: el modelo de "4+1" vistas de la arquitectura del software. *IEEE Software*, *12*(6).

[10] Witt, B., Baker, T. & Merrit, E. (1994). *Software Architecture and Design Principles, Models, and Methods*. Nueva York: Van Nostrand Reinhold.

[11] Ramírez, R. (s.f.). *Métodos para el desarrollo de aplicaciones móviles*. Barcelona: Universidad Abierta de Cataluña.