

Ayudar a los niños a comer bien mediante tecnologías de software móvil

Violetta Vylegzhanina, Douglas C. Schmidt, Pamela Hull, Janice S. Emerson,
Meghan E. Quirk y Shelagh Mulvaney

Universidad de Vanderbilt, Nashville Tennessee, Estados Unidos
{violetta.vylegzhanina, douglas.c.schmidt, pam.hull, shelagh.mulvaney}@vanderbilt.edu
Universidad Estatal de Tennessee, Nashville, Tennessee, EE.UU.
{jemerson, mquirk}@tnstate.edu

Resumen

Este artículo describe una aplicación móvil para Android que hemos desarrollado para simplificar la experiencia de compra de los participantes en el Programa Especialra Mujeres, Bebés y Niños (WIC),

que proporciona a las familias de bajos ingresos vales para comprar alimentos nutritivos y apropiados para la etapa de la vida. Nuestra aplicación ayuda a aliviar el uso tedioso y propenso a errores de los vales WIC en papel, permitiendo a los participantes escanear los alimentos en la tienda e identificar automáticamente si un artículo (incluyendo su tamaño y cantidad) está autorizado para el participante WIC inscrito. Además de servir como herramienta de compra, la aplicación también proporciona una plataforma para la educación nutricional a través de notificaciones de consejos saludables y una galería de recetas de aperitivos fáciles de preparar que se adaptan a las opciones de compra de los participantes de WIC y a otros patrones raciales/étnicos en las preferencias dietéticas. En este artículo se y métodos de desarrollo ágiles ninio a los que nos enfrentamos al crear nuestra aplicación para Android y se describe cómo los superamos aplicando la normalización de los datos, la suavización de la información y el uso de la tecnología.

Se autoriza la realización de copias digitales o impresas de la totalidad o parte de esta obra para uso personal o en el aula, sin necesidad de pagar nada, siempre que las copias no se realicen o distribuyan con fines lucrativos o comerciales y que las copias lleven este aviso y la cita completa en la primera página. Para copiar de otro modo, o volver a publicar, colgar en servidores o redistribuir a listas, se requiere un permiso específico previo y/o una tasa.

SIGPLAN'05 12-15 de junio de 2005, Lugar, Estado, País.
Copyright © 2004 ACM 1-59593-XXX-X/0X/000X...\$5.00

Categorías y descriptores de materias D.2.11 [Ingeniería de software]: Arquitecturas de software; H.2.1 [Gestión de bases de datos]: Diseño lógico; J.3 [Aplicaciones informáticas]: Ciencias médicas y de la vida

Palabras clave Aplicación móvil; Android; patrones de diseño de software, Open mHealth; Agile; normalización de datos; análisis de datos.

1. Introducción

El Programa Especial de Nutrición Suplementaria para Mujeres, Bebés y Niños (WIC) [1,2] es administrado por el Servicio de Alimentación y Nutrición (FNS) del Departamento de Agricultura de los Estados Unidos (USDA). El programa WIC concede subvenciones federales a los estados para la distribución de alimentos suplementarios, la derivación a servicios de salud y la educación nutricional para mujeres embarazadas, lactantes y puérperas sin lactancia de bajos ingresos, así como para bebés y niños de hasta cinco años que se encuentren en situación de riesgo nutricional.

Como parte del proyecto Nashville CHildren Eating Well (CHEW), financiado por el USDA, el objetivo de nuestro trabajo presentado en este artículo fue desarrollar una aplicación Android para ayudar a los padres y tutores de los niños inscritos en el programa WIC a realizar sus compras de forma más eficiente y eficaz, en un

para promover aperitivos y bebidas saludables en el entorno doméstico. Este artículo explora los retos tanto en el ámbito del programa WIC como en los enfoques técnicos que aplicamos al desarrollo de la aplicación CHEW. También presentamos nuestras soluciones a estos retos, basadas en la aplicación del diseño y la normalización del esquema de la base de datos, los patrones de software y los métodos de desarrollo ágiles. Por último, describimos nuestro trabajo futuro sobre el desarrollo de análisis de datos que analicen los datos de las compras de los participantes en el programa WIC para personalizar los consejos saludables para los usuarios y para ayudar a evaluar el impacto de la aplicación en la mejora de la educación nutricional.

El resto del documento está organizado como sigue: La sección 2 describe el dominio WIC y sus principales retos; la sección 3 resume la arquitectura e implementación de la aplicación CHEW; la sección 4 explica cómo abordamos los principales retos del dominio WIC; la sección 5 explica cómo abordamos los principales retos tecnológicos; la sección 6 compara nuestro trabajo en la aplicación CHEW con trabajos relacionados; y la sección 7 presenta las observaciones finales.

2. Una visión general del dominio WIC y de los principales retos del dominio

El programa WIC se centra en mejorar la salud y la nutrición de las mujeres embarazadas, las madres lactantes y los niños menores de cinco años con bajos ingresos. En la actualidad, el programa WIC atiende a más del 50% de los bebés nacidos en Estados Unidos. El USDA establece los requisitos generales del programa WIC, pero deja algunas áreas flexibles para que la agencia estatal tome decisiones selectas sobre cómo implementar el programa [1].

Los participantes en el programa WIC del estado de Tennessee reciben mensualmente dos tipos de vales en papel: un vale normal y un vale con valor en efectivo [2]. Los vales normales indican el paquete de alimentos del participante en función de su categoría de elegibilidad y edad (por ejemplo, mujer embarazada, madre lactante, bebé, niño de 3 o 4 años). Los vales ordinarios limitan las cantidades, tamaños y marcas de productos específicos aprobados por el programa WIC para cada participante. Los participantes reciben normalmente dos vales regulares al mes, con aproximadamente la mitad del paquete de alimentos en cada vale. Los vales de valor en efectivo, a diferencia de los vales normales, proporcionan un límite de cantidad en dólares (6 o 10 dólares) para la compra de frutas y verduras frescas o congeladas.

La experiencia de compra con vales de papel es complicada, ya que cada miembro de la familia inscrito en el programa WIC recibe vales regulares para diferentes paquetes de alimentos. Algunos productos (por ejemplo, la mantequilla de cacahuete, los alimentos para bebés, la leche de fórmula) se incluyen en ciertos paquetes de alimentos, pero no en otros, dependiendo de la etapa de la vida del participante. Además, cada paquete de alimentos tiene diferentes límites en cuanto a las cantidades y tamaños de los artículos en los vales normales. Algunos alimentos pueden elegirse de forma única o en cualquier combinación de cada uno de ellos. Por ejemplo, algunos vales ordinarios permiten a los participantes en el programa WIC elegir un máximo de tres leches secas descremadas o cualquier

combinación de tres leches de mantequilla, leches evaporadas y tofu. Para otros artículos, como los cereales, se pueden seleccionar varios artículos en función del número total de onzas de todos los paquetes.

Los vales de valor en efectivo se proporcionan en diferentes cantidades de dólares en función de la categoría de elegibilidad de los miembros de la familia. En el caso de los vales de valor en efectivo, en lugar de la cantidad, el tamaño o los artículos, los usuarios tienen un importe máximo en dólares para utilizar, por lo que la selección de artículos depende del precio. El uso de los vales de valor en efectivo requiere conocimientos matemáticos para calcular y hacer un seguimiento de los precios de las distintas combinaciones posibles de productos a granel con precio por peso, por pieza o por paquete. Este proceso puede ser complicado, especialmente para los participantes con escasos conocimientos de cálculo.

Los participantes en el programa WIC a menudo se enfrentan a retrasos a la hora de pagar en la tienda, ya que cada vale debe ser una transacción propia. Las familias con varios beneficiarios del programa WIC deben separar los artículos de cada participante antes de pasar por caja. Una vez en la caja registradora, los cajeros deben determinar si un artículo está autorizado para el vale o los vales utilizados. Además, cada vale de papel sólo puede utilizarse en una compra. Estos factores contribuyen a una utilización subóptima de las prestaciones, ya que los participantes en el programa WIC a menudo no compran todos los alimentos nutritivos que se les permite obtener con cada vale.

Las tecnologías de software móvil presentadas en este documento se diseñaron para simplificar la experiencia de compra de los participantes en el programa WIC. Para hacer frente a estos retos, se han establecido los siguientes requisitos y limitaciones en nuestra solución de software para la aplicación CHEW:

- *Minimizar la implicación del usuario:* los participantes en el programa WIC suelen sentirse frustrados al hacer la compra porque deben llevar un registro manual de todo lo que compran, lo que resulta tedioso y propenso a errores. Asimismo, los participantes con escasos conocimientos de aritmética tienen dificultades para calcular los precios de los productos hortofrutícolas. Uno de los objetivos de nuestra aplicación para Android era, por tanto, automatizar y simplificar al máximo la experiencia de compra de los usuarios.
- *Automatizar la adaptación y la entrega de la educación nutricional a través de la aplicación* - Los participantes de WIC actualmente reciben educación nutricional durante sus citas trimestrales en la clínica de WIC cuando reciben sus vales de papel. Sin embargo, también es beneficioso proporcionar contenido educativo nutricional complementario a los participantes entre las citas a través de la aplicación para teléfonos inteligentes. Así, nuestra aplicación para Android anima automáticamente a los tutores de los inscritos en el programa WIC a que sirvan a sus hijos pequeños (y posiblemente a toda la familia) tentempiés y bebidas saludables, ofreciéndoles consejos informativos y recetas de tentempiés fáciles de preparar.

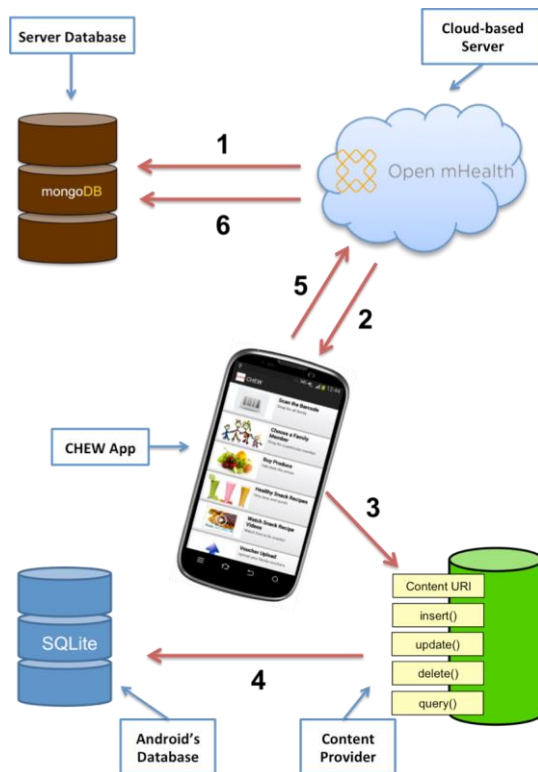
3. Visión general de la arquitectura e implementación de la aplicación CHEW

Antes de describir nuestras soluciones a los retos del dominio resumidos en la sección 2, esta sección presenta una visión general de la arquitectura de la aplicación CHEW, que se muestra en la figura 1. Un servidor basado en la nube ejecuta los servicios de almacenamiento de datos de Open mHealth [5] (que se analizan con más detalle en la sección 5.3), que proporciona una arquitectura abierta que mejora la integración entre las soluciones de salud móvil, y almacena la lista de elementos aprobados por el WIC en una base de datos MongoDB [6] (paso 1).

Cuando la aplicación CHEW se ejecuta por primera vez,

recupera los datos que contienen elementos aprobados por el WIC desde el servidor en la nube de Open mHealth (paso 2) y almacena en caché una copia de estos datos en el smartphone en un proveedor de contenidos de Android [3,4] (paso 3). Este proveedor de contenidos almacena los datos localmente en una base de datos SQLite [7] (paso 4), que encapsula los datos y gestiona su acceso por la parte de la aplicación CHEW dirigida al usuario.

Cuando la aplicación se utiliza durante un viaje de compras, almacena las opciones de compra de los usuarios en el proveedor de contenidos. Estos datos se sincronizan periódicamente con el servidor en la nube (paso 5),



realizamos varios viajes de compras con los usuarios para probar la aplicación en un entorno real de despliegue en las tiendas de comestibles locales. Estos breves sprints y las reuniones periódicas con

que lo almacena en una base de datos MongoDB para el posterior análisis de las elecciones de compra de los participantes (paso 6).

La aplicación CHEW está escrita en su mayor parte en Java, con cierto uso de sentencias SQL incrustadas en el proveedor de contenidos para acceder al WIC y a los datos de los usuarios almacenados en la base de datos SQLite. El número de líneas de código fuente de la aplicación CHEW es de ~8k y el número de clases es de ~70, excluyendo 3 bibliotecas de terceros derd.

4. Resolver los desafíos del dominio WIC

Como se indica en la sección 2, los principales retos de dominio a los que se enfrentaron los usuarios de las prestaciones del programa WIC fueron la dificultad para seleccionar y hacer un seguimiento de los artículos seleccionados aprobados por el programa WIC y la dificultad para calcular los precios de las frutas y verduras debido a las escasas habilidades numéricas de la mayoría de los participantes. Otro reto fue proporcionar la educación nutricional complementaria a través de la aplicación CHEW. Esta sección describe cómo aplicamos los principios del desarrollo ágil de software.

para ayudarnos a resolver estos retos de forma eficiente y eficaz.

4.1 Aplicación de métodos de desarrollo ágiles

Aplicamos los principios del desarrollo ágil de software mientras trabajábamos en la aplicación CHEW para lograr un desarrollo iterativo e incremental, lo que permitió un alto grado de colaboración con las partes interesadas, respuestas flexibles al cambio y (en última instancia) una mayor calidad y eficacia del software resultante. Empleamos sprints semanales que consistían en el diseño, el desarrollo y las pruebas. Al final de cada sprint nos reuníamos con los principales interesados en el proyecto para demostrar nuestro progreso en el desarrollo de la aplicación CHEW y clasificar sus comentarios.

También nos reunimos a menudo con los participantes de WIC para demostrar y probar nuestro prototipo, por ejemplo,

Figura 1: Arquitectura de la aplicación CHEW.

Las partes interesadas aumentaron la flexibilidad y la calidad del prototipo de la aplicación CHEW, por ejemplo, detectamos y corregimos errores y mejoramos el código más fácilmente a medida que el desarrollo avanzaba de forma incremental.

La figura 2 muestra la estructura y los ejemplos de hitos en el proceso de desarrollo de la aplicación CHEW. Aplicamos un modelo en espiral para introducirnos progresivamente en el programa WIC y comprender mejor el problema que debíamos resolver. Al mismo tiempo, planificamos funciones de la aplicación que ayudaran a simplificar la experiencia de compra de las familias del programa WIC y a automatizar y personalizar partes clave de su educación nutricional. También tuvimos que aprender y dominar las funciones de software de Android necesarias para crear la aplicación y resolver los retos del ámbito del WIC.

La figura 2 también muestra varios ejemplos del proceso en espiral cuando aplicamos los principios ágiles de desarrollo de software. En concreto, a medida que íbamos conociendo mejor las funciones de Android y Open mHealth, estábamos mejor equipados para diseñar implementaciones de software adecuadas, lo que a su vez nos ayudó a ofrecer soluciones más eficaces a los participantes de WIC.

4.2 Minimizar la participación de los usuarios

Como se ha comentado en el apartado 2, los usuarios de las prestaciones WIC tienen actualmente una experiencia complicada, ya que deben comprobar manualmente que seleccionan los artículos WIC aprobados durante sus viajes de compra, así como llevar un seguimiento manual de los precios y las cantidades de los artículos. A continuación describimos cómo hemos diseñado la aplicación CHEW para automatizar y simplificar al máximo la experiencia de compra de los usuarios.

4.2.1 Automatización de la experiencia de compra del WIC

La aplicación CHEW necesitaba un medio que permitiera a los participantes transferir al dispositivo toda la información sobre un artículo alimentario concreto, notificar al usuario que el artículo está aprobado para su compra y, a continuación, hacer un seguimiento de su cantidad. Llegamos a la conclusión de que escanear el código de barras de un producto sería la forma más eficaz de automatizar estas tareas. Por lo tanto, programamos la aplicación CHEW para que almacenara una base de datos de códigos de barras de artículos aprobados por el WIC, junto con todos los metadatos necesarios sobre los artículos, como sus nombres (por ejemplo, leche entera) y cantidades (por ejemplo, 16 onzas). Al cotejar un código de barras escaneado con el código de barras almacenado en una base de datos del teléfono inteligente, la aplicación recuperaría la información que necesita.

Desarrollar nosotros mismos una herramienta de escaneo de códigos de barras supondría reinventar soluciones que ya existen, así que exploramos los paquetes disponibles y elegimos ZXing ("Zebra Crossing") [8], que es una biblioteca de procesamiento de imágenes de códigos de barras de código abierto. Como des-

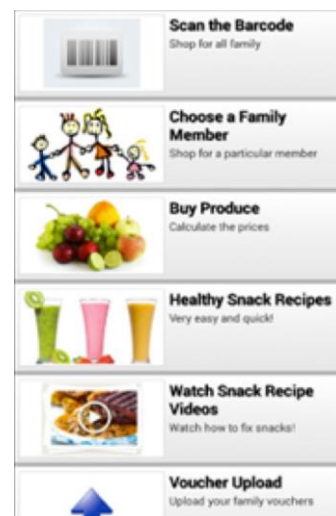


Figura 3. Pantalla principal de la aplicación.

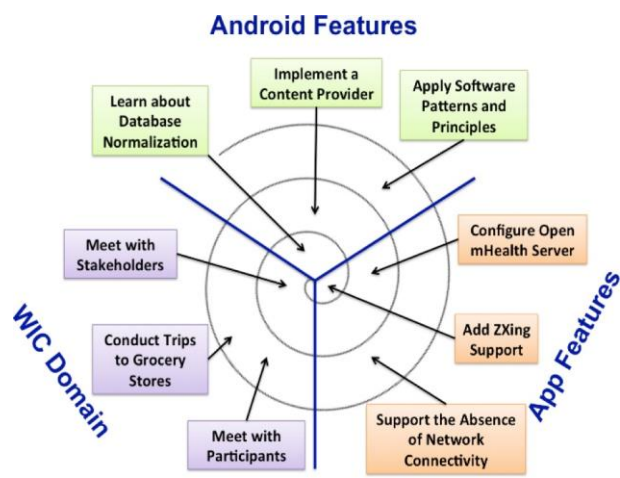
Como se ha explicado en el apartado 5.2, una de las limitaciones a las que nos enfrentamos durante el desarrollo de la aplicación CHEW fue la de poder utilizar las funciones de la aplicación sin conexión a la red, ya que no podíamos confiar en que las familias del WIC tuvieran un plan de datos o acceso a la red Wi-Fi en las tiendas de alimentación. El uso de la biblioteca ZXing resultó ventajoso en este caso porque utilizaba la cámara integrada en los dispositivos móviles para escanear y descodificar los códigos de barras en el dispositivo, sin necesidad de comunicarse con un servidor remoto. Para apoyar las diferentes preferencias de los usuarios y simplemente el uso de la aplicación CHEW, permitimos a los participantes de WIC comprar de dos maneras diferentes: (1) comprar para toda la familia a la vez o (2) comprar para cada miembro de la familia por separado. Podrían intercambiar estos métodos de compra cualquier número de veces durante sus compras.

La figura 3 muestra la pantalla principal de la aplicación CHEW. Un usuario puede hacer clic en el botón "Escanear el código de barras", donde después de escanear un artículo la aplicación mostraría todos los miembros de la familia que podrían obtener este artículo basándose en la información de sus vales. El comprador del programa WIC puede entonces seleccionar el número de artículos que debe recibir cada miembro de la familia. A la inversa, el comprador podría seleccionar artículos para un miembro específico de la familia haciendo clic en el botón "Elegir un miembro de la familia". En este caso, se le presentaría una *ListView* [4] de los miembros de la familia, en la que al seleccionar un miembro específico se abre el perfil del mismo y permite al comprador ver los artículos que puede obtener este miembro en particular, lo que ya ha comprado, y una opción para escanear artículos específicamente para este miembro.

Mientras un usuario escanea los artículos, la aplicación CHEW consulta al proveedor de contenidos que almacena los datos de los artículos y vales en la base de datos SQLite para comprobar si un artículo está aprobado y si se ha alcanzado su límite. La programación de esta lógica en la aplicación fue uno de los retos técnicos más difíciles a los que nos enfrentamos, como se explica en la sección 5.1.

4.2.2 Proporcionar la calculadora de productos hortofrutícolas

Como se ha comentado en el apartado 2, muchos usuarios de las prestaciones del programa WIC tienen conocimientos limitados de aritmética, lo que les dificulta calcular los precios de las frutas y verduras frescas y hacer un seguimiento de los precios para saber cuándo alcanzan el límite especificado en los vales de valor en efectivo para cada miembro de la familia. En consecuencia, muchos participantes no suelen utilizar el vale de valor en efectivo en toda su extensión. Por tanto, tuvimos que idear una solución



para ayudar a los participantes en el programa WIC a sacar el máximo partido a sus vales de valor en efectivo.

Un vale de valor en efectivo de WIC especifica la cantidad de dólares hasta la que los participantes pueden comprar frutas y verduras, que incluyen productos frescos, envasados, productos congelados envasados, productos a granel

Figura 2.1 Proceso de desarrollo de la aplicación CHEW.

con un precio por artículo, y productos sueltos con un precio por libra. Hemos creado una Actividad Android independiente (que es una cosa única y focalizada que un usuario puede hacer, como interactuar con el usuario a través de la interfaz de la pantalla táctil de un dispositivo móvil) para tratar el vale de valor en efectivo. Como se muestra en el centro de la Figura 3, un usuario puede hacer clic en el botón "Comprar productos" para abrir una actividad que le presenta información sobre el precio total permitido para la familia, la cantidad total ya gastada y la cantidad total restante, así como las cantidades para cada miembro de la familia por separado. También se presentan al usuario las opciones para comprar cuatro tipos diferentes de productos, por ejemplo, frescos envasados, congelados envasados, productos a granel con un precio por artículo, y productos a granel con un precio por libra.

Uno de los objetivos del proyecto CHEW es analizar los datos de las compras para estudiar el impacto de la aplicación CHEW en las elecciones de compra de los participantes de WIC. Por lo tanto, era importante que la aplicación CHEW recogiera los nombres de todos los artículos que compran los participantes, pero minimizando la participación del usuario. Cumplir este objetivo fue difícil cuando se trata de productos agrícolas, ya que los precios cambian a menudo y esta información no se almacena en la base de datos del teléfono inteligente. Como resultado, no pudimos automatizar completamente este proceso en nuestro prototipo; una solución más completa probablemente implicaría una estrecha interacción con los sistemas de precios electrónicos utilizados por las tiendas de comestibles que participan en el programa WIC.

Como los productos frescos y congelados envasados tienen códigos de barras, permitimos a los usuarios que simplemente escanearan los paquetes para obtener los nombres de los productos. A continuación, introdujeron los precios y seleccionaron las cantidades de productos y los miembros de la familia a los que se aplican estos productos. Dado que los participantes pueden escanear los envases, también podían comprar la parte de productos envasados del vale de valor en efectivo cuando escaneaban artículos de sus vales normales. Si escaneaban un artículo de productos agrícolas, simplemente se les transfería a la actividad de manipulación de productos agrícolas.

Dado que era inviable escanear productos sueltos (que carecen de códigos de barras estándar), los participantes tenían que introducir manualmente los nombres de los artículos. Escribir el nombre completo de un artículo sería complicado para un usuario, por lo que utilizamos la opción de autocompletar que ofrece la clase *AutoCompleteTextView* [4] de Android. Implementamos esta función creando un *CursorAdapter* [4] personalizado que implementa una interfaz *Filterable* [4] y lo conectamos a una *AutoCompleteTextView*, que luego mostró a los usuarios los nombres sugeridos de frutas y verduras almacenados en el proveedor de contenidos.

La compra de "productos a granel por artículo" requiere que el usuario introduzca el precio de un artículo. Del mismo modo, la compra de "productos a granel por kilo" requiere que el usuario pese el producto y calcule el precio. Una vez más, nuestro objetivo era hacer este proceso lo más sencillo posible para los usuarios. Como los usuarios tenían que pesar los artículos en una balanza, pensamos que encontrar un widget que imitara una balanza de productos reales sería la interfaz más intuitiva. Elegimos el widget

de la rueda de Android [9].



CASH VOUCHER	
\$ 1.75	Apples
\$ 3.23	Peaches
\$ 0.87	Cucumbers
\$ 1.75	Carrots
DAIRY	
1	PRTY FF BUTTERMILK
2	NSYA EXTRA FIRM TOFU
FRUITS & VEGETABLES	
2	JCYJCE 100% APPLE JUICE
GRAINS	
1	KRO WHOLE WHEAT BREAD
PROTEIN	
1	KRO GRADE A LRG EGGS RPC
1	PTRPN CRMY PEANUT BUTTER
13	oz FRDA PEAS BLACKEYED DRI...

La figura 4 muestra un primer plano de cómo utilizamos este widget en nuestra aplicación CHEW. Los usuarios pueden desplazar la rueda izquierda para elegir las libras tal y como las ven en la báscula real, y pueden desplazar la rueda derecha para seleccionar los incrementos. A continuación, pueden seleccionar el miembro de la familia que recibe el artículo e introducir el nombre del producto y su precio por libra. Debido a pequeñas discrepancias, la aplicación sólo calculaba el coste aproximado de los productos, pero nuestras reuniones periódicas con los participantes en el programa WIC indicaron que seguía siendo bastante útil para ellos.

A medida que los participantes en el programa WIC compran productos, la aplicación CHEW hace un seguimiento de los precios y actualiza la pantalla de la actividad con los importes en dólares gastados y los que quedan en el vale. Así, los participantes pueden concentrarse en sus compras, en lugar

Figura 4. Calculadora de producto por libra.

de calcular y gestionar los precios manualmente, lo que resulta tedioso y propenso a errores.

4.2.3 Seguimiento automático de las selecciones

La aplicación CHEW hace un seguimiento de todos los alimentos que eligen los participantes de WIC, incluyendo las cantidades y los tamaños de cada uno. Un comprador de WIC puede ir al perfil de cada miembro de la familia y ver los artículos seleccionados para cada miembro. La figura 5 muestra una actividad que muestra los productos elegidos en un *ListView*. Escribimos un *SimCursorAdapter* [4] personalizado para mostrar los artículos en un formato similar al presentado en los vales (mostrado en la Figura 7).

Por ejemplo, los vales especifican límites en el número de artículos, en el número de onzas o en el importe en dólares. Por lo tanto, nuestro adaptador personalizado aprovecha los diferentes diseños de fila de *ListView* para mostrar el número de artículos, las onzas o los dólares. Los separadores del *ListView* también ayudan a dividir los artículos en categorías.

4.3 Automatización de la adaptación y el suministro de

Figura 5. Elementos seleccionados Actividad.

educación nutricional

Además de ayudar a las familias del programa WIC en su experiencia de compra, como se ha descrito anteriormente, la aplicación CHEW también pretende proporcionar educación nutricional, como se describe a continuación.

4.3.1 Proporcionar recetas de aperitivos saludables y atractivos

Nuestra aplicación CHEW permite a los padres y tutores ver y seleccionar recetas de aperitivos saludables fáciles de preparar para sus hijos. Cada receta contiene un número limitado de pasos, que se muestran mediante texto e imágenes para hacerla más entretenida y fácil de seguir.

Para simplificar aún más este proceso, la aplicación CHEW recuerda a los participantes que deben elegir algunas de las recetas antes de cada viaje de compras.

Cuando los usuarios seleccionan una receta, sus ingredientes se cargan automáticamente en una lista de la compra que se utilizará durante la misma. Estos ingredientes no se corresponden necesariamente con los artículos que figuran en los vales WIC, y los participantes pueden comprarlos con los vales o por separado.

Para que la aplicación CHEW resulte más atractiva, ofrece videos de recetas que muestran cómo preparar tentempiés saludables. Aunque no todos los participantes pueden ver los videos, ya que se transmiten por Internet, hemos incluido esta opción para que los usuarios puedan utilizarla siempre que puedan. La figura 6 muestra las actividades de Android que permiten a los usuarios ver/elegir recetas de aperitivos saludables y ver videos de recetas.

4.3.2 Envío de notificaciones de consejos saludables

Además de ofrecer una galería de recetas de bocadillos saludables, otro componente de la educación nutricional en el que hace hincapié el proyecto CHEW es el de los consejos saludables proporcionados a través de notificaciones. La aplicación CHEW almacena una lista de consejos saludables generales en su proveedor de contenidos. Algunos ejemplos de estos consejos son ayudar a los padres a lidiar con los niños que son quisquillosos con la comida (por ejemplo, los padres pueden necesitar ofrecer un nuevo alimento hasta 15 veces antes de que su hijo se lo coma), el tamaño adecuado de las porciones para los niños en edad preescolar y la distribución de bocadillos saludables a lo largo del día.

La aplicación CHEW utiliza los componentes Android *Service*, *BroadcastReceiver* y *AlarmManager* [3,4] para enviar notificaciones a los usuarios en intervalos de tiempo específicos (por ejemplo, cada dos días, etc.). En el futuro, realizaremos un análisis de datos para personalizar las notificaciones para cada participante, en función de su origen étnico y de sus datos de compra, como se explica en la sección 7.

5. Resolver los desafíos técnicos

Nos enfrentamos a varios retos técnicos durante el diseño y la implementación de la aplicación CHEW. Algunos de ellos se debieron a la imposibilidad de utilizar funciones eficientes de Android, ya que tuvimos que suplir la ausencia de conectividad a la red. En esta sección se describen estos retos clave y se presentan nuestras soluciones.

5.1 Tratar con datos complicados y no estructurados que pueden cambiar con frecuencia

Problemas. Durante las primeras etapas del desarrollo de la aplicación CHEW nos dimos cuenta de que la aplicación tendría que almacenar y procesar una gran

La falta de conectividad a la red, que se comenta en el apartado 4.2.1, hizo que los datos se almacenaran localmente en el smartphone. Además de los datos de las compras, tuvimos que almacenar los datos que contenían las descripciones de los productos de las tiendas de comestibles participantes y los datos que contenían las descripciones de los garantes, lo que fue especialmente complicado. A continuación se describen las técnicas para superar los problemas relacionados con el almacenamiento y el procesamiento eficaz de estos datos.

Uno de los principales retos a los que nos enfrentamos fue cómo almacenar y procesar las descripciones de los vales que especificaban los límites de las marcas, cantidades y tamaños de los alimentos. Esta información no se presentaba en un formato común y contenía una lógica complicada. La figura 7 muestra tres ejemplos de estos vales. El vale de la parte superior de la figura permite al usuario obtener hasta una cantidad específica de un alimento concreto, como 36 onzas de cereales, o hasta un número específico de un producto concreto o hasta un número específico de un producto diferente, pero no ambos, por ejemplo, 3-11,5 a 12 onzas congeladas o 3-46 a 48 onzas de zumo aprobado por el WIC.

Un ejemplo más complicado es cuando un usuario puede obtener alguna cantidad de un artículo concreto o cualquier combinación de artículos, pero no ambos, por ejemplo, una caja de 1 a 3 cuartos de leche en polvo descremada o elegir 3 (cualquier combinación) de estos: 1 litro de suero de leche, 1 lata de leche evaporada, 14-16 onzas de tofu. Además, cada usuario puede obtener una cantidad determinada de productos en dólares, que se le entrega en forma de otro tipo de vale con valor en efectivo.

El vale del centro de la figura 7 especifica una lógica diferente, por ejemplo, este vale no permite al usuario obtener una combinación de artículos. Por último, el vale de la parte inferior de la figura es totalmente diferente de los dos anteriores; especifica alimentos para bebés, dependiendo de si el bebé es alimentado total o parcialmente con leche materna o con leche artificial. Los tipos y las cantidades de alimentos que puede recibir un bebé también dependen de su edad.

Las descripciones de los vales que se muestran en la figura 7 son muy poco estructuradas, lo que supone un reto a la hora de almacenar estos datos de forma estructurada en una base de datos relacional, como SQLite, al tiempo que se preserva la lógica de las descripciones. Además, los vales pueden dividirse y los usuarios podrían (1) utilizar ambos vales en un solo viaje de compras o (2) pueden dividir cada uno durante dos viajes de compras diferentes. Además, si los usuarios reciben sus vales más tarde de lo previsto, recibirán vales diferentes, con cantidades reducidas de alimentos permitidos.

Otro reto era vincular los datos de los productos de las tiendas con las descripciones de los vales para que la aplicación identificara correctamente si las elecciones de los usuarios eran correctas, que era el requisito fundamental. Por ejemplo, los conjuntos de datos proporcionados por las tiendas de comestibles solo contenían nombres de alimentos específicos y categorías de alimentos más amplias, como los lácteos, mientras que los vales especificaban los límites de diferentes tipos de alimentos, como la leche o el queso, dentro de una categoría de alimentos, como los lácteos. Tuvimos que encontrar una solución con la que la aplicación CHEW entendiera la conexión entre los datos de la tienda y los datos de los vales para que el usuario de la aplicación pudiera hacer las elecciones adecuadas y hacer un seguimiento de las cantidades elegidas y los tamaños de los alimentos.


Por último, los artículos aprobados por el programa WIC en cada tienda cambian a menudo, así como la información de los vales; por ejemplo, se añaden algunos tipos de alimentos y se eliminan otros, y las cantidades permitidas también cambian. Por lo tanto, necesitábamos una solución que requiriera pequeños cambios cada vez que se actualizaran los conjuntos de datos de las tiendas y los vales.

Solución. Utilizamos la normalización de los datos para organizarlos en la base de datos SQLite de Android. Aplicamos la tercera forma normal (3NF) [10,11] para dividir grandes



cantidades de datos en pequeñas tablas. Aunque esta solución dio lugar a una proliferación de tablas, pudimos dotar de mucha más estructura a datos que inicialmente parecían bastante desestructurados, como las diferentes descripciones de los vales. 3NF hizo más flexible nuestra base de datos al eliminar la redundancia y la dependencia incoherente. En particular, garantizó que cada tabla

WIC Foods for Partially Breastfeeding Mom



Your WIC Foods:


- Increase your choice of food
- Offer a variety of fruits and vegetables
- Help improve your family's health
- Follow the Dietary Guidelines and MyPyramid recommendations

What You Will Receive:

GRAINS	FRUITS and VEGETABLES	DAIRY	
36 oz cereal 1 - 12 to 16 oz whole wheat bread or other whole grain products such as: Brown rice Bulgur Oatmeal Barley Soft corn tortillas Whole wheat tortillas	3 - 11.5 to 12 oz frozen or 3 - 46 to 48 oz containers of WIC approved juice \$10 cash value voucher for fresh or frozen fruits and vegetables	4 gallons milk - Reduced Fat, Fat Free, Low Fat, or Sweet Acidophilus 1-3 quart box nonfat dry milk or choose 3 (any combination) of these: 1 quart buttermilk 1 can evaporated milk 14-16 oz of tofu 16 oz cheese	1 dozen eggs 16 oz dried beans/peas 1 - 16 to 18 oz jar of peanut butter

<http://health.state.tx.us/wic>

WIC Foods for your 2 year old Child (24 through 35 months)



Your WIC Foods:


- Increase your choice of food
- Offer a variety of fruits and vegetables
- Help improve your family's health
- Follow the Dietary Guidelines and MyPyramid recommendations

What You Will Receive:

GRAINS	FRUITS and VEGETABLES	DAIRY	PROTEIN
36 oz cereal 2 - 12 to 16 oz whole wheat bread or other whole grain products such as: Brown rice Bulgur Oatmeal Barley Soft corn tortillas Whole wheat tortillas	2 - 64 oz containers WIC approved juice \$6 cash value voucher for fresh or frozen fruits and vegetables	3 gallons milk - Reduced Fat, Fat Free, Low Fat, or Sweet Acidophilus 1 quart buttermilk or 1 can evaporated milk 16 oz cheese	1 dozen eggs 16 oz dried beans/peas

<http://health.state.tx.us/wic>

WIC Foods for your Infant (Birth through 11 months)



Your WIC Foods:

- Increase your choice of food
- Offer a variety of fruits and vegetables
- Help improve your family's health
- Follow the Dietary Guidelines and MyPyramid recommendations

What You Will Receive:

GRAINS	FRUITS and VEGETABLES	DAIRY	PROTEIN
All infants at 6 months: 3 - 8 oz boxes of infant cereal	Fully breastfeeding infants at 6 months: 64 - 4oz containers infant fruits & vegetables (Stage 2 or 2nd Stage foods only) Partially breastfed and fully formula fed infants at 6 months: 32 - 4 oz containers of infant fruits & vegetables (Stage 2 or 2nd Stage foods only)	Fully breastfed infants: Mom's breastmilk! Partially breastfed infants: Mom's breastmilk and infant formula in amounts to supplement your baby's needs Formula fed infants: Number of cans will depend on powder or concentrate, and age of infant	Fully breastfed infants at 6 months: 31 - 2.5 oz jars of baby food meat

<http://health.state.tx.us/wic>

Figura 7. Ejemplos de vales WIC.

tenía una clave, donde los atributos no clave dependían sólo de la clave y de ningún otro atributo.

Consideraciones de diseño. Aplicar 3NF en exceso no suele ser práctico. Por ejemplo, tener muchas tablas pequeñas puede degradar el rendimiento. Por lo tanto, aplicamos 3NF sólo a los datos que cambiaban con frecuencia, como las descripciones de los vales. A continuación describimos el proceso que utilizamos para organizar los datos de los vales no estructurados en una base de datos SQLite de Android.

Proceso de diseño. La figura 8 muestra una parte del esquema de la base de datos SQLite de la aplicación CHEW que gestiona los datos de una tienda y los vales. Hemos creado identificadores para todos los posibles vales (voucher-

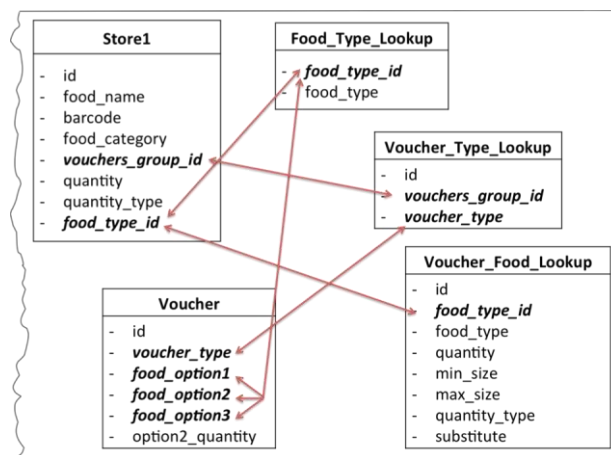


Figura 8. Parte del esquema de la base de datos.

er_type). Como los vales podían dividirse, los tratamos como vales diferentes y creamos identificadores separados. Si los usuarios especificaban que querían utilizar todo el vale a la vez, la aplicación simplemente realizaba uniones de tablas al consultar la base de datos. Utilizamos una lógica similar para los vales con cantidades reducidas de productos que los participantes reciben cuando llegan tarde a recoger los vales.

Dado que un artículo alimentario concreto puede aplicarse a varios vales, también creamos un identificador único para un grupo de identificadores de vales a los que se aplica este artículo (vouchers_group_id). Por ejemplo, un "1" podría representar los vales V1, V2 y V4, que permiten comprar pan. También creamos identificadores para diferentes tipos de alimentos, como los cereales o la leche (food_type_id).

Desgraciadamente, tuvimos que introducir manualmente estos identificadores y el identificador único de un grupo de vales en las bases de datos recibidas de las tiendas de alimentación participantes (la automatización de este proceso está prevista en nuestro futuro trabajo, como se explica en la sección 7). A continuación, creamos una tabla de búsqueda para hacer coincidir los identificadores de los alimentos con los nombres reales de los tipos de alimentos (Food_Type_Lookup) y una tabla de búsqueda para hacer coincidir el identificador del grupo de vales con los identificadores de cada uno de los vales (Voucher_Type_Lookup). Estas tablas de búsqueda nos permitieron crear un vínculo entre los datos de las tiendas y los datos de los vales, lo que fue una característica clave para que la aplicación CHEW cumpliera los requisitos del dominio presentados en la sección 2.

Para organizar los datos y permitir consultas uniformes en la base de datos, creamos una tabla (Voucher) que contiene cada identificador de voucher y tres columnas adicionales para especificar las posibles opciones de comida (un máximo de tres en la actualidad), aunque las descripciones de los vouchers tuvieran las tres opciones. Si un vale especificaba una opción para un tipo de comida, la fila contendría el identificador de ese tipo de comida (en caso contrario, contendría un identificador negativo). La parte más difícil de nuestra solución era manejar los datos que especificaban que algunos alimentos podían comprarse en una combinación. Así, identificamos una opción de tipo de alimento que podría permitir combinaciones (food_option2) y creamos una columna separada en la tabla, especificando un número de artículos combinados (option2_quantity). Si las combinaciones no estaban permitidas para un tipo de alimento concreto, la columna contenía un cero.

También creamos una tabla independiente con el identificador del tipo de comida como clave (Voucher_Food_Lookup); esta tabla especificaba las cantidades y tamaños permitidos de los tipos de comida. Una vez más, tuvimos que idear una forma estructurada de incorporar los datos relativos a los artículos de la tabla que podían comprarse de forma combinada. Para resolver este problema, creamos una columna separada para contener

enlaces a otros tipos de alimentos en la misma tabla (sustituto).

Por ejemplo, algunos vales permitían comprar suero de leche, leche evaporada y tofu en cualquier combinación de tres. Si identificamos un suero de leche como "7", la leche evaporada como "8" y el tofu como "9", entonces "7" señalaría "8", "8" señalaría "9" y "9" señalaría "7". Si no se permitían combinaciones, la columna contenía un identificador negativo. Este diseño nos permitía consultar la tabla de vales para comprobar si existía un número para la combinación de artículos, así como utilizar esta información para consultar esta tabla y ver qué tipos de alimentos reales podían comprarse en una combinación.

Aunque la mayoría de los vales eran relativamente similares, los vales para bebés eran diferentes, como se muestra en la figura 6. Por lo tanto, separamos la información sobre estos vales en tablas de base de datos separadas (que no se muestran en la figura).

Resultado. Una vez organizados los datos de los vales no estructurados en la base de datos SQLite, nuestra aplicación CHEW tenía que utilizar consultas complicadas que implicaban múltiples uniones de tablas para obtener la información necesaria. La aplicación de 3NF, sin embargo, redujo el posible número de cambios en la base de datos y en el código que haríamos en el futuro al minimizar las dependencias entre los datos. Este enfoque también desvinculó los datos y la lógica para utilizarlos, especificando los datos en una base de datos (incluso los datos para la lógica), pero realizando la lógica de forma programada.

5.2 Apoyar la ausencia de conectividad a la red

Problemas. Muchas de las familias que participan en el programa WIC carecen de planes de datos o de conexión a Internet en casa, lo que supuso un reto a la hora de ofrecer ciertas funciones de la aplicación CHEW, como la descarga de imágenes de recetas desde un servidor HTTP remoto y la transmisión de vídeos de recetas. Nuestro objetivo era garantizar que los tutores y sus hijos vieran estas recetas y que éstas fueran fáciles de seguir.

Para ser más atractivas, las recetas debían contener imágenes de cada paso, idealmente presentadas como vídeos de alta resolución paso a paso. Para evitar que los participantes en el programa WIC dediquen toda la memoria de su smartphone a almacenar este contenido, los vídeos deben transmitirse y las imágenes deben descargarse de un servidor remoto y almacenarse en la memoria caché de un dispositivo, lo que motiva la necesidad de conectividad a la red.

Dado que no podíamos confiar en que las familias de WIC tuvieran acceso a Internet, tuvimos que asegurarnos de que la aplicación CHEW funcionara correctamente proporcionando recetas atractivas, sin depender de la red. Una opción era excluir las imágenes de todos los pasos de la receta e incluir únicamente las imágenes de un número limitado de recetas que la aplicación pudiera almacenar localmente en el dispositivo. Sin embargo, leer los pasos de la receta en texto plano sería menos convincente para los participantes y requeriría una mayor participación del usuario. Por lo tanto, tuvimos que mostrar imágenes de cada receta y de cada paso de la misma para hacerla más atractiva para los usuarios, pero tuvimos que encontrar una solución para hacerlo localmente en el dispositivo.

Solución. Como resulta problemático almacenar un gran número y tamaño de imágenes, tuvimos que llegar a un compromiso con nuestros interesados. Terminamos utilizando sólo un pequeño número de recetas, donde cada receta contiene como máximo dos pasos, por lo tanto, como máximo dos imágenes, y donde todas las imágenes son pequeñas. La aplicación CHEW almacena las imágenes en la carpeta de recursos de Android (res) y las rutas de las imágenes en la base de datos SQLite.

Resultado. Aunque esta solución no es la más eficiente para tratar las imágenes en Android, era necesaria dadas las dificultades a las que nos enfrentamos. A medida que mejore el acceso ubicuo a la conectividad a Internet, esperamos volver a examinar todo el potencial de formas más eficientes de almacenar, recuperar y mostrar imágenes.

También dejamos la opción de los vídeos de recetas para los usuarios aunque no puedan ver los vídeos a menudo. Utilizamos la API de YouTube [12] para transmitir los vídeos; un usuario puede ver los vídeos

Las miniaturas se muestran en un *ListView* en el que al hacer clic en una miniatura se muestra el vídeo al usuario (Figura 6). En futuros trabajos podemos planear tener una única Actividad que muestre una receta con su imagen y vídeo. Si la conectividad a Internet está presente, mostraremos una miniatura de vídeo para que el usuario pueda ver el vídeo de la receta si es necesario, y si la conectividad no está presente, simplemente mostraremos una imagen de la receta solamente.

5.3 Encontrar una forma eficaz de analizar los datos

Problemas. Un objetivo clave del proyecto CHEW es recoger y evaluar los datos de las compras de los participantes para acceder al impacto de la educación nutricional proporcionada a través de la aplicación y realizar los ajustes necesarios en el programa. Sin embargo, cuando los compradores del programa WIC utilizan vales de papel, es inviable recoger sus elecciones de compra para analizar esta información con fines de estudio nutricional. Por lo tanto, necesitábamos una solución que permitiera un análisis de datos eficiente para ayudar a los estudios de nutrición.

Solución. Para implementar esta solución, empleamos Open mHealth [5], que es una arquitectura abierta diseñada para mejorar la integración entre las soluciones de salud móvil. La plataforma Open mHealth fomenta la colaboración entre los desarrolladores de software, los expertos clínicos y los investigadores de la salud para abordar el problema de extraer significado e inferencias científicamente válidas a partir de los datos de mHealth recopilados, que a menudo implican muchos sesgos y variabilidad, presentando

herramientas más sofisticadas y eficaces para la visualización y el análisis de datos [13].

Visión general de Open mHealth. Como se muestra en la figura 9, Open mHealth ayuda a que el ecosistema de la sanidad móvil pase de una arquitectura en silos a una arquitectura de "cintura estrecha", en la que un protocolo de comunicaciones común actúa como un simple punto común en la cintura estrecha, y en la que la innovación florece a través de interfaces o API abiertas, tanto por encima como por debajo de la cintura.

Open mHealth ofrece las siguientes características [14]:

- *Reutilización del software* para ayudar al desarrollo de nuevas aplicaciones mediante el intercambio de componentes de software entre los desarrolladores de software y los innovadores de la salud.
- *Estandarización*, proporcionando un conjunto compartido de APIs abiertas para los almacenes de datos del back-end para permitir que el software del cliente tenga una manera uniforme de acceder a los datos.

La figura 10 muestra los principales componentes de Open mHealth, que se definen como las unidades modulares de software que se describen a continuación:

- *Unidad de almacenamiento de datos (DSU).* Una DSU proporciona una forma uniforme de acceder a los datos a través de una serie de llamadas a la API.
- *Unidad de Procesamiento de Datos (DPU).* Una DPU representa una unidad de trabajo que se realiza sobre los datos JSON. Es el bloque de construcción para extraer características relevantes de los flujos de datos.

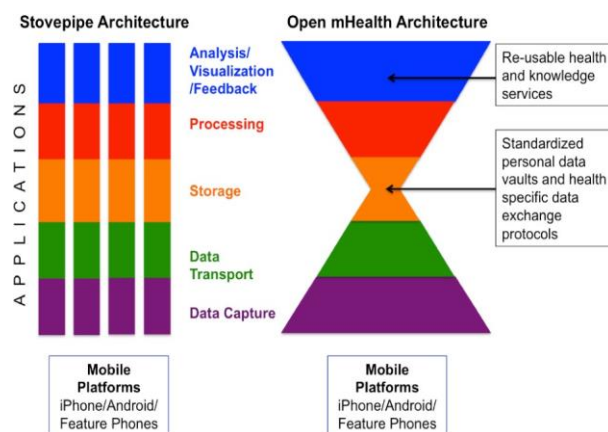


Figura 9. Arquitectura del tubo de la estufa frente a la del reloj de arena.

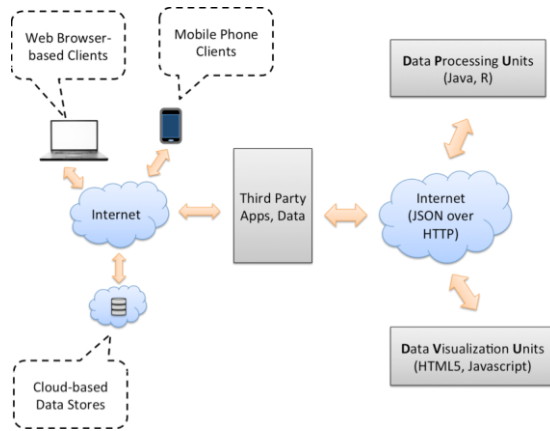


Figura 10. Componentes abiertos de mHealth.

- **Unidad de visualización de datos (DVU).** Una DVU permite la presentación visual de las características procesadas por una DPU y los patrones.

Los módulos abiertos de visualización de mHealth pueden combinar datos de muchas aplicaciones y dispositivos móviles. Asimismo, los módulos de evaluación pueden integrarse directamente en una aplicación para mejorar el impacto clínico.

Configuración abierta de mHealth. Alojamos un servidor basado en la nube utilizando Amazon Web Services [15]. Como se muestra en la Figura 1, el servidor ejecuta una base de datos MongoDB que interactúa con Open mHealth. La base de datos del servidor almacena los datos de los productos de las tiendas de comestibles, que se cargan en la base de datos SQLite de un dispositivo móvil a través del *IntentService* de Android [3,4] cuando la aplicación se ejecuta por primera vez. La aplicación puede cargar periódicamente los datos de las compras de los usuarios al servidor para su futuro análisis, suponiendo que los usuarios puedan obtener conectividad de red, por ejemplo, durante su visita a la clínica WIC.

Resultado. Al alojar un servidor basado en la nube que ejecuta Open mHealth, creamos una plataforma que permite el procesamiento eficiente de los datos recopilados para obtener información útil para la acción. Open mHealth también facilita el intercambio de datos entre plataformas, lo que mejorará las futuras iteraciones del proyecto.

5.4 Aplicación de patrones de software

Problemas. Como se menciona en el apartado 5.1, esperamos que se produzcan cambios frecuentes en nuestra aplicación CHEW, por ejemplo, cuando se añadan, actualicen o eliminen del programa vales y productos. Hemos hecho un esfuerzo considerable para que el esquema de la base de datos sea flexible a los cambios, como se muestra en la figura 8. También diseñamos nuestra aplicación para que requiriera un mínimo de cambios en el código a medida que cambiaban los requisitos, aplicando principios y patrones de software, tanto por medio de nuestros propios patrones como aplicando marcos de trabajo de Android que aprovechan los patrones para mejorar la reutilización del código y hacer que la aplicación sea más flexible a los cambios.

Soluciones. Utilizamos el análisis de homogeneidad y variabilidad [16] para separar las partes variables de la aplicación de las que no lo son. Por ejemplo, tuvimos que representar a los usuarios de la aplicación y sus vales. Llegamos a la conclusión de que los vales que reciben los usuarios representan las partes variables, ya que pueden recibir unos diferentes cada vez y las descripciones de los vales cambian a menudo. Por lo tanto, aplicando el análisis de la homogeneidad y la variabilidad, identificamos que el patrón *Strategy* [17] ayudaría a desacoplar las interfaces de las implementaciones para que éstas puedan variar sin afectar al código del cliente que utiliza las interfaces comunes.

Aplicando la *estrategia*, creamos una clase abstracta *Member*,

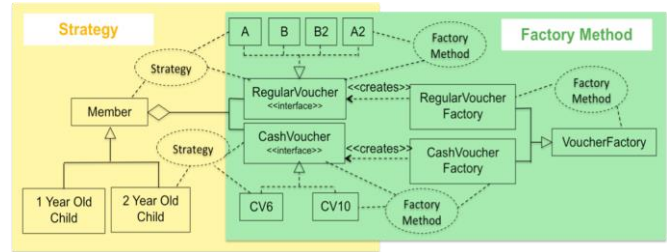


Figura 11. Estrategia y método de fábrica.

que representa a un usuario de la aplicación, y varias clases concretas que heredan de *Member*, que representan tipos de usuarios reales, como un niño de 3 a 4 años (Figura 11). A continuación, creamos interfaces para

voucher y vouchers de valor en efectivo, y varias clases que representan un voucher específico que implementan una de las interfaces.

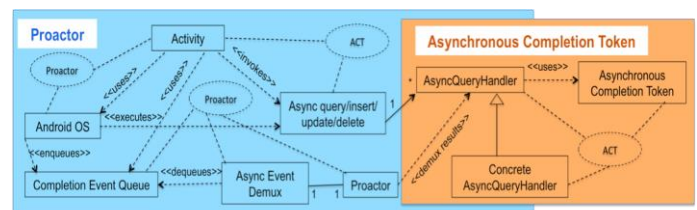
Utilizando el patrón *Strategy*, en el que cada clase concreta *Member* se compone con los vouchers apropiados, también pudimos aplicar el principio *Composite Reuse* (CRP) [18], que favorece la composición frente a la herencia. Este diseño aumentó la flexibilidad de nuestra aplicación CHEW, tanto al encapsular los vouchers (la familia de algoritmos) como al permitir a los usuarios cambiar los vouchers que quieran utilizar en tiempo de ejecución.

Como se explica en el apartado 5.1, los usuarios pueden gastar todos los vales en un solo viaje de compras o dividirlos entre distintos viajes. Para maximizar la flexibilidad, aplicamos el patrón *Factory Method* [17] para permitir a los usuarios de la aplicación CHEW seleccionar los vales que quieren utilizar en tiempo de ejecución. Este patrón encapsula la creación de los tipos de vales dejando que las subclases decidan qué vales crear. Aplicamos este patrón para crear una clase abstracta *VoucherFactory* y las dos clases concretas que heredan de ella, *CashVoucherFactory* y *RegularVoucherFactory*, que crean los tipos específicos de vales.

También utilizamos muchos marcos de trabajo de Android en nuestra aplicación CHEW que aprovechan los patrones de software para ayudar a la reutilización del código y facilitar el desarrollo de la aplicación encapsulando los detalles tediosos y propensos a errores de los desarrolladores de aplicaciones. Por ejemplo, nuestra aplicación utiliza *AsyncQueryHandler* [4] para ejecutar las operaciones del proveedor de contenidos de forma asíncrona sin bloquear la interfaz de usuario, ya que *AsyncQueryHandler* proporciona métodos de devolución de llamada que se ejecutan tras la finalización de las operaciones del proveedor de contenidos.

AsyncQueryHandler se implementa utilizando los patrones *Proactor* y *Asynchronous Completion Token* [20] (Figura 12). Utiliza el patrón *Proactor* para simplificar el desarrollo de aplicaciones asíncronas, dividiendo la funcionalidad de una aplicación en operaciones asíncronas (como consultas a la base de datos) y manejadores de finalización que utilizan los resultados de las operaciones asíncronas para implementar la lógica de negocio de la aplicación. Asimismo, utiliza el patrón *Asynchronous Completion Token* para permitir que una aplicación demultipleje y procese eficazmente las respuestas de las operaciones asíncronas. Al aplicar estos dos patrones juntos, *AsyncQueryHandler* permite desacoplar los mecanismos de asincronía independientes de la aplicación de la funcionalidad específica de la aplicación, y simplificar los algoritmos de gestión de eventos.

Nuestra aplicación CHEW también utiliza el marco Android *AsyncTask* [4] para simplificar la creación de tareas de larga duración, como la consulta de



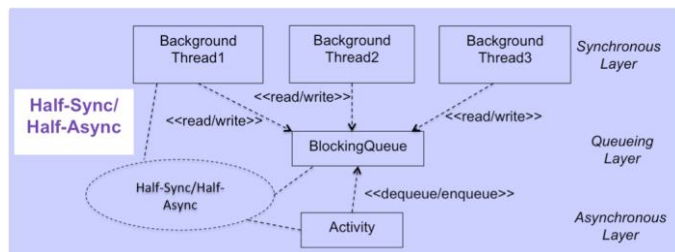


Figura 13. Medio-sincronización/medio-sincronización.

la base de datos, sin necesidad de comunicarse con Threads y Handlers. El marco de trabajo *AsyncTask* aprovecha el patrón *Half-Sync/Half-Async* [20,21] para integrar las operaciones síncronas y asíncronas en un hombre eficiente y bien estructurado mediante el desacoplamiento de estas operaciones para simplificar la programación concurrente mientras se apoya la eficiencia de la ejecución (Figura 13). Descomponiendo el sistema general en tres capas (es decir, síncrona, asíncrona y de cola), *AsyncTask*

centraliza la comunicación entre capas porque la capa de cola media todas las interacciones; y las políticas de sincronización en cada capa están desacopladas de manera que cada capa puede utilizar diferentes estrategias de concurrencia. Utilizamos el marco *IntentService* de Android para extraer datos de un servidor remoto de Open mHealth a la base de datos SQLite de la aplicación en su primera ejecución. El marco *IntentService* implementa el patrón

Command Processor [19] para encapsular la solicitud de un servicio, la solicitud de descarga de datos en nuestro caso, como un objeto que se pasa a un Servicio para que lo ejecute (Figura 14).

El uso del *procesador de comandos* garantiza que el hilo del cliente no se bloquee durante el procesamiento de los comandos, y que diferentes usuarios puedan comunicarse con un servicio de diferentes maneras mediante comandos. Así, el mismo código puede ser reutilizado para múltiples propósitos.

Resultado. La aplicación de patrones de software y los marcos de trabajo de Android que aprovechan los patrones de software mejoran la capacidad de mantenimiento y la flexibilidad de nuestra aplicación CHEW. Los marcos de trabajo de Android también nos permitieron centrarnos en los requisitos de la aplicación y minimizar el tiempo de desarrollo al abstraer muchos detalles de implementación de bajo nivel. Además, aumentaron la reutilización del código, ya que podíamos ampliar sus comportamientos de forma sencilla para satisfacer las diferentes necesidades de la aplicación.

6. Trabajos relacionados

Esta sección compara nuestro trabajo en la aplicación CHEW con los esfuerzos de investigación y desarrollo relacionados. La aplicación WIC de San Diego para Android [22] e iOS [23] notifica a los usuarios actualizaciones sobre las actividades, promociones y eventos de WIC. También proporciona información sobre alimentación saludable y recetas elaboradas con alimentos aprobados por WIC.

La aplicación WIC Calculators Android [24] e iOS [25] está destinada a las agencias de WIC, pero no a los participantes reales, para ayudar a determinar las cantidades de fórmula infantil, identificar los productos

uctos que cumplen los requisitos mínimos del pliego de condiciones para los productos de grano y trigo enteros.

La aplicación EBT Shopper para Android e iOS [26] simplifica la experiencia de compra de los participantes en el programa WIC al permitirles determinar los artículos elegibles para el programa WIC a medida que escanean los productos en una tienda, y al enumerar los artículos que se pueden comprar en función del paquete de alimentos del programa WIC. Se utiliza en los estados con prestaciones electrónicas de WIC (no con vales de papel). La documentación disponible no aclara cómo esta aplicación realiza un seguimiento de las selecciones de cada persona y de los límites de los vales WIC.

Las aplicaciones mencionadas anteriormente sólo proporcionan un apoyo limitado a los participantes de WIC en comparación con la aplicación que estamos creando. Sirven como herramienta de compra (la aplicación EBT Shopper) o como herramienta de educación nutricional (la aplicación WIC de San Diego), o ni siquiera son utilizadas directamente por los participantes de WIC (la aplicación WIC Calculators). Nuestra aplicación intenta ofrecer los mayores beneficios a los participantes de WIC sirviendo tanto como herramienta de compra como de educación nutricional, simplificando y automatizando la antes engorrosa experiencia de compra de los participantes y ayudándoles a tener estilos de vida más saludables.

7. Observaciones finales

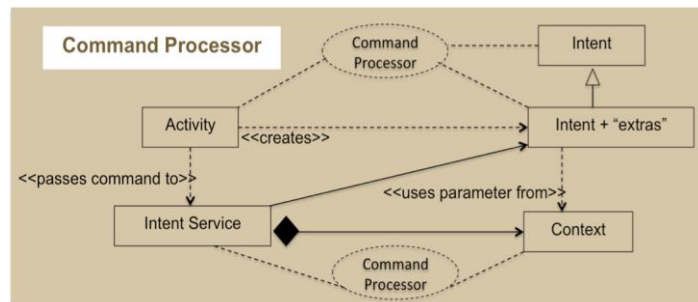
Este documento describe la aplicación CHEW, que se implementa en las plataformas Android y Open mHealth para simplificar la experiencia de compra de los participantes en el programa WIC de Tennessee y proporcionar educación nutricional. Los participantes en el programa WIC de Tennessee utilizan actualmente vales de papel para obtener alimentos nutritivos suplementarios. Estos tipos de vales son difíciles de utilizar para los participantes, ya que deben llevar un registro manual de los productos que obtienen y de las cantidades de cada uno, lo que resulta tedioso y propenso a errores.

Muchos de los participantes en el programa WIC también tienen problemas con los números, por lo que a menudo no utilizan sus vales de valor en efectivo para comprar productos, ya que no pueden calcular y seguir fácilmente los precios en la tienda. La educación nutricional que se imparte entre las citas del programa WIC tiene por objeto fomentar y apoyar la elección de un estilo de vida saludable.

Nuestra aplicación CHEW ayuda a aliviar los problemas con los vales de papel, ya que permite a los participantes escanear los productos en la tienda e identificar automáticamente si pueden obtener un producto en función de su receta de vales. La aplicación también hace un seguimiento de las cantidades y tamaños permitidos y ofrece una calculadora fácil de usar de los precios de los productos para ayudar a los participantes a utilizar sus vales de valor en efectivo al máximo. La aplicación CHEW también sirve como herramienta de educación nutricional, ya que ofrece consejos saludables y atractivas recetas de aperitivos.

Nuestra experiencia en el desarrollo y la aplicación de la aplicación CHEW hasta el momento ha dado lugar a las siguientes lecciones aprendidas:

- **Mantener la coherencia entre las bases de datos manualmente es tedioso y propenso a errores.** Tuvimos que editar manualmente los datos tablas base recibidas de las tiendas de comestibles participantes para vincularlas con las tablas que almacenan las descripciones de los vales (pusimos manualmente los identificadores de los distintos tipos de alimentos y grupos de vales). Queremos automatizar este proceso en futuros trabajos, por ejemplo, empleando técnicas de aprendizaje automático para asentar los valores de una columna de una tabla en función de los datos observados en otras columnas. Tenemos previsto estudiar la viabilidad de esta solución y evaluar su capacidad para producir resultados precisos.
- **Se necesitan soluciones de análisis de datos para ofrecer**



consejos personalizados para cada usuario. Nuestra actual aplicación CHEW ofrece consejos generales sobre salud a cada usuario. Esta función puede mejorarse para aumentar la educación nutricional mediante la personalización de los consejos en función de las elecciones de compra y el origen étnico del usuario. Por ejemplo, se podría animar a los participantes a comprar más manzanas y zanahorias para recibir una cantidad suficiente de fibra dietética que mejore la salud digestiva y prolongue la sensación de saciedad. Trabajando con nuestras partes interesadas, también aprendimos que las etnias

El origen étnico desempeña un papel importante a la hora de decidir qué alimentos deben evitarse para mantenerse sanos. Por lo tanto, nuestro trabajo futuro desarrollará soluciones de análisis de datos para analizar el historial de compras del usuario, teniendo en cuenta su origen étnico, para proporcionar consejos adaptados a cada usuario, lo que mejorará aún más la educación nutricional.

- **Evaluación del impacto de la aplicación CHEW en la mejora de la educación nutricional.** Como se describe en la sección 5.3, hemos integrado la plataforma Open mHealth para proporcionar medios eficaces de visualización de los datos de compra de los participantes para ayudar a los estudios de nutrición a evaluar el impacto de la aplicación en la mejora de la educación nutricional. Aunque hemos dedicado mucho tiempo a comprender los componentes de Open mHealth y a integrarlos en nuestra solución, queda mucho trabajo por hacer para estudiar los distintos componentes de Open mHealth, como las unidades de visualización de datos (DVU), para ayudar a los estudios de nutrición a evaluar los datos recopilados de forma más eficiente.
- **Aprovechar la Realidad Aumentada móvil para simplificar aún más la experiencia de compra de los participantes.** La realidad aumentada móvil es un nuevo enfoque para visualizar información cibernética sobre imágenes físicas [27]. En el trabajo futuro tenemos previsto apoyar los vales de WIC para grupos de usuarios especiales que consisten en participantes con alergias alimentarias, intolerancia a la lactosa e intolerancia al gluten. Dado que estos participantes tienen opciones más limitadas de los artículos que los participantes de categorías no especiales, es posible que no encuentren fácilmente los productos aprobados en la tienda. Por lo tanto, planeamos utilizar la Realidad Aumentada móvil para ayudar a los participantes con categorías especiales de vales a localizar los artículos aprobados más rápidamente en la tienda de comestibles [27,28]. Por ejemplo, los participantes apuntarían su teléfono a la estantería de la tienda y la aplicación reconocería automáticamente todos los productos sin gluten y los marcaría para el usuario.
- **Apoyo a la detección automática de tiendas.** Actualmente, nuestra aplicación CHEW pide a los usuarios que seleccionen manualmente la tienda de comestibles en la que piensan comprar. En el futuro tenemos previsto mejorar la aplicación para que detecte automáticamente la tienda adecuada en función de su ubicación. La aplicación podría almacenar las ubicaciones de las tiendas participantes y detectar una ubicación para identificar una tienda concreta cuando el usuario vaya a comprar. Sin embargo, esta capacidad requeriría una conexión a la red, que no podemos apoyar de forma consistente en el momento actual. Sin embargo, a medida que más participantes en el programa WIC tengan acceso a Internet, esta función será más útil.
- **Apoyo a múltiples plataformas.** Actualmente estamos desarrollando la aplicación CHEW para la plataforma Android. Esta elección plantea limitaciones porque no podemos dar soporte a los participantes que utilizan otras plataformas operativas móviles. Por lo tanto, en el futuro estamos planeando crear una aplicación híbrida que incorpore HTML5 dentro de un contenedor nativo delgado para combinar elementos de aplicaciones nativas y HTML5 (y así permitir la conectividad en línea y fuera de línea) para soportar múltiples plataformas operativas móviles.

Agradecimientos

Este proyecto fue apoyado por la Iniciativa de Agricultura y Alimentación #2011-68001-30113, del Instituto Nacional de Alimentación y Agricultura del USDA, programa de Investigación Integrada, Educación y Extensión para Prevenir la Obesidad Infantil - USDA-NIFA- AFRI-003327.

Referencias

1. USDA: Departamento de Agricultura de los Estados Unidos. Servicio de Alimentación y Nutrición: Mujeres, bebés y niños (WIC). www.fns.usda.gov/wic/about-wic-wic-glance. Consultado en enero de 2014.

2. Departamento de Salud: About WIC.
<http://health.state.tn.us/wic/>. Consultado en enero de 2014.
3. Meier, R. *Professional Android 4 Application Development*. John Wiley & Sons, 2012.
4. Murphy, M. L. *The Busy Coder's Guide to Android Development*. CommonsWare, 2013. commonsware.com/Android/.
5. Open mHealth. openmhealth.org. Consultado en enero de 2014.
6. MongoDB. www.mongodb.org. Consultado en enero de 2014.
7. SQLite. www.sqlite.org. Consultado en enero de 2014.
8. zxing: Biblioteca de procesamiento de imágenes de códigos de barras 1D/2D multiformato con clientes para Android, Java. code.google.com/p/zxing. Consultado en enero de 2014.
9. Android-wheel: Android Picker widget.
code.google.com/p/android-wheel/. Consultado en enero de 2014.
10. Kreibich, J. A. *Uso de SQLite*. O'Reilly Media, 2010.
11. Soporte de Microsoft: Descripción de los fundamentos de la normalización de bases de datos.
support.microsoft.com/kb/283878. Consultado en enero de 2014.
12. Desarrolladores de Google: YouTube Android Player API.
developers.google.com/youtube/android/player. Consultado en enero de 2014.
13. Chen, C., Haddad, D., Selsky J., Hoffman, J. E., Kravitz, R. I., Estrin, D. E., y Sim, I. Making Sense of Mobile Health Data: Una arquitectura abierta para mejorar la salud a nivel individual y poblacional. *Journal of Medical Internet Research* 14, 4 (2012), p10. www.jmir.org/2012/4/e112. Consultado en enero de 2014.
14. Open mHealth: Developers. openmhealth.org/developers. Consultado en Jan 2014.
15. Amazon Web Service. aws.amazon.com/. Consultado en enero de 2014.
16. Coplien, J., Hoffman, D. y Weiss, D. Commonality and Variability in Software Engineering. *IEEE Software* 15, 6 (1998), 37-45.
17. Gamma, E., Helm, R., Johnson, R. y Vlissides, J. *Design Patterns: Elements of Reusable Object Oriented Software*. Addison-Wesley, 1994.
18. Knoernschild, K. *Diseño en Java: Objects, UML, and Process*. Addison-Wesley, 2001.
19. Buschmann, F., Meunier, R., Rohert, H., Sommerlad, P. y Stal, M. *Pattern-Oriented Software Architecture Volume 1: A System of Patterns*. John Wiley & Sons, 1996.
20. Schmidt, D., Stal, M., Rohnet, H. y Buschmann F. *Pattern-Oriented Software Architecture Volume 2: Patterns for Concurrent and Networked Objects*. John Wiley & Sons, 2000.
21. Schmidt, D. C. y Cranor, C. D. Half-Sync/Half-Async: An Architectural Pattern for Efficient and Well-structured Concurrent I/O. *Proc. 2nd Pattern Languages of Programs '95*.
22. Universidad Estatal de San Diego: California WIC. SDSU WIC Android App Screenshots. www.sdsuwic.org/wic-program/wic-san-diego-android-app-now-available.html. Consultado en enero de 2014.
23. Universidad Estatal de San Diego: California WIC. Install WIC San Diego iPhone App. sdsuwic.org/wic-program/wic-san-diego-iphone-app-coming-soon.html. Consultado en enero de 2014.
24. Google play. WIC Calculators.
play.google.com/store/apps/details?id=com.bluepanestudio.FormulaAndWholeGrainCalculator&hl=en. Consultado en enero de 2014.
25. Vista previa de iTunes. WIC Calculators.
itunes.apple.com/us/app/wic-calculators/id501212303?mt=8. Consultado en enero de 2014.
26. EBT Shopper: Compras WIC, simplificado.
<http://ebtshopper.com>. Consultado en enero de 2014.
27. Jules White, Douglas C. Schmidt y Mani Golparvar-Fard, "Applications of Augmented Reality", *IEEE Proceedings Special issue on Applications of Augmented Reality*, 2014 (por aparecer).
28. Investigación de IBM. La realidad aumentada hace que las compras sean más personales: Una nueva aplicación móvil de

IBM Research ayuda tanto a los consumidores como a los minoristas. www.research.ibm.com/articles/augmented-reality.shtml. Consultado en enero de 2014.