Mobile Application for an Online Logic Lab (Aplicación Móvil para Laboratorio de Lógica en Línea)

Debora Sarahi Rodriguez Aguilar, Rosa Maria Quiñones Arellano, Juan C. Acosta Guadarrama
Universidad Autónoma de Ciudad Juárez, Mexico

Resumen—This is a detailed work on how to run DLV System, for Answer Set Programming, on mobile Android OS devices, in such a way users can exploit the app potential. The scope includes novice users who wish to run DLV for the first time.
Este trabajo describe de forma detallada y específica, cómo utilizar el Sistema DLV, para Answer Set Programming, en dispositivos con sistema operativo Android, de tal manera que los usuarios exploten el potencial de la app; incluyendo a los usuarios novatos que quieran correr DLV por primera vez.

Index Terms—Aplicación Móvil, Lógica Matemática, Base de Datos Deductiva, ASP, DLV, Android, Laboratorio en Línea.

1. Introducción

DLV [4] es un sistema de bases de datos deductivo que ha sido creado por un equipo de investigación de Italia y Austria (The University of Calabria[1] and The Vienna University of Technology[2]). Como herramienta para los usuarios, DLV es capaz de llevar a cabo deducciones en distintas lógicas a través del lenguaje formal Answer Set Programming [11], [10].

Es importante mencionar, que una base de datos deductiva, es un programa que se encarga del proceso de razonamiento. Este programa, utiliza una base de conocimiento, selecciona los datos y los pasos adecuados para exhibir los resultados.

2. Definición del Problema

DLV es un sistema inteligente que provee un ambiente de programación con la lógica. Originalmente funciona mediante el análisis de la naturaleza de una entrada específica. "El sistema es capaz de aplicar las técnicas que reflejan mejor la complejidad del problema en cuestión, por lo que los problemas *fáciles* se resuelven rápido, mientras que los problemas más *difíciles* solo implican métodos de alto costo computacional"[8]. Es a partir de un proyecto de laboratorio de lógica en línea [12] que surge la iniciativa de implementar DLV a plataformas móviles, utilizando dicho sistema como una especie de caja negra bajo la cual se llevan acabo las operaciones y métodos.

2.1. Objeto de Estudio

El objetivo general de dicha implementación es el estudio, la interacción y la ubiquidad entre los usuarios y el sistema DLV, por medio de un dispositivo conocido y de uso diario, como el móvil. Si bien consultamos páginas web y hacemos uso de programas en nuestras computadoras, el dispositivo móvil se encuentra constantemente en nuestras manos y resulta prático ingresar a la aplicación en lugar de entrar a un buscador, ingresar una URL y esperar a que cargue la información.

Es bien sabido, que los dispositivos móviles son hoy día los más utilizados por su portabilidad y simplicidad. Es debido a estas dos características que se considera viable convertir toda una base de datos deductiva en línea, a una aplicación móvil a la cual se pueda tener acceso sin complicaciones. La movilidad pasa a ser un objetivo primordial de este proyecto, es un término redundante de cierta forma, pero en palabras simples la aplicación móvil para un laboratorio de lógica es eso, un acceso móvil a las funcionalidades de todo un sistema deductivo. En cuanto a la plataforma seleccionada podemos decir que Android está considerado como el sistema operativo más usado del mundo, con una cifra de un 37,93 % de usuarios, Android deja atrás a Microsoft con un 37,91 %, según datos de StatCounter [1].

3. Diseño inicial del prototipo

Para el prototipo de la aplicación móvil se creó un diseño inicial. En él se puede apreciar la estructura de la app, la cual se encuentra compuesta por:

- Pantalla de bienvenida, para la cual se planea marcar una duración de 4 segundos aproximadamente. La duración de la pantalla de bienvenida, permite cargar los componentes de la aplicación.
- Contenido principal, con cuadros de texto editables para las sentencias.
- Menú lateral, con el cual se navegará en las diferentes secciones requeridas para el laboratorio de lógica.

1. The University of Calabria es una institución pública situada al Sur de Italia. Es posible consultar más información acerca de esta institución por medio de su página oficial http://www.unical.it/portale/international/.

2. The Vienna University of Technology es una de las mayores universidades de Viena, en Austria. Para más información se debe ingresar a su página oficial https://www.tuwien.ac.at/en/.

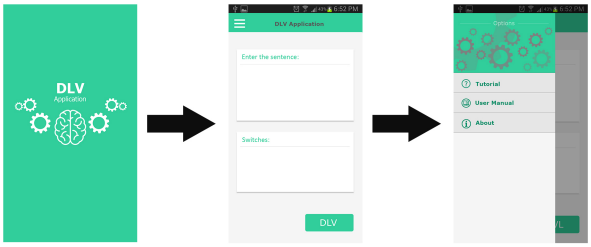

Figura 1. Diseño preliminar del prototipo de la app.

Es importante destacar que el diseño planteado inicial es básico. Esto se debe a que en primer plano se trabajará con la funcionalidad de la app, es decir, con la conexión entre la app y el servidor a fin de que este último efectúe el procesamiento de las sentencias insertadas y las ejecute DLV. Una vez completadas estas acciones, se procederá a continuar con el diseño de la interfaz gráfica. Nuestro estudio se enfoca a definir e implementar potencialidades de este dispositivo en la investigación y aplicaciones de lógica matemática e inteligencia artificial.

3.1. Funcionalidades

Las funcionalidades de la aplicación inicialmente son simples: El usuario ingresa y se encuentra con una pantalla de bienvenida que le ofrece la opción de iniciar e inmediatamente se le permite al usuario ingresar sentencias lógicas en el lenguaje de *Answer Set Programming* (ASP) [10], [11], para ser procesadas mediante un botón. Al ser procesadas las sentencias, se le arroja al usuario el resultado lógico propuesto por el sistema de bases de datos deductivo. Además de ofrecer estas características, la aplicación Android permite ingresar al manual de usuario con la finalidad de dar a conocer los comandos y palabras reservadas propias del sistema. Por lo tanto, si un usuario nuevo ingresa a la aplicación y desconoce en su totalidad lo que se puede hacer con DLV, el manual de usuario le facilitará dicha tarea, por ende la aplicación contará con un menú lateral desplegable que le permitirá al cliente cambiar entre la sección de procesamiento y la de manual de usuario fácilmente.

Existen otras funcionalidades más del lado del dispositivo móvil, que tienen que ver con sus potencialidades y cómo explotarlas combinando con DLV. Sin embargo, estas caen fuera del alcance actual de este artículo. A continuación mostraremos el desarrollo inicial de la misma.

4. Especificación de la tecnología de desarrollo

Para el desarrollo del prototipo de la aplicación móvil, se utilizó el IDE Android Studio[3], Java y PHP, a fin de efectuar una conexión entre la app y el servidor.

3. Android Studio es el IDE oficial para Android, proporciona las herramientas más rápidas para crear apps en todas las clases de dispositivos Android.

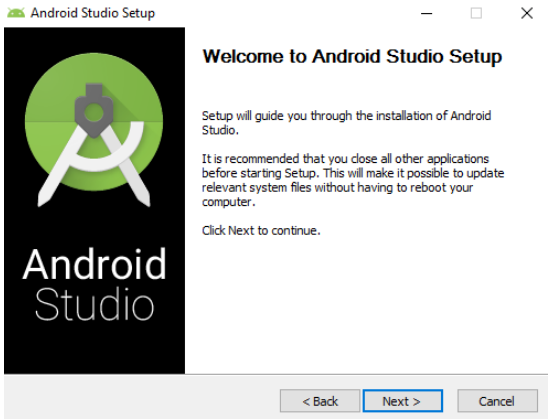

Figura 2. Ejecución del archivo de instalación de Android Studio.

A continuación describimos a detalle los pasos realizados para la instalación de las herramientas en una computadora con Windows 10, que no debe ser muy diferente de realizarse en otras plataformas como MacOS o Linux.

4.1. Instalación y configuración de Android Studio

1. Para la instalación del IDE[4] de desarrollo es necesario ingresar a la página oficial[5] de Android Studio y proceder a descargar el instalador.
2. Una vez finalizada la descarga, se guardará en la computadora un archivo con extensión .exe. Tal archivo, se debe ejecutar e inmediatamente aparecerá una ventana como la de la Figura 2. Se debe seleccionar "Next" para continuar.
3. A continuación, se seleccionan los componentes a instalar—véase Figura 3. El SDK[6] de Android, es un componente vital debido a que nos permite desarrollar aplicaciones y ejecutar un emulador del sistema Android de la versión que sea. Por otro lado, el Android Virtual Device es un componente necesario en caso de que el programador desee hacer pruebas en dispositivos físicos.
4. En la ventana siguiente aparecen los términos y condiciones—véase Figura 4. Es importante hacer lectura de ellos y la aceptación de los mismos, depende de cada usuario.
5. Después, se elige la ruta donde se instalara Android Studio y el componente de SDK—véase Figura 5. Si la instalación se desea realizar en alguna carpeta diferente por fines de organización, es momento de definirlo.
6. La ubicación de los accesos directos es algo opcional. Lo importante de la nueva ventana es proceder con la instalación—véase Figura 6.

4. IDE (del inglés Integrated Development Environment, significa Entorno de Desarrollo Integrado) es un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

5. Página oficial de Android Studio https://developer.android.com/studio/index.html

6. SDK (del inglés Software Development Kit, significa Kit de Desarrollo de Software) es un conjunto de herramientas de desarrollo que le permite al programador crear aplicaciones informáticas para un sistema concreto.

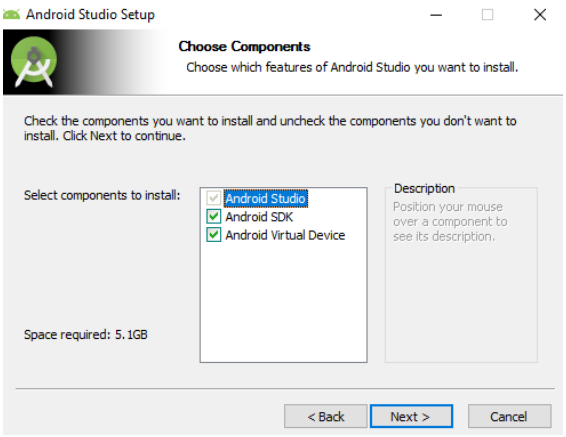

Figura 3. Componentes de instalación SDK y Android Virtual Device.

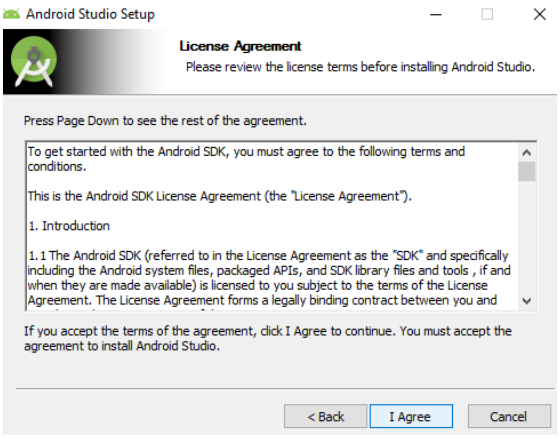

Figura 4. Términos y condiciones de la licencia de software de Android SDK.

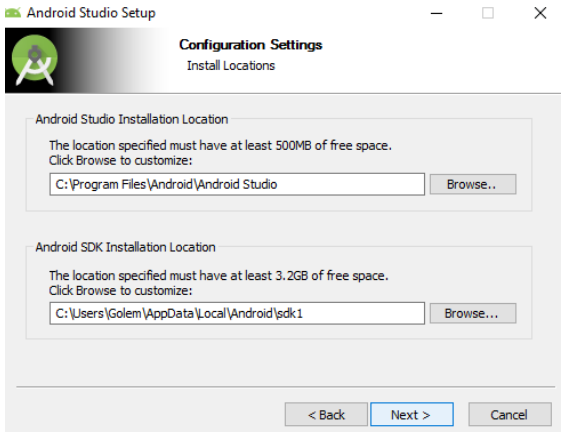

Figura 5. Ruta de instalación de Android Studio y sus componentes.

7. Durante el proceso de instalación de Android Studio, se instalan componentes como SDK, JDK[7] e inclusive NDK[8] con la intención de que la instalación se realice de una forma automática y los usuarios no necesiten buscar cada componente por su propia cuenta—véase Figura 7. Una vez terminado el proceso de instalación, se selecciona "Next".
8. Finalmente, se procede a iniciar Android Studio—véase Figura 8.

7. JDK (del inglés Java Development Kit, significa Kit de Desarrollo de Java) es un conjunto de herramientas de desarrollo, depuración y monitoreo de aplicaciones Java.

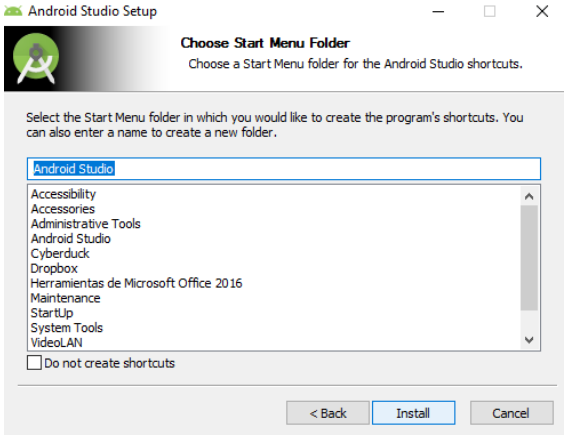

Figura 6. Ubicación de accesos directos de instalación.

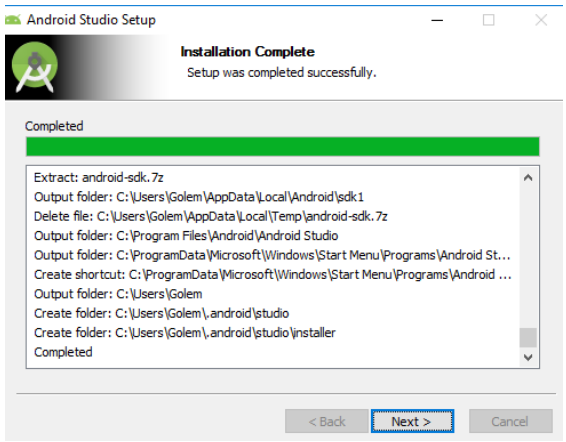

Figura 7. Ventana final de instalación de componentes.

4.2. Creación del proyecto en Android Studio

1. Cuando iniciamos el IDE de Android tenemos la posibilidad de abrir un proyecto existente, crear uno nuevo e inclusive podemos importar uno creado en otro IDE como por ejemplo Eclipse[9]—véase Figura 9.
2. Una vez realizado el paso anterior, Android Studio procede a solicitar la configuración general del proyecto, requiriendo un nombre y una especificación de la ruta en la cual se guardará el mismo—véase Figura 10.
3. La Figura 11 muestra una pantalla en donde se selecciona la plataforma y dispositivos en los que

8. Android NDK (del inglés Native Development Kit, significa Kit de Desarrollo Nativo) es un conjunto de herramientas que permite implementar partes de tu app usando lenguajes de código nativo como C y C++.
9. Eclipse es un famoso entorno de desarrollo de Java, C/C++ y PHP.

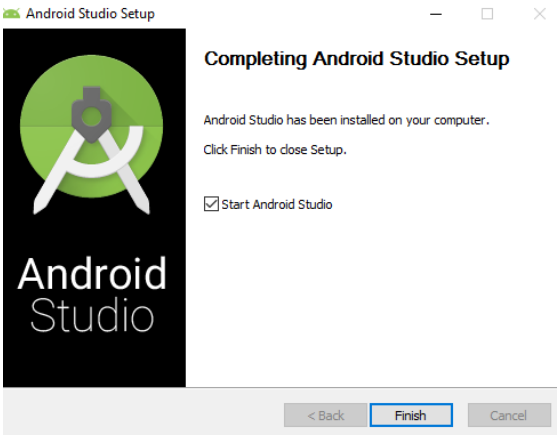

Figura 8. Último paso en la instalación del software Android Studio.

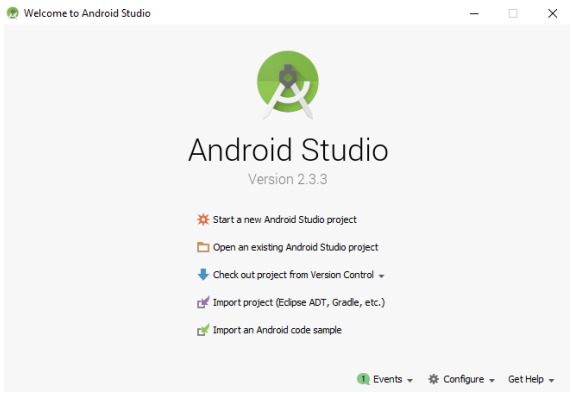

Figura 9. Primeros pasos en la creación de un proyecto en Android Studio.

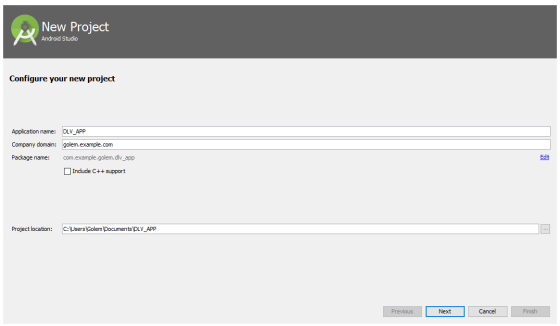

Figura 10. Configuración general del proyecto en Android Studio.

funcionará la aplicación. En este caso optamos por seleccionar teléfonos móviles y tablet, debido a que estos mismos se adecuan a las necesidades del proyecto, principalmente por su uso universal. Por otro lado, hemos considerado conveniente trabajar en la plataforma de Android en sus versiones IceCream-

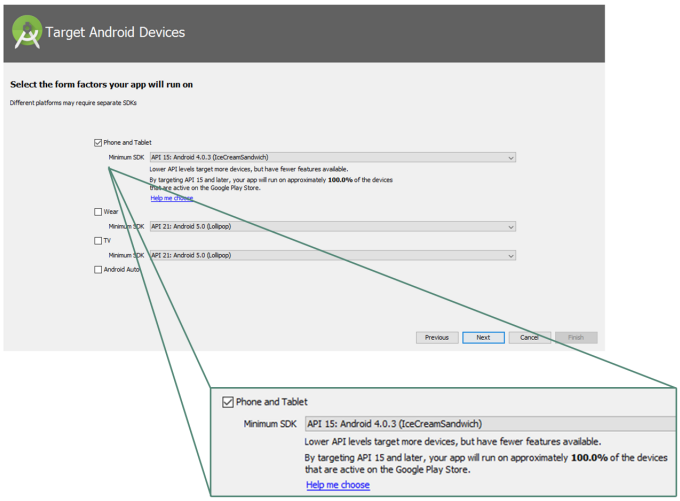

Figura 11. Selección de plataforma y dispositivos compatibles para la aplicación.

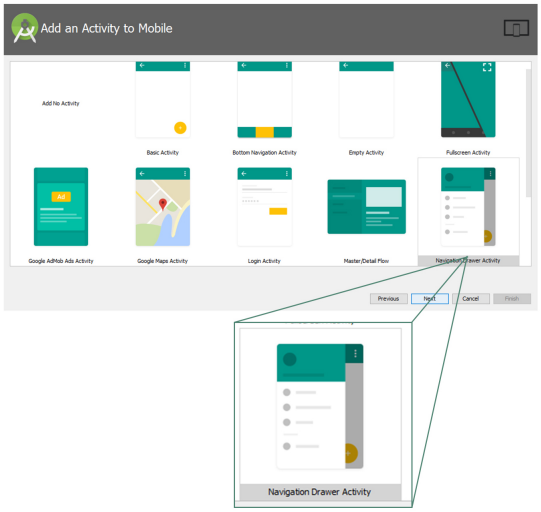

Figura 12. Selección de la estructura inicial del proyecto.

Sandwich 4.0.3[10] a Nougat 7.1[11], ya que de acuerdo a Android Studio de esta manera se abarca un mayor margen de dispositivos y por ende de usuarios.

4. A continuación, se define la estructura inicial del proyecto, seleccionando un Navigation Drawer[12], que implementa un menú lateral en la aplicación—véase Figura 12.
5. Finalmente, es posible realizar modificaciones en

10. Android IceCreamSandwich es la novena versión del sistema operativo para smartphones Android desarrollado por Google. Lanzado el 19 de octubre de 2011 ; Android 4.0 es diseñado con la intención de crear un sistema operativo para smartphones y tablets.
11. Android Nougat es una versión del sistema operativo para dispositivos móviles Android. Fue dado a conocer el 18 de mayo de 2016 en el evento Google I/O.
12. Navigation Drawer, es un panel de navegación lateral que muestra las principales opciones de navegación de la app.

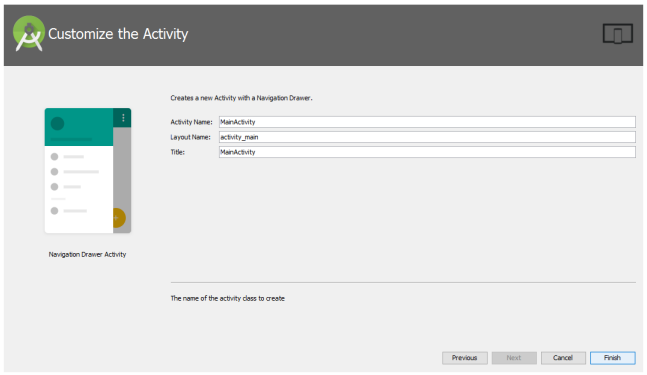

Figura 13. Configuraciones en la clase Activity.

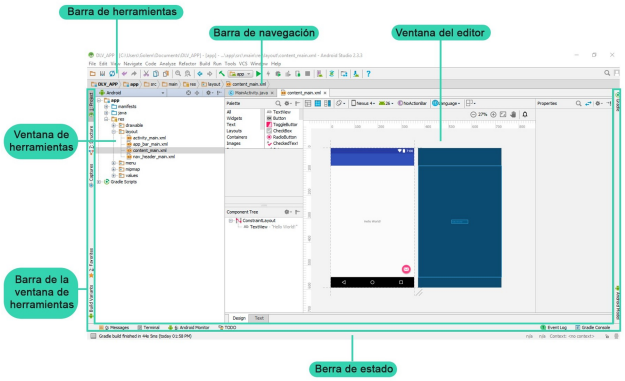

Figura 14. Secciones de la interfaz de Android Studio.

el nombre de la clase Activity[13], o bien utilizar el nombre predeterminado *MainActivity*[14]. Para finalizar las modificaciones se debe pulsar el botón "Finish"—véase Figura 13.

4.3. Interfaz de Android Studio

Cuando se crea un proyecto en Android Studio es importante empezar por conocer el entorno gráfico del software con el que se estará trabajando. En este caso Android Studio cuenta con una interfaz como la que se muestra en la Figura 14.

5. Desarrollo de pruebas en Android Studio

Android es un sistema operativo basado en Linux, con un núcleo de sistema operativo libre, gratuito y multiplataforma. Caracterizado por algunos por su interfaz fluida y sencillez, se posiciona como uno de los sistemas operativos con mayor demanda en el mercado [3]. Por tanto, hoy día Android es compatible con dispositivos tales como celulares, tabletas, Smart TV, Smart Watch, etcétera.

13. Una Activity es un componente de la aplicación que contiene una pantalla con la que los usuarios pueden interactuar para realizar una acción, como marcar un número telefónico, tomar una foto, enviar un correo electrónico o ver un mapa.

14. MainActivity, es un método que utiliza Android para iniciar una instancia invocando los métodos especificos que corresponden a las etapas del ciclo de vida.

5.1. Requisitos

La plataforma bajo la cual se encuentra desarrollada la aplicación, permite la ejecución e instalación automática de un archivo con extensión apk. Dicha operación únicamente se podrá llevar a cabo en dispositivos compatibles en cuanto a especificaciones de software (Sistema IceCreamSandwich 4.0.3 a Nougat 7.1) y hardware (tableta o teléfono móvil).

En caso de haber realizado instalaciones de este tipo con anterioridad el proceso será completamente automático. Para la instalación se necesitará:

- Android Studio actualizado, software gratuito que se puede encontrar en su página oficial https://developer.android.com/studio/index.html?hl=es-419.
- Cable de datos, para conectar el dispositivo móvil a la computadora. Cabe destacar, que la conexión del dispositivo móvil con Android Studio, únicamente se puede realizar por medio de un cable de datos y no de forma inalámbrica.
- Dispositivo móvil con sistema operativo Android 4.0.3 o superior.
- Software DLV previamente descargado e instalado en un servidor.

5.2. Ejecución del proyecto

Android Studio es un IDE caracterizado por la ejecución y renderizado de aplicaciones en tiempo real, además de ofrecer una compilación rápida de los proyectos creados en dicho entorno. Un punto importante acerca de la ejecución de sus aplicaciones es que esta misma se puede realizar de dos formas:

1. Utilizando un emulador de dispositivos Android como por ejemplo, Genymotion[15].
2. Utilizando un dispositivo físico que se encuentre previamente conectado a nuestra computadora.

Nota. En caso de elegir la última opción, Android Studio solicitará que el dispositivo cumpla con requisitos como: versión IceCreamSandwich 4.0.3 o superior, modo de desarrollador[16] y de depuración USB activados.

En la realización de este proyecto se optó por seleccionar la segunda opción, debido a que de esta manera podríamos interactuar con un dispositivo real. A continuación, se explican a detalle los pasos a seguir para su instalación:

1. En caso de no tener abierto el proyecto de la aplicación móvil, procedemos a buscar su ubicación y lo abrimos en Android Studio.
2. El siguiente paso es el más importante porque se trata de la conexión del dispositivo, previamente configurado.
3. En la barra de herramientas de Android Studio, se selecciona la opción de *run*. De esta manera comprobaremos si existen errores en el proyecto o en este caso ejecutaremos el proyecto—véase Figura 15.

15. Genymotion es un emulador de Android que aprovecha la arquitectura x86 para ejecutar de forma fluida y rápida distintos dispositivos Android.

16. El modo de desarrollador es un menú de opciones avanzadas que ofrece Android a los desarrolladores.

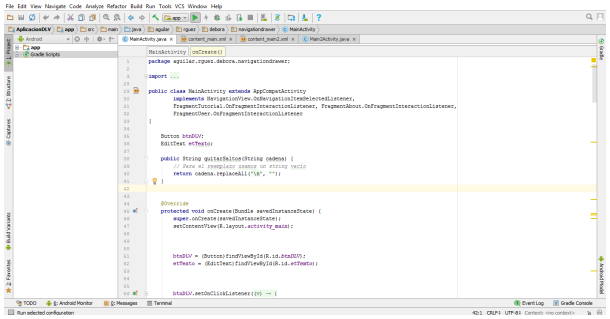

Figura 15. Ejecución de un proyecto en Android Studio.

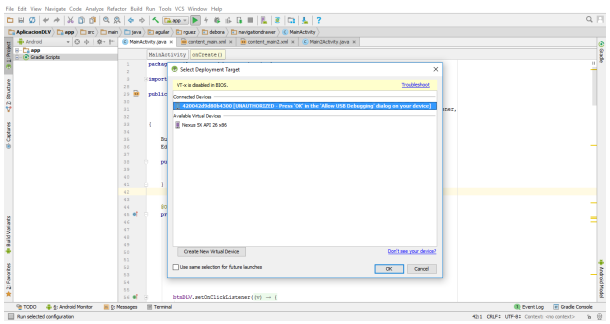

Figura 16. Android Studio, detecta automáticamente dispositivos conectados.

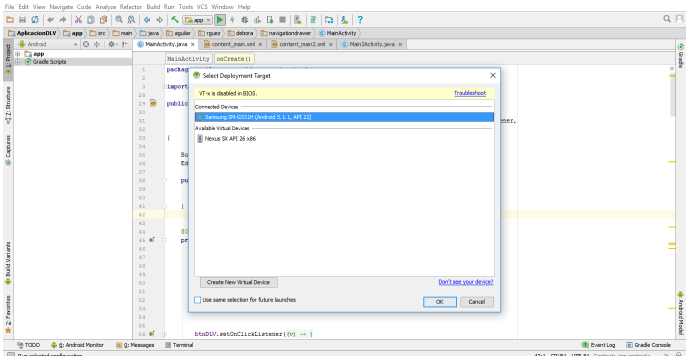

Figura 17. Ventana de selección del emulador.

4. A continuación, se abre una nueva ventana que identifica cualquier dispositivo móvil conectado a la computadora—véase Figura 16.
5. En este paso, se solicitará en el móvil que el usuario permita la depuración de USB[17]. Es importante que se acepte dicha solicitud, de lo contrario se estaría negando la ejecución de la aplicación y Android Studio finalizara el proceso.
6. Finalmente, se selecciona el dispositivo móvil a utilizar—véase Figura 17. Una vez hecho esto, de forma inmediata, la app se instala y ejecuta en el móvil.

5.3. Prueba de funcionamiento del prototipo

Tal como se muestra en la Figura 18, la interfaz de la aplicación móvil ejecuta las funcionalidades antes descri-

17. La depuración USB consiste en permitir que un ordenador se conecte directamente a un smartphone para poder transmitir información.

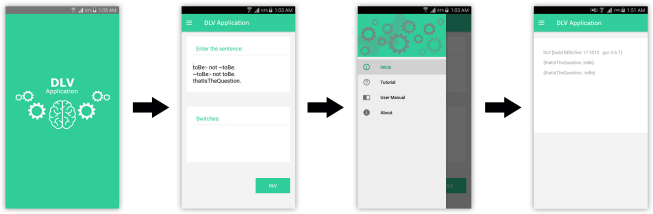

Figura 18. Ilustración de interfaz final.

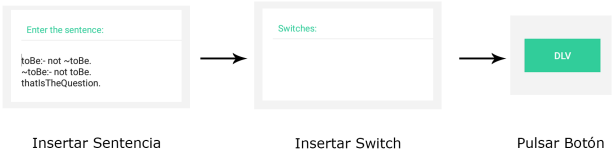

Figura 19. Ilustración de funcionamiento.

tas—véase Sección 3.1. El procesamiento o funcionalidad de la aplicación se lleva a cabo de la siguiente forma:

- El usuario ingresa una sentencia en el cuadro de texto correspondiente o bien hace uso del texto predefinido por la app. Una vez insertada la sentencia a procesar, como se puede ver en la Figura 19, el usuario debe pulsar el botón.

- La sentencia ingresada es enviada al servidor y almacenada en una nueva variable, eliminando los posibles saltos de línea.

El lenguaje de programación Java, permite realizar la conexión del cliente móvil con el servidor en PHP. El envío de sentencias lógicas se ilustra en el siguiente fragmento de código:

```
...
String SentenciaIngresada,sentencia,servidor;
servidor="http://192.168.1.78/android_dlv/
    registro.php?texto=";
SentenciaIngresada=etTexto.getText().toString
    ();
sentencia=quitarSaltos(SentenciaIngresada);

httpHandler handler=new httpHandler();
String txt= handler.Get(servidor+sentencia);
...
```

La recepción de datos y almacenamiento en PHP se realiza en las siguientes líneas:

```
if(isset($_POST['texto'])) {
$texto=$_POST['texto'];
...
```

- La nueva variable es procesada en DLV y almacenada en un archivo de texto, haciendo uso de un interprete de comandos.

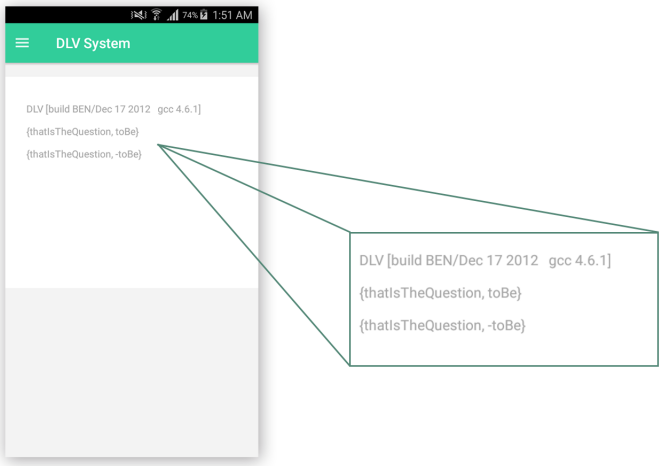

Figura 20. Muestra los modelos.

```
if(isset($_POST['texto'])) {
$texto=$_POST['texto'];
shell_exec("echo '{$texto}' | /home/
    debora_aguilar/Dropbox/htdocs/dlv/dlv2012.
    bin '' -- 2| awk '{print $0}' > dlv.txt");
}
...
```

- Por lo tanto, en el momento en que la aplicación carga un nuevo MainActivity, el servidor busca el archivo de texto almacenado, extrae su contenido y se procede a enviarlo a la aplicación.

```
...
$prueba=fopen("dlv.txt", "r") or die("Error");
while (!feof($prueba)) {
$linea=fgets($prueba);
$aDatos[$contador]=$linea;
$contador=$contador+1;
}
$tamano = sizeof($aDatos);
$comando="";

for ($i=0; $i<$tamano ; $i++) {
$comando.=$aDatos[$i];
}
echo $comando;
```

- Finalmente los modelos de la sentencia procesada, se visualiza en una nueva pantalla de Android, concretamente en un área de texto—véase Figura 20.

6. Descripción del funcionamiento

El procesamiento de la aplicación funciona como una conexión normal entre un cliente y un servidor, en donde la aplicación cumple el papel de cliente y un script de PHP es el servidor. En caso de ser un programador recurrente de HTML y PHP, es posible que se note la similitud entre el envío de datos de un formulario a un archivo PHP por medio del método POST.

6.1. Procesamiento del Servidor

El funcionamiento o procesamiento del servidor se lleva a cabo de la siguiente manera:

- El servidor se encarga de tres acciones principales: recibir datos, procesar datos y enviar respuestas.
- En el primer caso, el servidor verifica si existe alguna solicitud de recepción de datos. En caso de que exista alguna solicitud de este tipo, el servidor acepta los datos por medio de un método POST e inmediatamente los almacena en una variable del mismo tipo.
- Una vez almacenados los datos, el servidor comienza a cumplir su segunda función, por lo cual se procede a pasar los datos a DLV para que los procese, haciendo uso de un shell exec,[18] que interpreta comandos de tipo UNIX.
- Finalmente, el servidor envía su respuesta al cliente. En este caso, el cliente está pidiendo que se haga lectura de un archivo de texto que contiene el resultado del procesamiento de DLV. Para lo cual, el servidor verifica la existencia del archivo; si existe lo abre y vacía su contenido en un arreglo línea por línea. Después el servidor solicita la lectura de los datos almacenados en el arreglo y los imprime como respuesta a la petición del cliente. Por otro lado si el servidor no encuentra el archivo de texto solicitado envía un mensaje de error a la solicitud del cliente.

6.2. Peticiones del Cliente

Las peticiones que realiza el cliente, que en este caso es la aplicación, se realizan por medio de la clase HttpHandler[19]. Esta clase se encarga de crear una comunicación entre un lenguaje Java y un servidor web en PHP. HttpHandler habilita una comunicación cliente y aporta la funcionalidad de efectuar peticiones de tipo get o post a cualquier servidor.

Para fines de este proyecto, se realizaron pruebas en un servidor web local. Por lo tanto, las peticiones eran dirigidas a una dirección IP local, que almacenaba el servidor PHP—véase 6.1.

Hasta este punto se tiene conocimiento de como funciona el servidor y como se comunica la aplicación con un servidor web, sin embargo todavía se necesita conocer la forma adecuada de enviar datos al servidor y recibir la respuesta del mismo.

6.2.1. Envío de datos y respuesta del servidor

Para el envío de datos, lo único que necesitamos es conocer la dirección del servidor y tener definidos los datos a enviar. Una vez cumplidos estos requisitos la aplicación envía los datos haciendo uso de la clase HttpHandler. Como se puede apreciar, en este caso la aplicación esta enviando una cadena llamada "SentenciaIngresada", la cual puede hacer referencia a cualquier texto insertado por el usuario.

```
...
String SentenciaIngresada,sentencia,servidor;
```

18. shell exec: Ejecuta un comando mediante el intérprete de comandos y devuelve la salida completa como una cadena.
19. HttpHandler es un manejador que se invoca para procesar peticiones HTTP.

```
servidor="http://192.168.1.78/android_dlv/
    registro.php?texto=";
SentenciaIngresada=etTexto.getText().toString
    ();
sentencia=quitarSaltos(Sentencia_ingresada);

httpHandler handler=new httpHandler();
String txt= handler.Post(servidor+sentencia);
...
```

La respuesta del servidor se efectúa de la misma manera a excepción de que únicamente se solicita la dirección del servidor, es decir, no se requiere de ninguna cadena de texto que este a la espera de ser procesada. Las acciones que se pueden realizar con la respuesta del servidor son finitas, pero en este caso únicamente se visualiza la respuesta solicitada.

```
...
httpHandler handler=new httpHandler();
String txt= handler.post("http://192.168.1.78/
    android_dlv/registro.php");
TextView t=(TextView)findViewById(R.id.
    etContenedor);

t.setText(txt);
...
```

Nota: El proceso de peticiones y respuestas que efectúa la clase HttpHandler no funcionará a menos que se hallan incluido los permisos de Internet en el archivo correspondiente al manifiesto de Android. Para más información acerca de los permisos requeridos en Android, consultar https://developer.android.com/guide/topics/security/permissions.html?hl=es-419.

7. Conclusión

DLV es un sistema de bases de datos deductivo que se ejecuta por medio de una interfaz de línea de comandos, su funcionamiento consiste en la lectura y procesamiento de sentencias lógicas.

Los usos del sistema DLV son diversos en el ámbito de la investigación. Es por ello que se considera viable concretar la creación de una aplicación móvil, que por un lado atienda a la problemática existente, con respecto al estatus actual de DLV (y otros solvers) como un archivo binario ejecutable en línea de comandos, y que por otro lado combine un sistema lógico con tecnologías móviles que ayuden a explotar las potencialidades de esta última y explorar nuevos temas de investigación.

Este proyecto es solo una iniciativa, el potencial de los sistemas *Answer Set Programming* (ASP) [10] puede llevar a la creación de todo un laboratorio de lógica. Un laboratorio que en teoría pueda implementar un manejo de sesiones para usuarios, la inclusión de nuevos sistemas como CLINGO [6], SMODELS [7] y CMODELS [5].

El desarrollo de aplicaciones móviles, es un medio que se puede explotar en torno al aprendizaje de disciplinas como la lógica o las matemáticas. Una interfaz móvil agrega nuevos atractivos a un proyecto, por el hecho de ofrecer portabilidad, además de otras características poco consideradas en investigación de Inteligencia Artificial, como GPS, datos biométricos, acelerómetro, pantalla táctil, reconocedor de voz, grabadora, etc. Todo esto se puede aprovechar en los usuarios, es decir, aquellos que de forma recurrente optan por hacer uso de un dispositivo móvil para conectarse a Internet, interactuar con servicios o sitios web y aplicaciones nativas.

Referencias

[1] StatCounter, *Operating system market share Worldwide*, StatCounter Global Stats, 2017. [En línea]. Disponible en: http://gs.statcounter.com/os-market-share.

[2] Android Studio, *Cómo descargar Android Studio y SDK Tools — Android Studio*, Developer.android.com, 2017. [En línea]. Disponible en: https://developer.android.com/studio/index.html?hl=es-419.

[3] P. Latam, S. Móviles and A. Moscaritolo, *El 99.6 % del mercado móvil le pertenece a Android y iOS*, PCMag Latam, 2017. [En línea]. Disponible en: http://latam.pcmag.com/sistemas-operativos-moviles/18490/news/el-996-del-mercado-movil-le-pertenece-a-android-y-ios.

[4] DLVSYSTEM, *DLVSYSTEM S.r.l. — DLV*, Dlvsystem.com, 2017. [En línea]. Disponible en: http://www.dlvsystem.com/dlv/.

[5] UTexas, *CMODELS - Answer Set programming System*, Cs.utexas.edu, 2017. [En línea]. Disponible en: http://www.cs.utexas.edu/users/tag/cmodels/.

[6] Potassco, *clingo and gringo*, Potassco.org, 2017. [En línea]. Disponible en: https://potassco.org/clingo/.

[7] P. Simons, *Computing the Stable Model Semantics*, Tcs.hut.fi, 2017. [En línea]. Disponible en: http://www.tcs.hut.fi/Software/smodels/.

[8] DLVSYSTEM, *DLVSYSTEM S.r.l. — DLVSystem site*, DLVsystem.com, 2017. [En línea]. Disponible en: http://www.dlvsystem.com.

[9] Android, *Android - Historia*, Android, 2017. [En línea]. Disponible en: https://www.android.com/intl/es_es/history/.

[10] V. Lifschitz, *What Is Answer Set Programming?*, Department of Computer Sciences,2008. [En línea]. Disponible en: https://www.cs.utexas.edu/users/vl/papers/wiasp.pdf

[11] M. Gelfond and V. Lifschitz. The Stable Model Semantics for Logic Programming. In R. A. Kowalski and K. A. Bowen, editors, *Logic Programming, Proceedings of the Fifth International Conference and Symposium ICLP/SLP*, pages 1070–1080, Seattle, Washington, 1988. MIT Press.

[12] J. C. Acosta Guadarrama. Logic lab, July 2016.