KemapMobile: una adaptación de la herramienta software KeMap a la plataforma Android

Autor: Diego Acaymo GonzálezArroyo
Director del proyecto: David Bañeres Besora
TFM de investigación – Máster Universitario de Software Libre
Universitat Oberta de Catalunya
dgonzalezarro@uoc.edu

Resumen

Este trabajo describe el proceso de creación de la herramienta software KemapMobile para la plataforma Android. El artefacto es una adaptación del software de escritorio KeMap que se integra en la plataforma educativa VerilUOC y permite a los estudiantes conectarse a ella para descargar, en la aplicación, ejercicios con los que poner a prueba y afianzar sus conocimientos sobre la simplificación de funciones lógicas booleanas con mapas de Karnaugh y tablas de verdad. La arquitectura cliente-servidor de la aplicación permite a los estudiantes disponer de ejercicios personalizados, y al profesorado obtener datos sobre el uso de la herramienta y sobre la progresión del alumnado.

1. Introducción

La necesidad de generar de una forma sencilla y rápida soluciones cercanas a la mínima para resolver ecuaciones booleanas con las que poder simplificar circuitos lógicos combinacionales dio pie a la aparición de diversas técnicas que solventan el problema sin usar los postulados y teoremas del álgebra de Boole, que tiene como gran inconveniente no conocer a priori el orden de aplicación de los teoremas que se deben utilizar para obtener la solución mínima.

Una de estas técnicas la describió M. Karnaugh en 1953, tomando como base para desarrollar su idea un trabajo anterior de E.W. Veicht, y cuya creación justificó argumentando que con la nueva técnica se evita el inconveniente de la lentitud en la búsqueda de la solución que deriva de desconocer el orden en que se deben aplicar las reglas del álgebra de Boole en este tipo de problemas, y se simplifica la dificultad de la resolución por no ser necesario usar dichas reglas[1].

El método de Karnaugh consiste en trasladar los datos de una función lógica o una tabla de verdad que representa el diseño de un circuito lógico combinacional a una matriz o mapa, y extraer los términos mínimos que en forma de suma de productos o en forma de producto de sumas conformarán la solución mínima. Karnaugh detalla en su texto una serie de reglas que permiten hacer agrupaciones de celdas adyacentes (adyacencia vertical y horizontal) de acuerdo con su contenido y extraer los términos mínimos con la observación y el reconocimiento de patrones geométricos que generan las agrupaciones de celdas sobre la matriz.

La razón del trabajo de investigación actual es solventar un problema cuya pregunta es si es posible crear un producto computacional que se ejecute en dispositivos móviles basados en la plataforma Android y que sirva como herramienta de aprendizaje para la comprensión de la técnica que Karnaugh ideó y de otros conceptos asociados a la síntesis de circuitos lógicos combinacionales, como son las tablas de verdad, los términos mínimos y las expresiones simplificadas en forma de suma de productos. El creciente uso de teléfonos móviles y tabletas en nuestra sociedad y las evaluaciones con resultado positivo hechas sobre el impacto que tuvo en los estudiantes el uso de otras herramientas software educativas con funcionalidades similares[2-6] justifica, en opinión del autor, el desarrollo de este trabajo.

Este documento se estructura en los siguientes capítulos:

1. Introducción
2. Contribución
3. Estado del arte
   3.1. Conclusiones de la revisión

4. Metodología de la investigación
5. Implementación de KemapMobile
   5.1. Android: paradigma de desarrollo
   5.2. Desarrollo del artefacto
   5.3. Contratiempos durante la implementación
6. Discusión de los resultados
7. Conclusiones
8. Referencias
9. Publicación del artículo

2. Contribución

El artefacto que resulta de esta investigación es una instanciación que sirve como prueba de concepto para mostrar la interacción con los métodos y modelos que se implementan en la plataforma educativa VerilUOC[6] de la Universitat Oberta de Catalunya, al adaptar la herramienta software ya existente KeMap para ejecutarla en dispositivos móviles compatibles con la plataforma Android.

KeMap es una herramienta software de escritorio que se ejecuta en máquinas virtuales Java compatibles con la versión 1.6 del JRE de Oracle y en función del usuario que se valida en la aplicación, contra la infraestructura de la Universitat Oberta de Catalunya, descarga de la plataforma VerilUOC un conjunto de ejercicios para que los alumnos puedan practicar y afianzar sus conocimientos sobre el método de los mapas de Karnaugh y de los conceptos asociados. Estos ejercicios disponen de una descripción del problema que se debe resolver, y permiten que se verifique la solución del alumno según las opciones que se escogen entre las permitidas para cada ejercicio.

La aplicación es capaz de comprobar la validez de las modificaciones en la tabla de verdad y en el mapa de Karnaugh, así como validar las agrupaciones de celdas en el mapa y la expresión simplificada en forma de suma de productos. Después de comprobar la solución que se introdujo, KeMap indica si las respuestas fueron correctas para cada opción de verificación que se escogió, y en caso de no serlo muestra pistas acerca de en que difiere la solución introducida de la solución correcta. Para cada ejercicio, además de la descripción del problema, la aplicación muestra un registro con el número de veces que se resolvió y el número de intentos que fueron correctos. También permite guardar el estado de resolución del ejercicio en el disco y así continuar con el problema en otra ocasión.

La adaptación de esta herramienta software a entornos móviles como resultado de este trabajo de investigación se llama KemapMobile, y entiende el autor que llena un hueco existente en la actualidad, pues si bien existen productos con funcionalidades parecidas para entornos de escritorio, no se ha encontrado ninguno que sea compatible con la plataforma Android y que tenga similares características con respecto a las funcionalidades o al tipo de arquitectura de KemapMobile. La arquitectura cliente-servidor en la que se basa el producto permite al profesorado recabar datos sobre su uso y sobre el éxito de los alumnos al resolver los ejercicios, y también permite modificar ejercicios existentes o añadir otros nuevos y que los cambios se reflejen en el terminal móvil del usuario que ejecuta la aplicación en el momento que se publique la modificación o adición del ejercicio.

3. Estado del arte

Esta revisión del estado del arte tiene como objetivo describir los avances más significativos sobre los artefactos que tienen alguna relación con el método descrito por M. Karnaugh[1], una de las técnicas de simplificación de mayor uso para trabajar con funciones lógicas que cuenten con un número de variables no superior a seis. El método de los mapas de Karnaugh cuenta con una serie de reglas que permiten agrupar (mediante el reconocimiento de patrones geométricos) celdas en una matriz que contiene datos provenientes de una función lógica o una tabla de verdad, y que son una forma de representar un circuito lógico combinacional.

A continuación se describen las características más significativas de cada artefacto que se analizó durante el desarrollo de la investigación:

WinLogiLab → El artículo de investigación escrito por Charles Hacker y Renate Sitte dio lugar al desarrollo, por parte del primero de los autores, de una instanciación del artefacto descrito en el documento[3] para la Universidad de Griffith, y tiene una licencia privativa que permite de forma gratuita su uso con fines académicos. La aplicación es compatible con sistemas operativos Microsoft Windows posteriores a la versión 95, y no fue posible averiguar el lenguaje de programación con el que se desarrolló por no aparecer en el artículo ni en la página web de la herramienta.

WinLogiLab contiene un conjunto de ocho módulos con diversas funcionalidades en cada uno de ellos, y de los que sólo están relacionados con el tema de investigación sobre simplificación de funciones lógicas con mapas de Karnaugh, y son BoolTut y WinBoolean. El módulo BoolTut provee al alumno de un sistema de tutoría inteligente con el uso de una guía que describe paso a paso el procedimiento de resolución de un mapa de Karnaugh o del algoritmo de Quine-McCluskey a

partir de los datos existentes en una tabla de verdad. Este módulo también cuenta con un juego en el que se le presenta al usuario un mapa de Karnaugh para que este agrupe las celdas necesarias y el programa pueda comprobar la corrección del ejercicio, además de mostrar en la misma ventana otro mapa con la solución correcta.

El segundo módulo objeto de estudio es WinBoolean, cuya creación responde a la idea de mostrar las relaciones entre los circuitos lógicos combinacionales, el álgebra booleana y las tablas de verdad. Este módulo permite definir los datos iniciales con cualquiera de las tres formas de representación descrita y obtener la solución mínima con el método de los mapas de Karnaugh, para lo que muestra junto a la solución, la tabla de verdad y el mapa con la agrupación de celdas correcta, y en otra pestaña de la misma ventana aparece la representación de la solución mínima en forma de diagrama de circuitos combinacionales.

Karmin → El artefacto descrito en el documento [4] es el resultado de un trabajo de investigación sobre enlaces conceptuales hecho por Raúl I. Jáuregui y otros para la Universidad Nacional de la Patagonia "San Juan Bosco". El programa está escrito en el lenguaje de programación C# y se ejecuta sobre sistemas operativos Microsoft Windows que dispongan de la plataforma .NET a partir de la versión 2.

Al contrario que muchas de las herramientas objeto de esta evaluación, Karmin cuenta únicamente con dos funcionalidades básicas. La primera permite al usuario rellenar los datos de una tabla de verdad para que el programa muestre la función lógica original y calcule la simplificación, mientras que en la segunda el usuario puede rellenar un mapa de Karnaugh y el software calcula la función original y el resultado simplificado, la desventaja es que no muestra la agrupación de las celdas en el mapa, con lo que la utilidad de la herramienta se limita a calcular los resultados a partir de esos dos únicos métodos de entrada de datos.

Uno de los puntos fuertes de Karmin es la sencillez de uso, y que cuenta con una interfaz gráfica cuidada y fácil de adaptar a terminales móviles de pantalla reducida, además de contar con un servicio de ayuda general con información acerca de la instalación y del funcionamiento del programa.

Karma → Esta aplicación descrita por C.E. Klock en el documento[7] y desarrollada por F. R. Schneider y otros en el Instituto de Informática de la Universidad Federal de Rio Grande do Sul tiene una licencia GPL que la convierte en software libre, se implementó con el lenguaje de programación Java y se ejecuta como un applet en navegadores web o como aplicación de escritorio sobre sistemas operativos que dispongan de una máquina virtual Java compatible.

Karma dispone de una interfaz gráfica atractiva y fácil de adaptar a las pantallas reducidas de los dispositivos móviles por su diseño compacto y modular, que cuenta con nueve módulos de los que sólo dos tienen una relación directa con la técnica de los mapas de Karnaugh. El primero consiste en un sistema de aprendizaje que combina el uso de juegos con la tutoría inteligente, donde los juegos son de preguntas cortas que el usuario debe responder para que el software decida si lo ha hecho correctamente, y los módulos de tutoría inteligente muestran los resultados según los datos que introduzca el usuario, guiándolo durante todo el proceso. Este sistema de aprendizaje cuenta con ejercicios para practicar conceptos relacionados con la agrupación de celdas en un mapa de Karnaugh.

El segundo módulo del programa que guarda relación con el tema de investigación actual muestra de forma conjunta una tabla de verdad y un mapa de Karnaugh asociado a esta, en la que ambas formas de representar los datos son editables por el usuario y donde se reflejan los cambios al modificar cualquiera de las dos, con lo que aporta valor al fin pedagógico de la herramienta. Los datos del mapa de Karnaugh y de la tabla de verdad se pueden crear de forma manual por el usuario, y también se pueden cargar desde plantillas contenidas en la aplicación para que esta devuelva la expresión simplificada, aunque también es capaz de reducir expresiones lógicas que se introducen en forma de función. El software hace los cálculos con el algoritmo de Quine-McCluskey y al finalizar el proceso muestra los implicantes seleccionados y los pasos del algoritmo. Este módulo cuenta con un servicio de ayuda básico que describe su funcionamiento.

Developing mobile learning applications for electrical engineering courses → Los autores Josh Potts, Nick Moore y Somsak Sukittanon presentan en el texto[6] el diseño y creación de un artefacto para la Universidad de Tennesse con la que ayudar a los estudiantes en la adquisición y comprensión de conceptos relacionados con la lógica digital y el procesamiento de señales. En el documento publicado se indica que el software se encuentra disponible para su descarga desde el repositorio de aplicaciones para la plataforma Android de Google, pero no se pudo encontrar y por tanto tampoco se pudo evaluar la instanciación del artefacto. En la publicación de conferencia[2] se explica que el programa es compatible con los sistemas operativos Google Android y Apple iOS, donde la versión para la primera plataforma se desarrolló con el lenguaje de

programación Java y la versión para iOS se creó con el lenguaje de programación Objective-C.

La aplicación tiene dos módulos diferentes, uno se orienta hacia el campo de la lógica digital y el otro hacia el procesamiento de señales (este último no guarda relación con el trabajo de investigación actual). El primer módulo tiene seis funcionalidades diferentes, de las que sólo dos tienen una relación directa con el tema de estudio. La primera de estas funcionalidades sirve para evaluar la salida que genera un circuito combinacional de acuerdo con los datos de entrada, y comprobar si los valores de salida introducidos por el usuario son correctos. La otra funcionalidad interesante con que cuenta la aplicación muestra a los alumnos un mapa de Karnaugh para que estos, a partir de los datos que contiene el mapa, puedan encontrar la solución mínima. Una limitación de la aplicación es que al comprobar el resultado no muestra en el mapa de Karnaugh las agrupaciones de celdas.

A game based learning content for tutoring in simplifying boolean functions → La instanciación del artefacto que creó Yong Suk Choi para la Universidad Hanyang de Seúl como resultado de la investigación descrita en el artículo[5], no se pudo obtener para evaluarla por no aparecer en el documento ninguna dirección desde la que descargarlo ni ninguna referencia acerca del nombre del producto que pudiera usarse para encontrarlo en Internet. En el texto tampoco se hace mención a la licencia con que cuenta el programa ni a las plataformas de ejecución compatibles.

El artículo indica que la herramienta combina dos modelos de aprendizaje, que son la tutoría inteligente y la enseñanza basada en juegos para motivar al estudiante y facilitar su comprensión sobre la técnica de los mapas de Karnaugh. En el juego el autor hace una analogía entre una red de pesca y un mapa de Karnaugh para que el alumno practique las reglas de agrupación de celdas, mientras que la tutoría inteligente se implementa con un mapa de Karnaugh en el que el alumno debe agrupar las celdas de acuerdo al contenido de estas; cuando el usuario crea un grupo que no es correcto, la aplicación muestra comentarios o indicaciones en forma de pistas que sirvan de guía para completar el ejercicio.

Una de las diferencias con otras herramientas evaluadas que usan la tutoría inteligente, la enseñanza basada en juegos o una combinación de las dos, es el diseño de la interfaz gráfica de la aplicación, que se asemeja a un videojuego clásico con varios personajes (avatares) y un escenario colorido que muestra un temporizador y un contador de puntos. La mayor limitación del software es la escasez de funcionalidades, ya que no trata conceptos estrechamente relacionados con los mapas de Karnaugh como son las tablas de verdad o las funciones lógicas, y se centra exclusivamente en afianzar conceptos sobre la agrupación de celdas.

Boole-Deusto → La herramienta creada por Javier García Zubía, Jesús Sanz Martínez y Borja Sotomayor Basilio en el departamento de arquitectura de computadores de la universidad de Deusto fue creada con el lenguaje de programación C++, y cuenta con una licencia privativa cuyo uso y distribución es libre si se hace con fines personales o educativos.

Boole-Deusto se creó para trabajar sobre los sistemas operativos de la familia Microsoft Windows, sin que existan en la actualidad versiones conocidas para los sistemas operativos de plataformas móviles, y que cuenta con una interfaz gráfica poco atractiva que carece de un servicio de ayuda o instrucciones de uso para las diferentes funcionalidades con las que cuenta el programa. Esta aplicación se divide en dos secciones, en la que la primera se orienta hacia el análisis y el diseño de circuitos combinacionales, mientras que la segunda lo hace hacia el análisis y el diseño de circuitos secuenciales. La parte relacionada con los circuitos secuenciales queda fuera de la materia de investigación por lo que sólo se hace referencia a la primera sección de la herramienta.

Entre las funcionalidades más destacadas en el apartado de lógica combinacional se encuentra la posibilidad de que el usuario pueda definir circuitos combinacionales con la creación de expresiones booleanas y con el uso de tablas de verdad, formas canónicas y mapas de Karnaugh. Una característica muy útil de la aplicación es que al definir un circuito combinacional con alguno de los métodos descritos, los datos se reflejan de forma automática en los componentes de los demás métodos, si bien tiene la gran desventaja de que el diseño de la herramienta no cuenta con una interfaz gráfica que permita ver al mismo tiempo la representación de los datos para los diferentes métodos. Con los datos introducidos Boole-Deusto es capaz de obtener la solución mínima, aunque los autores no indican en el documento analizado[8] la técnica de simplificación de funciones lógicas con la que se calcula la expresión mínima, sí indican que se trata de un método no heurístico, por lo que siempre devuelve la solución óptima en forma de suma de productos o en forma de producto de sumas, con la que el programa puede generar el diagrama del circuito combinacional resultante de la simplificación.

Otra funcionalidad muy interesante con la que cuenta la aplicación es el “modo aprendizaje”, que consta de dos pruebas: una en la que el usuario agrupa

las celdas de un mapa de Karnaugh y el software comprueba si la agrupación es correcta, y otra en la que el usuario escribe una suma de productos o un producto de sumas y el software comprueba la validez de la solución.

KVD → Karnaugh-Veicht Diagram (KVD) es una herramienta[9] creada por Martin Hoffmann en 2011, de la que el autor desconoce si es el resultado de un trabajo de investigación académica por no poder encontrar documentación que lo acredite, permite ver en un mapa de Karnaugh la representación de una expresión booleana y simplificarla con el algoritmo de Quine-McCluskey.

El uso del programa es muy sencillo y dispone de una interfaz gráfica muy simple y amigable donde aparece una mapa de Karnaugh en la que el usuario introduce los datos que representan la función lógica que quiere simplificar. Cuando el usuario modifica el valor de una celda en el mapa, la aplicación comprueba si es posible realizar alguna agrupación, y en caso afirmativo la resalta y calcula el resultado de la simplificación de acuerdo con el contenido del mapa.

La característica más atractiva de esta aplicación y el motivo por el que se evaluó es que se ejecuta sobre la plataforma Android de Google y es de libre descarga y uso, si bien no se especifica la licencia del programa en el repositorio de aplicaciones para Android de Google ni en la página web del autor del software.

MiniKarnaugh → La aplicación desarrollada por Serban Stoenescu, de la que se desconoce si es el resultado de un proyecto de investigación, tiene como principal atractivo que es de uso libre, se ejecuta sobre la plataforma Android de Google y permite simplificar funciones lógicas. Este software[10] cuenta con una única ventana donde el usuario puede representar un circuito combinacional en forma de expresión lógica, tabla de verdad o mapa de Karnaugh para que la aplicación devuelva el resultado de la simplificación, con la desventaja de que no muestra las agrupaciones de celdas en el mapa y no informa de la técnica de simplificación que utiliza. Otra limitación de esta herramienta es el diseño poco elaborado de su interfaz gráfica de usuario.

3.1. Conclusiones de la revisión

Durante esta revisión del estado del arte se observó que las instanciaciones de artefactos que resultan de los artículos que se escogieron tienen un fin pedagógico, y para facilitar el aprendizaje del estudiante incorporan modelos de enseñanza con juegos de preguntas cortas o retos y una tutoría inteligente que usa un sistema de guía con pistas. Una funcionalidad importante desde el punto de vista pedagógico con la que no todas las herramientas cuentan es mostrar la agrupación de celdas adyacentes en un mapa de Karnaugh al simplificar los datos que este contiene, aunque es cierto que varias aplicaciones hacen hincapié en practicar las reglas de agrupación y tienen módulos específicos para que los alumnos las aprendan o afiancen sus conocimientos sobre la cuestión. En los documentos que describen los artefactos evaluados que cuentan con la funcionalidad descrita no se hace mención a la técnica que usan para agrupar las celdas del mapa, por

Figura 1. Tabla comparativa de las características

ARTEFACTO	Valida las agrupaciones de celdas	Valida las tablas de verdad	Valida las expresiones simplificadas	Ejercicios con enunciado	Arquitectura cliente servidor	Compatible plataforma Android
WinLogiLab	o	-	-	o	-	-
Karmin	-	-	-	-	-	-
Karma	o	-	-	o	-	-
Developing*	o	-	o	o	-	o
A game based**	o	-	-	o	-	-
Boole-Deusto	o	-	o	o	-	-
K V D	-	-	-	-	-	o
MiniKarnaugh	-	-	-	-	-	o
KemapMobile	o	o	o	o	o	o

* Developing mobile learning applications for electrical engineering courses

** A game based learning content for tutoring in simplifying boolean functions

lo que fue interesante la lectura de un documento sobre sistemas expertos[11] que usa un conjunto de reglas que permite comprobar si dos celdas de un mapa son adyacentes entre sí, para que según el contenido de las mismas se pueda decidir si forman un grupo.

Una amplia mayoría del software que resulta de los documentos académicos que se evaluaron se diseñaron para ejecutarse sobre sistemas operativos de la familia Microsoft Windows, así que para revisar el estado del arte sobre el tema de investigación en la plataforma Android de Google fue necesario contar con aplicaciones de las que, en el momento de escribir este texto, se desconoce si son el resultado de una investigación académica. Para elaborar la revisión del estado del arte sobre el tema de actual, se evaluó un conjunto de herramientas que en su mayoría son instanciaciones de artefactos que aparecieron como resultado de proyectos de investigación que siguieron una estrategia de diseño y creación, y que guardan relación con la simplificación de funciones lógicas mediante el uso del método de los mapas de Karnaugh. La figura 1 muestra una comparativa de algunas características de las aplicaciones que se reseñaron en este estudio del estado del arte.

4. Metodología de la investigación

Para desarrollar la investigación se siguió una estrategia de diseño y creación a partir de la que se generó un artefacto en forma de instanciación que se describe en esta sección. Los datos que se recabaron y analizaron para hacer el trabajo se obtuvieron por observación directa del funcionamiento de las herramientas software similares existentes y con la revisión de los documentos que se indican en la sección de referencias y otros que no se añadieron por no tener entidad suficiente para incluirlos en esa sección.

Cada documento que se revisó fue considerado como una entidad propia que analizar, de forma cualitativa, más que considerarlo como un simple contenedor de datos con los que trabajar de forma cuantitativa, y para ello se estudiaron características no cuantificables como el tener una finalidad educativa, el tipo de funcionalidades que ofrece o la relevancia que tenía para este trabajo de investigación. El único análisis cuantitativo que se realizó fue para contabilizar el número de artefactos surgidos de esos documentos que fuesen compatibles con la plataforma Android, y para contar el número de funcionalidades que se describían en los documentos que se revisaron.

En el análisis por observación que se hizo sobre los artefactos que se estudiaron las conclusiones se tomaron con cautela ya que algunas de las valoraciones sobre los aspectos que se evaluaron se hicieron subjetivamente sin definir previamente unos criterios normalizados o estandarizados. Los que se evaluaron objetivamente fueron el número y tipo de funcionalidades que ofrecían y el grado de finalización del artefacto, mientras que la evaluación subjetiva se hizo sobre aspectos como la fiabilidad, la facilidad de uso y la estética. Un breve resumen de las conclusiones que se obtuvieron de la revisión de los documentos y de la observación directa se puede leer en la sección de estado del arte de este texto.

El artefacto que se creó es el resultado de un modelo de desarrollo que se basó en la técnica de prototipado. Se eligió este modelo porque tiene la gran ventaja de que no era necesario entender completamente cada detalle del problema para poder contar con un primer producto tangible que tuviese unas funcionalidades mínimas a partir de las cuales empezar a construir el artefacto. Antes de implementar el prototipo base, se estudió la arquitectura de la herramienta KeMap y su forma de interaccionar con la plataforma VerilUOC para determinar que a priori era factible el nuevo desarrollo, si bien existían algunos problemas iniciales de compatibilidad entre algunas de las tecnologías que usan KeMap y la plataforma Android, y que se pudieron solventar. Para crear cada nueva versión del prototipo se siguió un ciclo iterativo sobre la etapas siguientes:

1. Se formuló una teoría sobre la compatibilidad de una tecnología (Web Services, Swing, etc.) o una funcionalidad (validación de credenciales de usuario, etc.) concreta.
2. Se buscó y analizó información relacionada con la teoría que se quería comprobar y se derivó una hipótesis en la que trabajar sobre el prototipo.
3. Se ejecutó la hipótesis y y se observaron los resultados obtenidos al modificar el prototipo.
4. Con la observación de los resultados se confirmaba o refutaba la hipótesis.
5. Si se refutaba la hipótesis se volvía al segundo paso y se continuaba con la iteración hasta que se agotaban las hipótesis. Si no se confirmaba ninguna entonces no se aceptaba la teoría y se deducía que la tecnología o funcionalidad no era compatible con la plataforma Android.
6. Si se confirmaba la hipótesis se aceptaba la teoría y los cambios se reflejaban en la creación de una nueva versión base del prototipo.

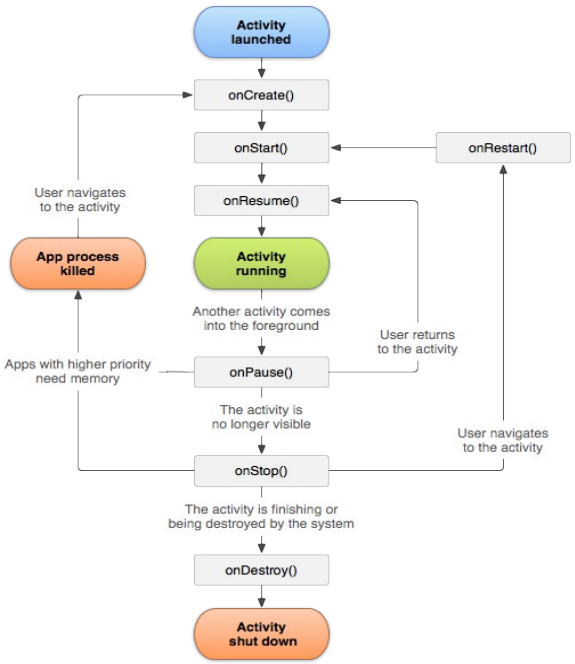
5. Implementación de KemapMobile

En esta sección se describe el proceso de creación de KemapMobile. Comienza con una breve introducción sobre el paradigma de desarrollo en Android, y continúa con la descripción de la implementación para finalizar con los problemas encontrados durante el proceso y las soluciones que se implementaron.

5.1. Android: paradigma de desarrollo

Las aplicaciones para Android se escriben en el lenguaje Java y se ejecutan en máquinas virtuales, para lo que Android cuenta con la máquina virtual Dalvik. Esta máquina virtual se encarga de ejecutar las aplicaciones, el framework de estas y las librerías existen en el entorno de ejecución, en adelante core-libraries, que también están escritas en Java. Una de las características principales de Dalvik es que no es compatible con el bytecode que se genera para la máquina virtual Java estándar, sino que usa su propio bytecode. Muchas de las core-libraries que acompañan a Dalvik en la plataforma Android existen también en el entorno de ejecución de Java estándar, por lo que en ocasiones basta con recompilar el código fuente de una aplicación Java existente para poder ejecutarse en la plataforma Android. Esta plataforma permite trabajar con aplicaciones multi-hilo, y destina una máquina virtual Dalvik a cada aplicación que se ejecuta con objeto de incrementar la estabilidad y la seguridad del sistema[12].

Figura 2. Ciclo de vida de una activity



This image has been reproduced from work created and shared by the Android Open Source Project and used according to terms described in the Creative Commons 2.5 Attribution License. See http://developer.android.com/reference/android/app/Activity.html

La pila de software de Android cuenta con 5 capas: el kernel Linux y herramientas de bajo nivel como los controladores del hardware están en la primera capa. En la segunda capa se encuentran las librerías nativas para trabajar, entre otros, con la base de datos relacional SQLite o con libc (librería estándar para el lenguaje C en Android). El entorno de ejecución de Android, compuesto por las core-libraries y la máquina virtual Dalvik aparecen en otra capas, y sobre estas está la capa del framework de aplicaciones que contiene el gestor de ventanas, el gestor de "activities" y otros, mientras que en la capa superior están las aplicaciones.

La estructura básica de una aplicación para la plataforma Android tiene 4 elementos principales. El primero es un archivo en lenguaje XML, en el directorio raíz de la aplicación, llamado AndroidManifest.xml que describe la aplicación al sistema que lo ejecuta. El resto de elementos son los archivos fuente en el lenguaje Java que contienen la lógica de la aplicación, los archivos en lenguaje XML que implementan la interfaz de usuario, y los recursos que la aplicación necesite como imágenes, ficheros de texto y otros. El paradigma de desarrollo de aplicaciones para Android tiene además otros conceptos únicos como activities e intents. El principal propósito de las activities es interaccionar con el usuario y están formadas por una ventana con la interfaz de usuario y código Java para interaccionar con los elementos que la componen, siendo habitual que las aplicaciones tengan más de una activity. El ciclo de vida de las activities también forma parte esencial del paradigma de desarrollo de Android y se puede ver en la figura 2. La labor de los intents es servir de intermediarios para comunicar las activities entre sí, pudiendo comunicar una activity de una aplicación con otra activity de la misma aplicación o de otra diferente.

5.2. Desarrollo del artefacto

El entorno de desarrollo en el que se creó el artefacto estuvo compuesto por las herramientas software Oracle JDK 1.6 y Eclipse 3.8. A la última se le añadió como complemento el framework ADT v.21 para integrar el SDK de Android en el entorno. Durante el proceso de desarrollo fue necesario el uso de otras herramientas auxiliares que ayudaron a comprobar y a

refutar algunas de las hipótesis que se plantearon al crear los prototipos. Se usó el analizador de paquetes Wireshark 1.6 para estudiar la estructura y el contenido de los datos transmitidos entre KeMap y la plataforma educativa VerilUOC. Otra herramienta auxiliar que ayudó al desarrollo fue SoapUI 4.5 que sirvió para hacer pruebas sobre los servicios web de VerilUOC, junto al depurador de código integrado en Eclipse que se usó para analizar el código de KeMap antes de adaptarlo a la plataforma Android y durante el desarrollo de KemapMobile. Para ejecutar los prototipos se usó una tableta Samsung Galaxy de 10.1" y una tableta virtual de 10.1" que proveía el framework ADT.

El desarrollo del artefacto tuvo cuatro etapas: análisis, diseño, implementación y pruebas. En la etapa de análisis se estudió la posible incompatibilidad entre las diferentes tecnologías que usaría la herramienta y se decidieron las funcionalidades del artefacto con la definición de los casos de uso que se necesitaron: selección de idioma, validación de las credenciales del usuario, selección de un ejercicio, modificación de la salida de una función en la tabla de verdad, modificación del mapa de Karnaugh, modificación de la expresión lógica simplificada, almacenamiento y carga de un ejercicio en el disco, y verificación de la solución dada a un ejercicio. En la etapa de diseño se describió como sería la interacción entre la interfaz de usuario y cada caso de uso analizado, se crearon hipótesis para solucionar los problemas que se detectaron durante el análisis y se definió la interfaz gráfica de usuario. Durante la etapa de implementación se creó la interfaz gráfica que se definió en la fase de diseño, se codificaron los casos de uso y se implementaron las hipótesis que se plantearon en la etapa anterior. En la fase de pruebas se comprobó que el comportamiento de los prototipos era el que se esperaba al ejecutar los juegos de pruebas.

Para implementar el proyecto se enlazaron, como librerías en las opciones de compilación de Eclipse, muchas de las clases existentes en la parte cliente del software de la plataforma VerilUOC, de donde se obtuvieron también otros recursos como imágenes y traducciones a otros idiomas de los textos para usar en la interfaz gráfica de usuario.

También fue necesario crear nuevas clases en lenguaje Java e implementar desde cero la interfaz gráfica y el código necesario para interaccionar con ella. Las clases que se crearon son LoginActivity e InitialConnection para implementar la funcionalidad relacionada con la elección del idioma y con la validación de las credenciales de un usuario contra la plataforma educativa VerilUOC, para lo que se usó el protocolo de autenticación OAuth. Estas dos clases se usan sólo en la activity inicial de KemapMobile junto al archivo login_activity.xml que implementa la interfaz gráfica de la activity. La clase FileChooser se usa como base para la activity que sirve para elegir el destino en el disco de un ejercicio que se quiera guardar. El otro componente de la activity es el archivo filechooser_activity.xml que implementa la interfaz de usuario, La activity que contiene la interfaz principal de la aplicación está compuesta por el archivo kemap_activity.xml y la clase KemapActivity, que usa la clase InitialData para descargar los ejercicios desde la plataforma VerilUOC junto a las clases VeriluocCallsMobile y KemapCallsMobile. Estas dos últimas clases son adaptaciones para KemapMobile de las clases VeriluocCalls y KemapCalls del software cliente de VerilUOC que no eran compatibles con la plataforma Android y de los que se disponía del código fuente.

Las funcionalidades que se diseñaron para la aplicación permiten al usuario obtener un conjunto de ejercicios de acuerdo con su perfil en la plataforma educativa VerilUOC. Al elegir un ejercicio se modifica en la interfaz principal el contenido de la tabla de verdad, las opciones de verificación disponibles para ese ejercicio, el mapa de Karnaugh y los botones de edición de la expresión lógica simplificada. Los ejercicios también tienen una descripción del enunciado en formato PDF que no se descarga inicialmente dentro del conjunto de ejercicios del perfil del usuario, sino que sólo se hace si este pide verlo.

El usuario puede verificar la adición de términos en el mapa de Karnaugh, las agrupaciones de celdas en el mapa, las modificaciones hechas sobre una tabla de verdad en la salida de una función según el valor de las entradas, y comprobar la solución de la expresión lógica simplificada en forma de suma de productos. Las opciones de verificación disponibles se establecen de acuerdo a la información que contiene cada ejercicio, y de acuerdo a ello el usuario puede escoger que soluciones quiere verificar, donde la única opción de verificación que es de uso obligatorio y siempre está disponible es la que comprueba la validez de las expresiones simplificadas. La aplicación permite guardar en el disco el estado de la resolución de un ejercicio para retomarlo posteriormente. La verificación de la solución se hace en la máquina que ejecuta el cliente para garantizar la escalabilidad de la plataforma educativa VerilUOC y evitar posibles sobrecargas durante los picos de utilización de las herramientas de la plataforma[6].

5.3. Contratiempos durante la implementación

Los principales problemas que surgieron durante el desarrollo del artefacto tuvieron que ver con la incompatibilidad entre Android y algunas tecnologías que usa la plataforma educativa VerilUOC. Estas incompatibilidades se identificaron en la etapa de análisis previa a la creación del primer prototipo y con el análisis de los datos que se recabaron se estimó que a lo largo del proceso iterativo de prototipado que se siguió como modelo de desarrollo era factible solucionar los problemas que se detectaron.

Figura 3. Extracto del código que se usó para llamar a los Web Services de VerilUOC

Al pulsar el botón Verificar en KemapActivity se ejecuta un método en la clase KemapCallsMobile.java que construye y ejecuta la llamada al servicio web y obtiene la respuesta:

```
public String dofileverifylocale(int pos, String year, boolean cont) {
  String  resultado = "";
  if(cont == true && plafile.isAllEmpty() == false) {
    Exercise ex = VexercicistableKM.get(pos);
    String namespace = "http://VRLUploadFile.mega.org/";
    String url = "http://wyoming.uoc.es:8080/VRLUploadFile/VerifyFile";
    String nombreMetodoWS = "ParseKeMap";
    String accionSOAP = "http://VRLUploadFile.mega.org/VerifyFile/ParseKeMapRequest";
    Vector<String> property_1 = new Vector<String>(2);    property_1.add("username");    property_1.add(niu);
    Vector<String>property_2 = new Vector<String>(2);
    property_2.add("idexercise");    property_2.add(String.valueOf(ex.getId()));
    Vector<String>property_3 = new Vector<String>(2);    property_3.add("idcourse");    property_3.add(ex.getIdCourse());
    Vector<String>property_4 = new Vector<String>(2);
    String cadena = fileToString(plafile.getFile());    property_4.add("filekm");    property_4.add(cadena);
    Vector<String> property_5 = new Vector<String>(2);    property_5.add("year");    property_5.add(year);
    Vector<String> property_6 = new Vector<String>(2);    property_6.add("DEBUG");   property_6.add("true");
    SoapObject rpc = new SoapObject(namespace, nombreMetodoWS);   //crea un objeto de tipo SoapObject para llamar al WS
    //De acuerdo a la documentacion del ws, hay 6 parametros que debemos pasar. Para obtener
    //informacion del WS , se puede consultar http://wyoming.uoc.es:8080/VRLUploadFile/VerifyFile?WSDL
    rpc.addProperty(property_1.get(0), property_1.get(1));
    rpc.addProperty(property_2.get(0), property_2.get(1));
    rpc.addProperty(property_3.get(0), property_3.get(1));
    rpc.addProperty(property_4.get(0), property_4.get(1));
    rpc.addProperty(property_5.get(0), property_5.get(1));
    rpc.addProperty(property_6.get(0), property_6.get(1));
    SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(SoapEnvelope.VER11);
    envelope.setAddAdornments(false);
    envelope.implicitTypes = true;
    envelope.bodyOut = rpc;
    envelope.dotNet = false;   //el ws no está hecho en .net, si true entonces devuelve error con VerilUOC
    envelope.encodingStyle = SoapSerializationEnvelope.XSD;
    HttpTransportSE androidHttpTransport= null;
    try {
      String conexion = url;
      androidHttpTransport = new HttpTransportSE(conexion);
      androidHttpTransport.debug = true;
      androidHttpTransport.call(accionSOAP, envelope);  //Consulta al WS
      resultado = envelope.getResponse().toString(); //Respuesta del WS
    }catch (Exception e){  return e.getMessage(); }
  }
  else { return "Error al validar la solución."; }
  ...
}
```

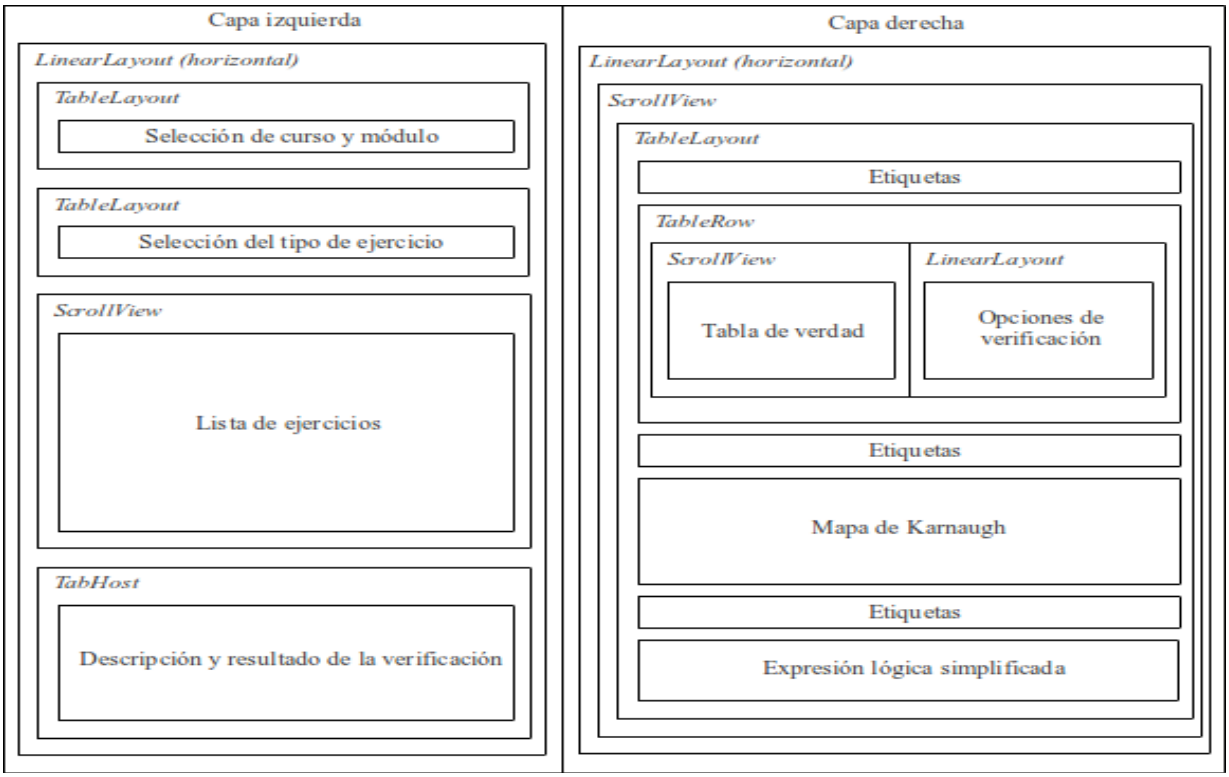
La primera incompatibilidad que se detectó se debía a que la herramienta software que se adaptó (KeMap) usa componentes Swing de la librería javax.swing, que no existe en la plataforma Android. Esto obligó a crear desde cero la interfaz gráfica de usuario del artefacto de acuerdo con el paradigma de desarrollo de Android.

Otra clase que se relaciona con la interfaz gráfica y tampoco existe en la plataforma Android es Point, de la librería java.awt que usa la máquina virtual Java de Oracle. Sí existe la clase android.graphics.Point que tiene parecidas funcionalidades aunque la forma de interaccionar con ella no es la misma que para hacerlo con java.awt.Point al no ser compatibles las llamadas a los métodos públicos que contienen ni tampoco las estructuras de las respuestas que se reciben, si bien esta clase sólo afecta a las funcionalidades que guardan relación con la agrupación de celdas y la modificación de términos en el mapa de Karnaugh.

La funcionalidad para ver el contenido en formato PDF de la descripción de los ejercicios también presentó problemas por no existir en el SDK de Android ninguna librería que permita renderizar el contenido de los archivos PDF. Se identificaron varias aproximaciones factibles a la solución del problema: la primera fue usar librerías de terceros, pero las que se estimaron que se podían implementar (Adobe Reader Mobile SDK, Foxit Embedded PDF SDK) tenían una licencia no libre y se descartó la idea inicialmente. La segunda aproximación consistió en mostrar el contenido del fichero con un visor de archivos PDF, pero también se descartó porque obligaba a disponer de uno instalado en el dispositivo móvil. La aproximación que se escogió fue abrir el archivo en el navegador web que existe por defecto en los terminales móviles que ejecutan Android y mostrar su contenido con la aplicación Google Docs. Esta solución tiene la desventaja de que Google puede cambiar en el futuro las condiciones del servicio y quedar inutilizada la funcionalidad, si bien se estimó válida como solución inicial al problema mientras no se encuentre una alternativa que se base en la primera aproximación.

Otra característica de la plataforma Android que obligó a rediseñar parte de la lógica inicial de la aplicación es la necesidad de lanzar hilos de ejecución diferentes del principal para ciertas operaciones que son susceptibles de impedir la interacción inmediata entre el usuario y la la interfaz gráfica, como la respuesta hecha a algún servicio de la red o las escrituras en el disco. Esta característica está activa desde la versión 3.0 de Android, por lo que si no se

Figura 4. Estructura de la interfaz principal

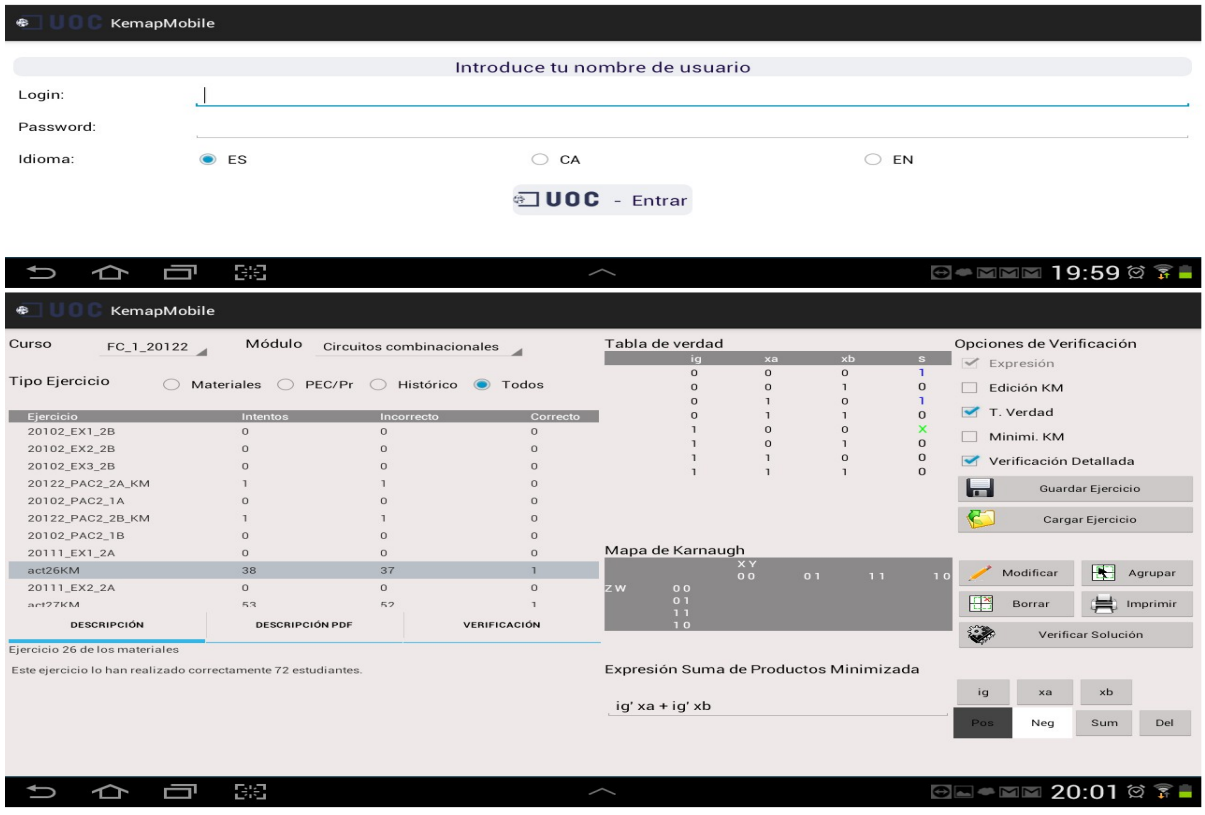


usaban otros hilos por ejemplo para descargar los ejercicios o para verificar las soluciones, el artefacto sería inservible en dispositivos móviles que usen versiones posteriores de la plataforma Android porque se obtendría un mensaje de error que genera la clase android.os.NetworkOnMainThreadException.

También se encontró un problema con el uso de la tecnología Web Services. Las core-libraries de Android no proporcionan clases para trabajar con esa tecnología, de la que sí dispone las core-libraries de la versión que usa la máquina virtual Java de Oracle. La versión para equipos de escritorio KeMap usa librerías de terceros (org.mega.vrluploadfile) que llaman a métodos contenidos en las core-libraries javax.xml.ws y javax.jws, por lo que fue necesario rehacer parte de la lógica existente para verificar la solución de los ejercicios. Para ello fue necesario dejar de usar las librerías de terceros incompatibles, se usó la librería KSoap2 con objeto de facilitar la codificación relacionada con las llamadas a los Web Services de la plataforma educativa VerilUOC. También se usó la herramienta software Wireshark para analizar la estructura de los paquetes de datos que se intercambiaron el software cliente y el software servidor de VerilUOC, y los datos que se recabaron se usaron con la herramienta SoapUI para hacer pruebas tipo "black-box testing" contra los Web Services de la plataforma educativa y estudiar las respuestas a diferentes consultas, antes de codificar la versión del prototipo donde se implementó el uso de la tecnología Web Services para verificar las soluciones de los ejercicios. Un extracto del código se puede ver en la figura 3.

Una dificultad inherente a la plataforma Android se deriva de la existencia de diversos tamaños de pantalla en los dispositivos móviles en las que se esta se ejecuta. El artefacto se creó como prueba de concepto para ejecutarlo en pantallas de 10.1", pero se diseñó para adaptarlo fácilmente a pantallas de menor tamaño al dividir la interfaz principal en dos capas independientes que se pueden separar en una activity para cada una: el lado izquierdo contiene los componentes para seleccionar los ejercicios y ver su descripción, mientras que la capa del lado derecho tiene los componentes que se usan para trabajar con el ejercicio que se eligió y verificar la solución que se introdujo. El diseño general de la interfaz gráfica principal del artefacto se ilustra en la figura 4.

Figura 5. Interfaz de acceso e interfaz principal ejecutándose en Android 4.0



6. Discusión de los resultados

La implementación del artefacto KemapMobile es aún un prototipo que no tiene todas las funcionalidades con las que cuenta la herramienta software KeMap para equipos de escritorio, si bien las que no se implementaron en la versión actual del prototipo pueden añadirse al artefacto al no detectarse incompatibilidades que lo impidan.

Estas funcionalidades son las que permiten almacenar por petición directa del usuario un ejercicio en el disco o cargar un ejercicio que se guardó en una sesión de trabajo anterior, que se puede añadir si se modifican las rutinas existentes que guardan y cargan los ejercicios desde el disco durante el proceso de verificación de una solución para hacerlo también por petición directa del usuario, y las que permiten verificar las agrupaciones y las modificaciones en un mapa de Karnaugh, que se pueden implementar con el uso de la clase Point de la librería android.graphics, si se vuelve a codificar parte de la lógica existente en KeMap y se adapta a la versión para la plataforma Android de la clase java.awt.Point que usa la máquina virtual Java de Oracle.

Las funcionalidades que sí implementa esta versión del prototipo son: la validación de las credenciales del usuario contra la plataforma educativa VerilUOC, una interfaz gráfica de usuario disponible en varios idiomas, la selección de diferentes cursos y ejercicios según el perfil del usuario, la visualización del enunciado o descripción del ejercicio contenida en un archivo PDF, y el almacenamiento y carga de un ejercicio en el disco durante el proceso de verificación. El resto de funcionalidades de las que dispone el artefacto actual tiene relación con la verificación de las soluciones de los ejercicios y permiten elegir qué opciones de verificación se usarán al corregir los ejercicios, entre las que se encuentran disponibles de acuerdo con las especificaciones de cada ejercicio la verificación de: la expresión lógica simplificada, la modificación del mapa Karnaugh (inactiva en esta versión), la agrupación de celdas en el mapa (inactiva en esta versión), la edición de la tabla de verdad y otra que permite obtener pistas al comprobar la solución del ejercicio si la que se introdujo fue incorrecta. Por último, el artefacto es capaz de verificar la validez de las modificaciones en las salidas de una función en una tabla de verdad y la corrección de la expresión lógica simplificada en forma de suma de productos.

La herramienta se diseñó para adaptarla en el futuro al tamaño de las pantallas reducidas de los teléfonos móviles, ya que esta versión sólo se implementó para tabletas mayores de 7”. El inconveniente del tamaño se puede solucionar fácilmente gracias a la división en dos capas (izquierda y derecha) de la interfaz principal, con lo que en tiempo de ejecución se podría ocultar o mostrar cualquiera de las dos según las necesidades.

La figura 5 muestra la interfaz gráfica de acceso a la plataforma VerilUOC y la interfaz gráfica principal del artefacto.

7. Conclusiones

En este trabajo se presentó el proceso creación de KemapMobile, un producto computacional que se ejecuta en dispositivos móviles que usan la plataforma Android y que sirve como herramienta de aprendizaje para la comprensión de la técnica que M. Karnaugh ideó y de otros conceptos que se asocian a la síntesis de circuitos lógicos combinacionales como las tablas de verdad y las expresiones lógicas simplificadas.

El artefacto que resultó de esta investigación sirve como prueba de concepto que muestra la interacción entre KemapMobile y la plataforma educativa VerilUOC de la Universitat Oberta de Catalunya para obtener ejercicios sobre la síntesis de circuitos lógicos combinacionales y validar las soluciones que introduce el alumno al resolverlos.

Al comparar KemapMobile con las herramientas que se describen en el estado del arte de este documento se observó que al contrario que el resultado de este trabajo, la mayoría de las funcionalidades de las herramientas que se estudiaron no se usan para validar las soluciones. No se encontró ninguna que valide la corrección de las modificaciones en una tabla de verdad y sólo dos que validen las expresiones lógicas simplificadas, y su función se centra en calcular una función mínima con los datos que introduce el usuario en una función lógica sin simplificar, en una tabla de verdad o en un mapa de Karnaugh. Otro aspecto que diferencia KemapMobile del resto de trabajos que se estudiaron es que se basa en una arquitectura cliente-servidor que permite al profesorado personalizar los ejercicios disponibles para cada estudiante y recabar datos para seguir la progresión de estos.

El autor considera que el resultado de este trabajo llena un hueco existente en la actualidad, pues si bien existen productos con funcionalidades parecidas para entornos de escritorio, no se encontró ninguno que sea compatible con la plataforma Android y que tenga similares características con respecto a las funcionalidades o al tipo de arquitectura que tiene KemapMobile.

8. Referencias

1. KARNAUGH, M. *The Map Method for Synthesis of Combinational Logic Circuits. American Institute of Electrical Engineers, Part I: Transactions of the Communication and Electronics,* 1953, vol. 72, no. 5. pp. 593-599. ISSN 0097-2452, 1953

2. POTTS, J. *Developing mobile learning applications for electrical engineering courses. , 2011. Proceedings of IEEE Southeastcon,* 2011

3. HACKER, C., SITTE, R. *Interactive teaching of elementary digital logic design with WinLogiLab. IEEE Transactions on Education,* 2004, vol. 47 no. 5 pp. 196-203, 2004

4. JÁUREGUI, R., et al. *Sistema KARMIN: una herramienta para la enseñanza de simplificación de funciones lógicas. , 2010. XVI Congreso argentino de ciencias de la computación, 2010* pp. 265-274, 2010

5. YONG, S.C. *A Game Based Learning Content for Tutoring in Simplifying Boolean Functions, IEEE 10th International Conference on Advanced Learning Technologies (ICALT), 2010*

6. BAÑERES D., et al. Entorno de soporte para el autoaprendizaje en el diseño de circuitos digitales, Jornadas de Enseñanza Universitaria de la Informática (17es: 2011: Sevilla), 2011

7. KLOCK, C. E., et al. KARMA: A Didactic Tool for Two-Level Logic Synthesis. , 2007. IEEE International Conference on Microelectronic Systems Education, MSE '07. , pp. 59-60, 2007

8. ZUBÍA, J.G. *Educational Software for Digital Electronics: BOOLE-DEUSTO., IEEE Proceedings of International Conference on Microelectronic Systems Education, 2003*

9. *KVD* [en línea]. Ver 2.0.1. Martin Hoffmann, c2011. Programa informático disponible en Google Play [descargado el 16 de abril de 2013]

10. MiniKarnaugh [en línea]. Ver. 5.0. Serban Stoenescu, c2013. Programa informático disponible en Google Play [descargado el 16 de abril de 2013]

11. LEE, E. T. *An Expert System for Karnaugh Map Minimization. , 1991. IECON '91 Proceedings of International Conference on Industrial Electronics, Control and Instrumentation,* 1517-1520 vol.2, 1991

12. BRÄHLER, S. *Analysis of the Android Architecture,* Tesis, Karlsruhe Institute of Technology, 2010

9. Publicación del artículo

El artículo se podría enviar a la revista *Computer Applications in Engineering Education de Wiley Periodicals, Inc.*, si bien antes sería necesario traducir el texto al idioma inglés y adecuar el formato a las normas de publicación.

Esta revista publica artículos de investigación en áreas que exploran nuevas herramientas de software y módulos multimedia para la enseñanza de la ingeniería y en otros campos que se asocian con las aplicaciones computacionales educativas relacionadas con las ingenierías.