UNIVERSIDAD NACIONAL DE LA PLATA

Facultad de Informática FOLIO 25 Nº

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Titulo: DROPSY: eDucational RObot Programming SYstem
Autores: Fuentes Matias Javier, Fernández Diego Walter
Director: Claudia Banchoff Tzancoff, Claudia Queiruga
Codirector: -
Asesor profesional: -
Carrera: Licenciatura en Informática y Licenciatura en Sistemas

Resumen

En esta tesina se desarrolló una plataforma informática denominada DROPSY destinada a niños, niñas y adolescentes que permite programar robots didácticos controlables tanto de manera presencial como a distancia, mediante una interfaz de programación visual basada en bloques.
Los componentes principales del desarrollo incluyen una aplicación móvil que se integra con el servidor XRemoteBot de carácter preexistente y encargado de comunicarse directamente con los robots, y un servicio de *streaming* confiable y eficiente para hacer accesible a la plataforma de manera remota a través de Internet.
Los desarrollos realizados en esta tesina son de código abierto y están disponibles para su descarga desde https://github.com/dropsy-unlp. El cliente móvil DROPSY se desarrolló para la plataforma Android utilizando la librería Blockly, una herramienta open source para la creación de editores de programación con bloques.
El objetivo principal de esta tesina es ofrecer una herramienta didáctica novedosa que permita a niños, niñas y adolescentes acercarse a la programación y experimentar programar elementos físicos como son los robots didácticos.

Palabras Claves

Programación visual con bloques - enseñanza de la programación – *streaming* de video – aplicaciones móviles – robots didácticos – software libre

Trabajos Realizados

El trabajo comenzó con una investigación de los antecedentes de enseñanza de programación con bloques y robots didácticos y las librerías existentes para su soporte. Asimismo se analizaron las tecnologías de *streaming* de video para hacer accesible a DROPSY desde Internet. Se estudió el servidor XRemoteBot desarrollado por Fernando López en su tesina de grado de Lic. en Informática, Luego se comenzó con el desarrollo de la plataforma DROPSY y la guía de actividades y ya, en una etapa avanzada se realizaron pruebas de concepto con el fin de evaluar el funcionamiento de la herramienta y realizar mejoras tempranas a la misma.

Conclusiones

El desarrollo de la aplicación DROPSY y todas las herramientas que la complementan dieron como resultado una plataforma con gran potencial de ser utilizada en la enseñanza de programación, tanto en escuelas que posean robots educativos como en aquellas que dispongan de conexión a internet, permitiendo en este caso visualizar la ejecución remota de los programas a través de *streaming*. Los resultados de las pruebas de concepto arrojaron resultados alentadores que nos estimulan a continuar el trabajo.

Trabajos Futuros

Como líneas de investigación a futuro se propone:
Profundizar los aspectos didácticos para una mayor integración al entorno escolar. Por ejemplo, realizando pruebas actividades en escuelas. También un análisis del impacto que produce el uso de la aplicación en niños/as y jóvenes.
Finalmente existe la posibilidad de desarrollar una aplicación nativa para entornos IOS, que al momento carecen de una versión de las librerías que posibilitan la programación con bloques.

Fecha de la presentación: Diciembre 2016

DROPSY: eDucational RObot Programming System

Matias Javier Fuentes

Diego Fernández

Tesina de grado de Licenciatura en Informática y Licenciatura en Sistemas

Universidad Nacional de La Plata

2016

Directoras: Claudia Banchoff Tzancoff

Claudia Queiruga

Índice general

Objetivos y fundamentación

En el presente trabajo se plantea la realización de una aplicación móvil con fines didácticos, centrada en el aprendizaje de los aspectos básicos de la programación. Controlar robots físicos a través de un programa informático otorga una nueva dimensión a una disciplina que, a primera vista, parece mantenerse en plano de lo abstracto y la vuelve más atractiva para quienes aún no han tenido la oportunidad de conocerla. Con la intención de facilitar aún más este camino optamos por introducir la utilización de una interfaz gráfica de bloques, de un uso simple e intuitivo, que permite traducir en código las instrucciones que el usuario ensambla con ésta y que luego podrá volcar en plano físico para observar las acciones que el robot realiza.
Adicionalmente se dota a la plataforma de un servicio de transmisión de video que tiene como fin último la utilización del cliente de programación de manera remota, incluyendo de esta manera a aquellas instituciones o usuarios que no disponen de los robots reales en sus instalaciones, pero aún así quieren acceder a los beneficios de esta herramienta.

Se desarrolló entonces una aplicación móvil, que se ejecuta sobre la plataforma Android y se integró la misma con un servidor escrito en Python, desarrollado en el marco de la tesina "XRemoteBot: Un servicio para programar robots en forma remota" , que interactúa en forma directa con los robots y además provee un servicio de streaming que habilita al usuario a observar el movimiento de los robots mientras ejecutan el código generado con la aplicación.

Consideramos que la combinación de tecnologías móviles de programación con bloques y el control en forma remota de robots hace que esta aplicación sea una opción muy simple y atractiva para personas de cualquier edad que estén interesadas en dar sus primeros pasos en programación.

Nuestro objetivo es entonces brindar una nueva herramienta que permite a usuarios sin conocimientos previos de programación familiarizarse con sus aspectos básicos y que a su vez cuente con 3 características distintivas:

- Esté especialmente desarrollada para ser usada en dispositivos móviles.
- Posea una interfaz de programación con bloques.

- Permita el control remoto de robots.

Si bien lo mencionado anteriormente no restringe el ámbito de utilización de la aplicación desarrollada a ningún entorno en particular, planteamos como objetivo principal que la plataforma sea utilizada como herramienta de enseñanza en el entorno educativo de escuelas primarias y secundarias.

Motivación

Las computadoras llegaron a las aulas de escuelas argentinas a través del Programa Nacional Conectar Igualdad en abril del 2010, que entregó a la fecha más de 5 millones de netbooks a estudiantes y docentes de escuelas secundarias públicas de todo el país. Esta iniciativa propone reducir la brecha digital y mejorar la calidad de la educación pública en la escuela secundaria, al promover valores como la integración y la inclusión social. El origen de este programa proviene de la iniciativa global del año 2006 de One Laptop per child[1] (OLPC), la cual se proponía como objetivo distribuir una computadora portátil de bajo costo (U$D 100) especialmente diseñada para la educación en las aulas y la alfabetización digital. El dispositivo estaba pensado para fines educativos, por lo cual tanto el hardware como el software estaban adaptados solamente para esos fines.
Sin embargo esta brecha digital que se intenta superar es solamente una de las múltiples dimensiones que realmente la definen. Otra aspecto importante de esta división es aquello que el "inventor" de la web Tim Berners-Lee (2013) identifica como la brecha que separa a las personas que saben programar de aquellas cuyas habilidades informáticas se limitan a saber usar las aplicaciones estándares como las de ofimática o buscar en Internet.

Saber programar en el siglo XXI nos abre nuevas posibilidades en diferentes planos, no solamente en el de las oportunidades laborales sino además en el de la creación de nuevos productos y servicios por medio de las tecnologías digitales, ya sea como iniciativa propia o un esfuerzo colectivo, permitiendo que el programador deje de ser un mero consumidor de ellas.

Es posible encontrar en la bibliografía múltiples posturas que sostienen que aprender a programar es un habilitador para entender e innovar en un mundo cada vez más tecnológico, permitiendo pensar y actuar creativamente ante problemas inesperados (Resnick M., 2008) (Díaz, Banchoff, Queiruga y Martín, 2014). Bajo esta premisa es necesario que las destrezas en programación sean adquiridas desde edades tempranas, poniéndose en juego la forma en que se enseña a programar.
En sus teorías, Papert (1986) enuncia que "el aprendizaje es más eficaz cuando es parte de una actividad que el sujeto experimenta como la construcción de un producto significativo".
Podemos pensar en estas ideas, enumeradas en su conjunto como "construccionismo" como el concepto de aprendizaje a través de la acción como factor de asimilación, diferenciándose así de

[1] "One laptop pero child" 2006, 17 Oct 2016 <http://www.laptop.org/en/vision/project/index.shtml>

los métodos pedagógicos tradicionales que centran sus esfuerzos en la etapa de transmisión del conocimiento.
Desde el Estado Nacional Argentino se promueve la incorporación de la enseñanza de programación en las escuelas de nuestro país haciéndose efectivo mediante la sanción de la resolución 244 del 2015 del Consejo Federal de Educación. En ella se hace mención a la incorporación de esta temática en la sección "3- Fortalecer el uso pedagógico de las TIC en las escuelas a partir de la implementación de un Plan Nacional de Inclusión Digital Educativa" (PNIDE), específicamente "el desarrollo de propuestas para la profundización en la incorporación de TIC (incluyendo programación), en el marco del PNIDE".

Nuestra tesina se desarrollará como parte de la línea de investigación del LINTI, "Programando con robots y Software libre"[2] aportando con una herramienta innovadora que asista en la enseñanza de programación. Se tomará como punto de partida XRemoteBot, desarrollado en el marco de la tesina "XRemoteBot: Un servicio para programar robots en forma remota", con el objetivo de enriquecerlo y extenderlo mediante la incorporación de una interfaz de programación con bloques que facilite los primeros pasos en el mundo de la programación, orientada especialmente a los niños y niñas de educación primaria y también adolescentes de educación de secundaria básica.

[2]"Programando con Robots y Software Libre" 2009. 27 Oct. 2016. <http://robots.linti.unlp.edu.ar/>

Capítulo 1

Programación en escuelas

1.1. Importancia

El aprendizaje de programación en niños/as de escuelas tanto primarias como secundarias es sin dudas un tema cada día más importante y que está siendo atendida por los gobiernos de diferentes países. Esta relevancia no sólo radica en el hecho de que este aprendizaje prepara a los/as jóvenes del siglo XXI, los cuales vivirán en un mundo cada vez más conectado y tecnológico y cuyo mercado laboral requerirá de estos nuevos saberes, sino que además desarrolla el pensamiento computacional que, según Wing (2006): "implica resolver problemas, diseñar sistemas y comprender el comportamiento humano, haciendo uso de los conceptos fundamentales de la informática".

Estas características hacen que la enseñanza de programación traiga aparejados numerosos beneficios. De acuerdo a un estudio realizado a niños que programaban con Logo, la enseñanza de la programación mejora habilidades en campos como las matemáticas, razonamiento y resolución de problemas (Clements, Battista y Sarama, 2001). Asimismo otros estudios evidencian un aumento de la creatividad y respuesta emocional en niños con dificultades de aprendizaje que fueron entrenados en programación (Clements y Swaminathan, 1995)
Papert (1999) lo resume de la siguiente manera: "cualquiera que haya observado a un niño de uno o dos años usar una computadora, probablemente se haya sentido fascinado por la facilidad con la que opera un aparato que para los adultos puede ser una fuente de infinita frustración. Una cosa es que un niño utilice un juego de computadora; otra muy distinta es que construya su propio juego". Aquí es, de acuerdo a Papert donde yace el verdadero poder de una computadora como medio de enseñanza, en la habilidad para facilitar y extender las habilidades naturales y entusiasmo de un niño por construir, explorar experimentar y sacar conclusiones por sus propios medios.
También existen beneficios fuera de la esfera del pensamiento lógico. Heese (2014) escribe: "Cuando se aprende a programar se aprende cómo revisar tu trabajo en busca de detalles, cómo

aplicar la lógica y cómo persistir en los esfuerzos por llevar a cabo una tarea. Además se aprende a formular una buena pregunta y cómo colaborar con otros, porque hoy la programación se realiza en equipo. Todas estas habilidades y patrones de aprendizaje perdurarán por mucho más tiempo que cualquier lenguaje de programación".

No se trata de construir una sociedad de programadores, sino de reflexionar sobre la relevancia que tiene para la formación de los jóvenes del siglo XXI aprender a programar y la adecuación de los currículos escolares en relación a esta nueva área temática. Esta idea fue expresada de forma brillante por Naughton (2012): "Creemos que todos los niños deberían tener la oportunidad de aprender ciencias de la computación, empezando en la escuela (...) Enseñamos física básica a cada niño, no con el objetivo principal de educar físicos si no porque todos ellos viven en un mundo gobernado por sistemas físicos. De la misma manera, todos los niños deberían aprender un poco de informática desde temprana edad porque van a vivir en un mundo en el que la computación está en todas partes[3]".

1.2. Situación en Argentina

1.2.1 Marco jurídico

En agosto de 2015, a partir de la aprobación de la resolución 263/2015, la Asamblea del Consejo Federal de Educación (el organismo de concertación, acuerdo y coordinación de la política educativa nacional, que está conformado por el Ministro de Educación de la Nación y los Ministros de Educación de todas las provincias) declaró de importancia estratégica para el sistema educativo argentino la enseñanza y el aprendizaje de la programación durante la escolaridad obligatoria, para fortalecer el desarrollo económico-social de la Nación.

Esto conformó un importante primer paso que oficializó la llegada de la programación al sistema educativo obligatorio argentino y además ubicó al país dentro del creciente grupo de naciones que le dan un lugar central al aprendizaje y la enseñanza de la programación como una herramienta clave de la escolaridad.

La resolución creó además la Red de Escuelas que Programan[4], proyecto en el que participaron inicialmente, en una prueba piloto, unos 300 centros educativos estatales. A la vez, la resolución le encomendó a los organismos que forman parte de la iniciativa Program.AR la ampliación de esta red hasta cubrir todas las escuelas del sistema.

---

[3]

[4]"Red de escuelas que programan" 2015. 27 Oct 2016
<http://program.ar/es-oficial-la-programacion-llegara-a-todas-las-escuelas-argentinas/>

1.2.2 Program.AR

Program.AR es una iniciativa del estado nacional creada en el año 2013 que busca que el aprendizaje significativo de las Ciencias de la Computación llegue a las escuelas argentinas. Esta iniciativa se concreta como un proyecto que lleva adelante la Fundación Sadosky del Ministerio de Ciencia, Tecnología e Innovación Productiva, según el cual:
"Program.AR tiene como objetivo impulsar la enseñanza de las Ciencias de la Computación y formar usuarios críticos de esas tecnologías para explotar al máximo las posibilidades y dejar de ser simples consumidores para poder agregar valor en los desarrollos y en el uso cotidiano de la computación. Desarrollar lo que se conoce como "pensamiento computacional" puede contribuir a incrementar la capacidad de resolución de problemas, el pensamiento lógico, la capacidad de abstracción, al tiempo que estas tecnologías brindan plataformas para desplegar la creatividad de los usuarios." [5]

1.3. Iniciativas globales

Hoy en día existe una gran cantidad de organizaciones cuyo objetivo es fomentar la enseñanza de programación, algunas de las cuales tienen varios años de existencia. A continuación se presenta una breve reseña de algunas de las iniciativas globales que resultaron de mayor relevancia para el desarrollo de esta tesina.

1.3.1 Code.org

Creada en el año 2013, es una organización sin fines de lucro dedicada a expandir tanto el acceso a las Ciencias de la Computación como así también la diversidad de los estudiantes que se interesan en las mismas. Es probablemente la organización con más desarrollo ya que es apoyada por las empresas globales más importantes de la industria de software, dentro de las que se encuentran Facebook, Microsoft y Google. Al día de hoy cuenta con más de 13 millones de alumnos inscritos.
Como podemos ver en la Figura 1, Code.org posee además el apoyo de personalidades como Barack Obama, Mark Zuckerberg, Bill Gates y Steve Ballmer, los cuales han realizado videos muy populares fundamentando la importancia de la enseñanza de programacion y que se han viralizado por las redes sociales y fueron reproduciones por millones de personas de todo el mundo.

[5] "Divulgacion: Program.ar" 2013 - <http://www.mincyt.gob.ar/divulgacion/programar-9920>

Dentro del portal de Code.org se ofrecen cursos de programación, material para los docentes, para los/as estudiantes en formatos muy atractivos con propuestas para resolver desafíos de programación de complejidad incremental y acorde a las edades de los/as estudiantes. Asimismo se promueve la competencia global de programación "La hora del código", que invita a todas las escuelas y personas interesadas en la programación a dedicarle una hora a programar. Usualmente se trata de un evento global y se desarrolla en el mes de diciembre. Este año en latinoamérica "La hora del código" fue impulsada por Program.AR y Code.org durante la semana del 3 de octubre.

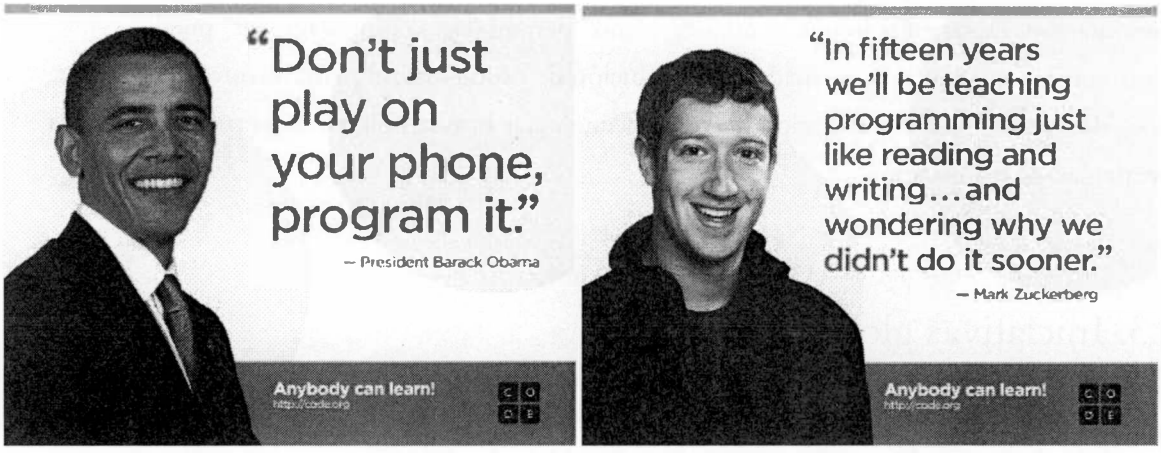

Figura 1: Publicidades de Barack Obama y Mark Zuckerberg para Code.org

La Figura 2 ilustra el formato de las actividades de programación propuestas en Code.org. Las mismas están basadas en programación con bloques, en la parte izquierda de la Figura 2 se muestra el programa que resuelve la actividad propuesta y en la parte derecha se visualiza la ejecución de dicho programa.

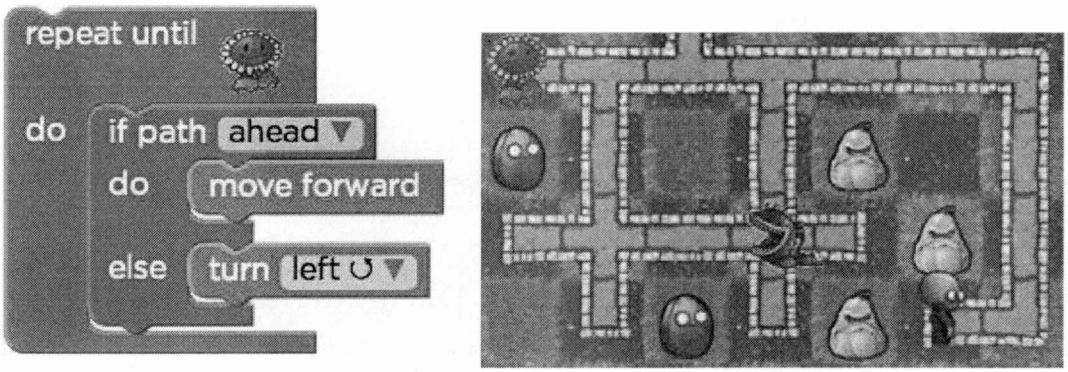

Figura 2: Bloque de código y representación visual, "completando laberinto usando IF ELSE", de learn.code.org

1.3.2. Codecademy

Codecademy[6] es una organización fundada en el año 2011 que tiene como fin proveer a los/as usuarios/as cursos para aprender distintos lenguajes de programación dentro de los que se encuentran Python, Java, PHP, JavaScript y Ruby. A diferencia de Code.org el usuario interactúa directamente con la sintaxis de cada uno de los lenguajes de programación, por lo tanto Codecademy está destinada a un sector distinto el cual generalmente está familiarizado con los aspectos básicos de la programación.
Con 24 millones de usuarios registrados en 2014, Codecademy es uno de los sitios para aprender programación más masivos.

1.3.3. Code school

Code school[7] es un sitio muy similar a Codecademy, cuyo foco es brindar a sus usuarios/as cursos que les permitan aprender distintos lenguajes de programación. Actualmente poseen más de 50 cursos, dentro de los que se encuentran Ruby, JavaScript, HTML/CSS y iOS. Los cursos de este sitio están construidos en torno a historias con el fin de que el usuario tenga la sensación de que está jugando un juego mientras aprende el lenguaje. Esta característica torna la experiencia más entretenida, principalmente para los usuarios más jóvenes.
El sitio cuenta con alrededor de 1 millón de usuarios registrados.

[6] "CodeCademy" 2016. 27 Oct 2016 <https://www.codecademy.com/es>
[7] "Code School" 2016. 27 Oct 2016 <https://www.codeschool.com/>

Capítulo 2

XRemoteBot: Un servicio para programar robots en forma remota

XRemoteBot [8] es una herramienta desarrollada por Fernando López, presentada en el año 2015 como parte de su tesina de grado para la carrera Licenciatura en Informática de la Universidad Nacional de La Plata. El mismo es un sistema que permite la programación de robots en forma remota y consiste de un servidor escrito en Python y un conjunto de clientes escritos en los lenguajes Python, Ruby y JavaScript.
XRemoteBot es software libre y se encuentra disponible en GitHub: https://github.com/fernandolopez/xremotebot.

2.1 El punto de partida

XRemoteBot es de vital importancia en el desarrollo de nuestro trabajo ya que constituye la base sobre la que se desarrolla DROPSY, siendo el objetivo principal de la aplicación Android desarrollada el de agregar un nuevo cliente que se comunique directamente con el servidor de XRemoteBot.

2.2 Características del servidor

El servidor de XRemoteBot está escrito en Python y fue desarrollado con el fin de facilitar la programación de los robots utilizados en el proyecto "Programando con Robots y Software Libre". XRemoteBot puede ser utilizado por instituciones educativas que posean robots similares a los utilizados en este proyecto y por aquellas que no los posean, permitiendo mediante un servicio de streaming de vídeo programar robots en forma remota y visualizar los resultados de

[8] "XRemoteBot: un servicio para programar robots en forma remota " 2015 <http://sedici.unlp.edu.ar/handle/10915/51032>

los programas a través de dicho streaming. Las siguientes características hacen que esto sea posible:

API WebSocket

La interfaz de programación basada en WebSockets[9] desarrollada por Fernando López para XRemoteBot es uno de los componentes más importantes del proyecto, es la que permite que los distintos clientes se comuniquen con el servidor. Al estar basada en la tecnología WebSockets permite el envío y recepción de mensajes de manera asincrónica.

Los mensajes utilizados deben estar en formato JSON y constan de 2 campos obligatorios y 2 campos opcionales:

- Entity: receptor del mensaje (obligatorio)
- Method: método a invocar (obligatorio)
- Args: array con argumentos a utilizarse en el método invocado (opcional)
- Msg_id: valor numérico que identifica al mensaje enviado (opcional)

Dentro de los posibles métodos que pueden ser invocados en el servidor se encuentran:

- Forward: permite mover el robot hacia adelante.
- Backward: permite mover el robot hacia atrás.
- GetLine: retorna los valores de los sensores de línea del robot.
- GetObstacle: retorna si se encuentra un obstáculo a una distancia menor o igual a la indicada.
- Stop: permite detener el movimiento del robot.
- TurnLeft: permite girar el robot hacia la izquierda.
- TurnRight: permite girar el robot hacia la derecha.

Además la API permite la invocación de los métodos:

- Authentication_required: retorna si el servidor requiere autenticación.
- Authenticate: permite autenticar al cliente.
- Get_robots: permite obtener todos los robots disponibles.
- Reserve: permite reservar un robot determinado.

[9] "Websocket protocol" 2011. 27 Oct 2016 <https://tools.ietf.org/html/rfc6455>

Duinobot y Myro

Tanto Myro[10] como Duinobot[11] conforman la base sobre la que se creó XRemoteBot, permitiéndole comunicarse con robots Multiplo N6[12] y Scribbler[13], delegando la comunicación a bajo nivel a estas librerías.

Desde el año 2009, en el marco del proyecto "Programando con Robots y Software Libre" del LINTI , se utilizan robots para enseñar programación en Python a alumnos/as de escuelas secundarias. Inicialmente se utilizaron robots Scribbler, mostrados en la Figura 3, los cuales son controlados por la librería Myro. Ésta es una librería Python desarrollada por el "Institute for Personal Robots in Education"[14] que permite controlar los robots Scribbler tanto a través de cable serial como Bluetooth. Myro requiere que el robot Scribbler tenga cargado un firmware especial que espera instrucciones en su puerto serial y que el mismo se conecte a una placa llamada IPRE Fluke, la cual cuenta con un adaptador bluetooth que le permite recibir comandos desde una computadora.

Los robots Scribbler son de origen estadounidense y por esta razón se dificulta su adquisición, por estos motivos, el equipo de LINTI buscó alternativas de origen nacional para evitar estas dificultades. A fines del año 2011, se adquirieron dos robots Multiplo N6 de fabricación nacional de la empresa RobotGroup, mostrados en la Figura 3. Tanto el software necesario para controlar estos robots como sus especificaciones se distribuyen bajo una licencia abierta.

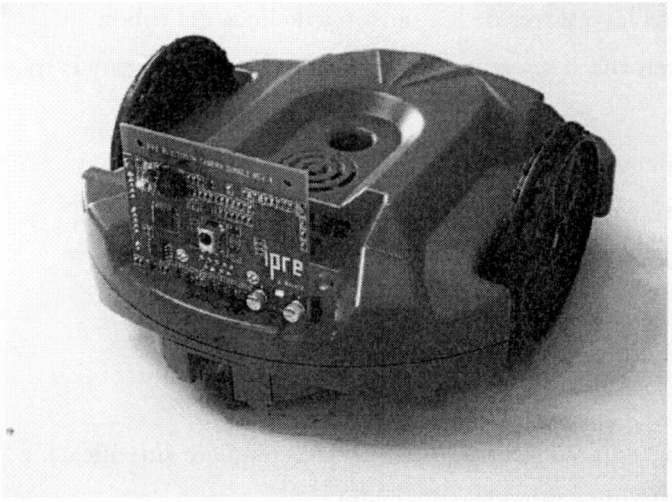

Figura 3 - Robot Scribbler

[10] "Myro Reference Manual" 2015 - <http://wiki.roboteducation.org/Myro_Reference_Manual>
[11] "Duinobot" 2015 - <https://github.com/Robots-Linti/duinobot>
[12] "Robot Multiplo N6" 2016 - <https://prezi.com/qn74kpv3ycg4/robot-multiplo-n6/>
[13] "Robot Myro 2" 2016 - <https://www.parallax.com/product/28136>
[14] "Institute for Personal Robots in Education" 2016 - <http://www.roboteducation.org/"

Duinobot es una librería Python desarrollada en conjunto entre un grupo de docentes y becarios del LINTI y la empresa RobotGroup, quien proveyó de los robots Multiplo N6 al proyecto. Esta librería es la que permite a XRemoteBot comunicarse con los robots "Multiplo N6" mediante la utilización de una placa XBee[15] conectada por USB al servidor.
Los robots Multiplo N6 a su vez, deben tener instalado un firmware especial que implementa el protocolo Firmata[16], el cual tiene como finalidad comunicar microprocesadores con software en una computadora, en este caso a través de placas XBee tanto en el robot como en la computadora que lo controla.

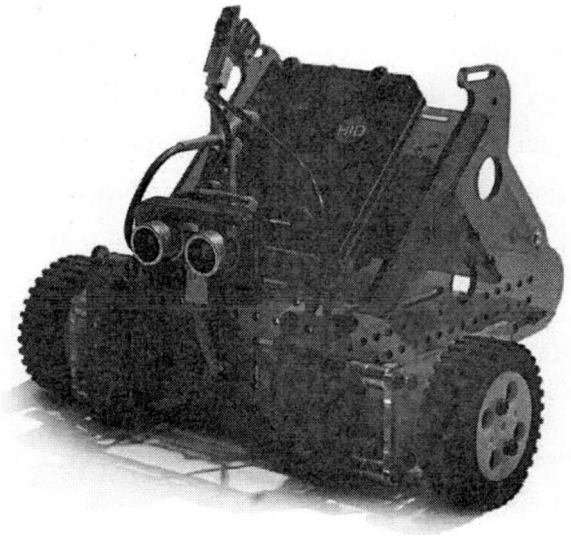

Figura 4 - Robot Multiplo N6

2.3 Modificaciones realizadas a XRemoteBot

Con la finalidad de adecuarlo a las necesidades de DROPSY hemos realizado algunas modificaciones al servidor XRemoteBot, las cuales se detallan a continuación:

- Modificación del modelo de reservas: las reservas no estarán condicionadas a un usuario en particular, de esta manera es posible reservar un robot simplemente indicando su modelo e id, lo cual se adecúa a nuestras necesidades ya que no existe manejo de usuarios en el cliente DROPSY. Además se agregó un método que permite la cancelación de una

[15] "XBee" 2016 - <https://www.digi.com/lp/xbee>
[16] "Firmata Protocol Documentation" 2016 - <https://github.com/firmata/protocol>

reserva, este método es llamado por el cliente DROPSY al finalizar la ejecución del código solicitado. En caso de que no se llame a este nuevo método, las reservas son canceladas automáticamente luego de un tiempo especificado. Esta modificación además incluye la eliminación de todos aquellos métodos relacionados a reserva de robots que no son necesarios para nuestro cliente.

- Script de instalación: se incluyó un nuevo script de instalación para los servicios de XRemoteBot (xremotebot/install_xremotebot.sh[17]) que contempla la descarga desde los repositorios de Github, así como todas las dependencias y librerías necesarias para su funcionamiento. Además incorpora todo lo necesario para iniciar el nuevo servidor de streaming.
- Script de inicio: en relación al servicio de streaming se realizaron las modificaciones correspondientes en el script de inicio (xremotebot/run[18] desde la raíz del proyecto) para iniciar el streaming junto con el resto de los servicios de XRemoteBot.

Las modificaciones incorporadas a XRemoteBot están disponibles en https://github.com/dropsy-unlp/xremotebot

2.4 Integración con DROPSY

Como ya mencionamos, nuestra tarea consistió en desarrollar un cliente del servidor XRemoteBot destinado a plataformas móviles y que pueda ser utilizado como instrumento didáctico por alumnos/as de escuelas primarias y secundarias.

Para lograr esta integración nuestra aplicación hace uso de la interfaz de programación basada en WebSocket de XRemoteBot, a través de la cual nuestro cliente envía mensajes, los cuales son manejados por el servidor y, en caso de ser necesario, se comunica con los robots Multiplo N6 y Scribbler, utilizando las librerías Duinobot y Myro respectivamente. Una vez finalizado el manejo del mensaje, el servidor se encarga de enviar al cliente una respuesta a través del mismo WebSocket. La integración de DROPSY con XRemoteBot, como así también las distintas partes de XRemoteBot descritas en esta unidad se muestran en la Figura 5.

[17] "Script de instalación de XremoteBot" 2016. 02 Nov 2016 <https://github.com/dropsy-unlp/xremotebot/blob/a6a529e626065d902b710670065189435599a620/install_xremotebot.sh>

[18] "Script de inicio de XremoteBot" 2016. 02 Nov 2016 <https://github.com/dropsy-unlp/xremotebot/blob/a6a529e626065d902b710670065189435599a620/run>

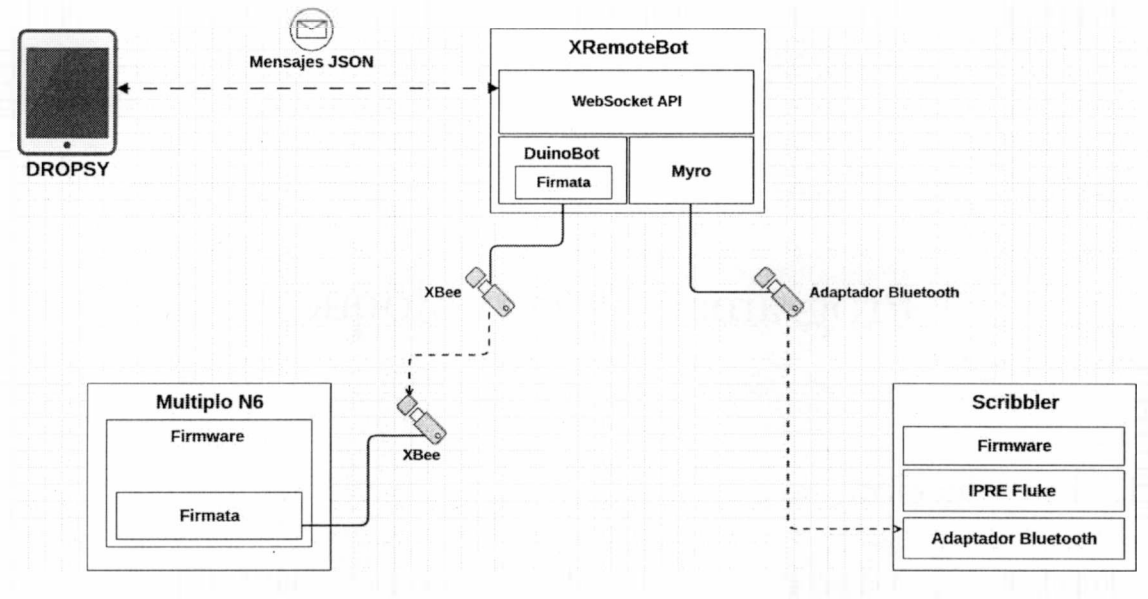

Figura 5 - Arquitectura de XRemoteBot y la integración con DROPSY

Capítulo 3

Programación con bloques

3.1. Introducción

Una de las principales dificultades a las que se enfrentan las personas y principalmente los/as niños/as que quieren iniciarse en la programación están vinculadas a la sintaxis de los lenguajes de programación profesionales. Los mismos poseen sentencias textuales rígidas escritas en lenguaje inglés, difíciles de recordar y que difieren entre los diferentes lenguajes de programación. Es por eso que existe una propuesta didáctica que actualmente está muy difundida para enseñar a programar a niños/as y adolescentes, que evita enfrentarse a los problemas de los lenguajes de programación textuales utilizados en la industria del desarrollo de software y que facilita el acercamiento a la programación. Esta práctica se llama programación con bloques[19] e implementa la metáfora de LEGOS o rompecabezas, consiste en la manipulación de bloques visuales predefinidos, los cuales mediante manipulación directa (drag&drop) pueden ordenarse secuencialmente. Estos bloques se corresponden con instrucciones de un lenguaje de programación y pueden combinarse unos con otros para resolver programas.
Mediante la utilización de lenguajes visuales basados en bloques los/as alumnos/as pueden entonces aprender los conceptos fundamentales de la programación imperativa como son las abstracciones a través de procedimientos o métodos, las estructuras de control, las variables, la asignación, etc. La facilidad de la utilización de bloques es tal que los niños y niñas pueden entenderlos y utilizarlos correctamente desde una edad muy temprana.
De esta manera, la programación visual basada en bloques permite un primer acercamiento a la programación que podría ser continuado con la enseñanza de un lenguaje de programación textual de los utilizados en la industria del desarrollo de software.

[19] "Programming with blocks" 2012. 28 Oct 2016
<http://blog.acthompson.net/2012/12/programming-with-blocks.html>

3.2. Principales herramientas

Por años ha habido herramientas que usan programación visual basada en bloques, algunas de las cuales fueron fuertemente influenciadas por otras, a continuación se detallan las que resultaron más relevantes para el desarrollo de nuestra tesina.

3.2.1. Alice

Alice[20], mostrado en la Figura 6, es una herramienta de programación con bloques muy popular, desarrollada por la Universidad de Carnegie Mellon. Es un entorno de programación 3D escrito en Java que permite la creación de animaciones en 3D.
Está diseñada especialmente para acercar a los/as alumnos/as con la programación orientada a objetos. Permite aprender los conceptos básicos de programación orientada a objetos mientras crean películas animadas o videojuegos, elementos de la vida cotidiana de los/as niños/as.
Los elementos dentro del entorno de desarrollo de Alice se arrastran y sueltan en los lugares donde queramos que interactúen y luego debemos programar su accionar.
Su última versión, Alice 3.0 permite a los usuarios exportar a programas Java.

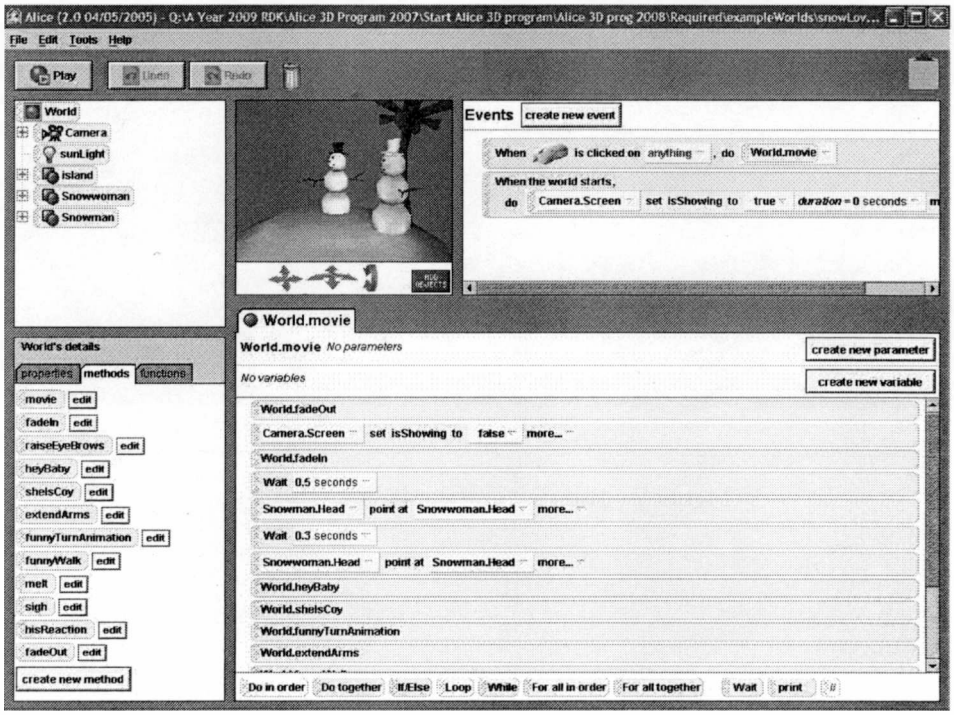

Figura 6 - Captura de pantalla de Alice

[20] "Alice" 2016. 28 Oct 2016 <http://www.alice.org/index.php>

3.2.2. Scratch

Scratch[21], mostrado en la Figura 7, es una herramienta que incluye un lenguaje de programación con bloques y una comunidad online donde los/as niños/as pueden programar y compartir recursos interactivos como historias, juegos y animaciones con personas de todo el mundo.
El objetivo de Scratch es promover la creatividad con medios digitales, el trabajo colaborativo y el pensamiento computacional. Es un proyecto del grupo Lifelong Kindergarten en el MIT Media Lab[22].
Scratch es uno de los proyectos pioneros en el uso y expansión de la programación con bloques, desde su lanzamiento en 2007 permitió a más de 11 millones de jóvenes de todo el mundo la creación de más de 14 millones de juegos, animaciones e historias, permitiendo además a los mismos compartir estas creaciones con su comunidad online.

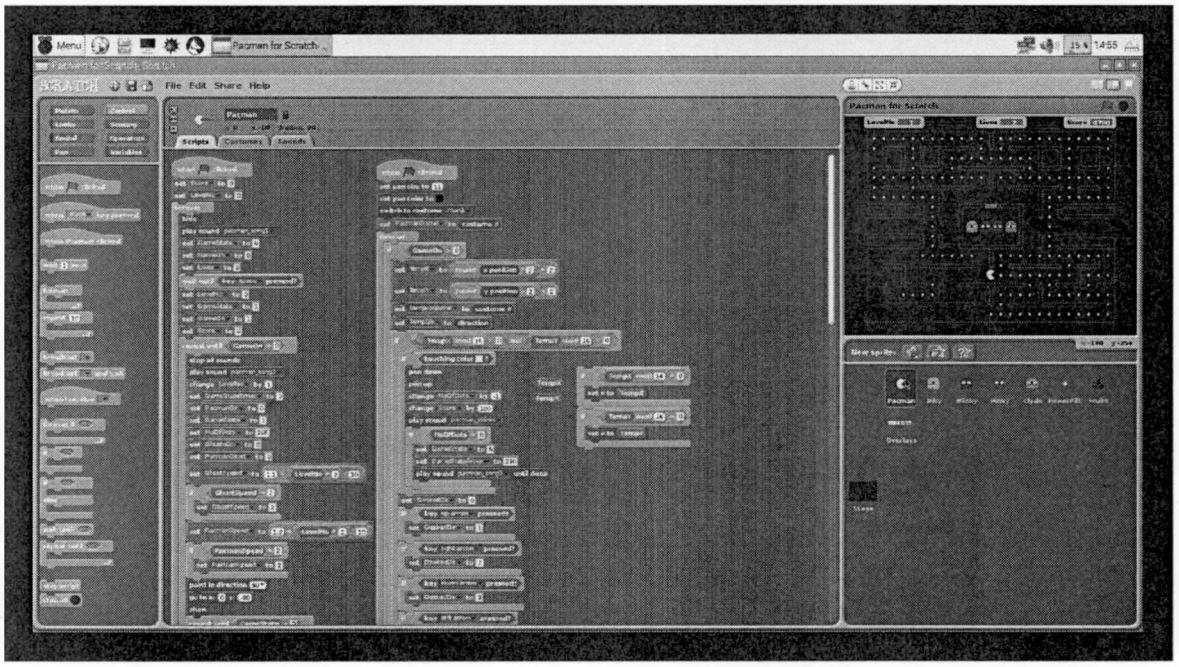

Figura 7 - Interfaz gráfica de Scratch 1.4

[21] "Scratch" 2016. 28 Oct 2016 <https://scratch.mit.edu/>
[22] "Lifelong Kindergarten" 2016. 28 Oct 2016 <https://www.media.mit.edu/research/groups/lifelong-kindergarten>

3.2.3 App Inventor

App Inventor[23], mostrado en la Figura 8, es una plataforma fuertemente influenciada por Scratch que fue desarrollada en conjunto entre el MIT [24]y Google. Está orientada al uso por personas sin conocimiento de programación y permite la creación de aplicaciones móviles para el sistema operativo Android mediante programación visual basada en bloques. Fue presentada en el año 2010 y al dia de hoy lleva creadas más de 4.7 millones de aplicaciones Android.
En un principio APP Inventor utilizaba una librería Java llamada Open Blocks para la programación con bloques, esto hacía que la instalación y ejecución fuese muy engorrosa para usuarios no técnicos. Es por esto que Google se dispuso a migrar la librería Open Blocks a JavaScript, con el objetivo de poder ejecutarse en navegadores web eliminando la necesidad a los usuarios de realizar instalaciones. Esto fue el puntapié inicial de lo que sería Blockly, que no sólo es la interfaz gráfica de un proyecto, sino una librería que puede ser integrada con cualquier proyecto que requiera una interfaz de programación con bloques.
El desarrollo y mantenimiento pasó a ser exclusivamente del MIT en el año 2012 y un año más tarde presentó la versión 2.0, cuya principal novedad fue la capacidad de ejecutarse en navegadores web gracias a la incorporación de Blockly como interfaz gráfica de la programación con bloques.

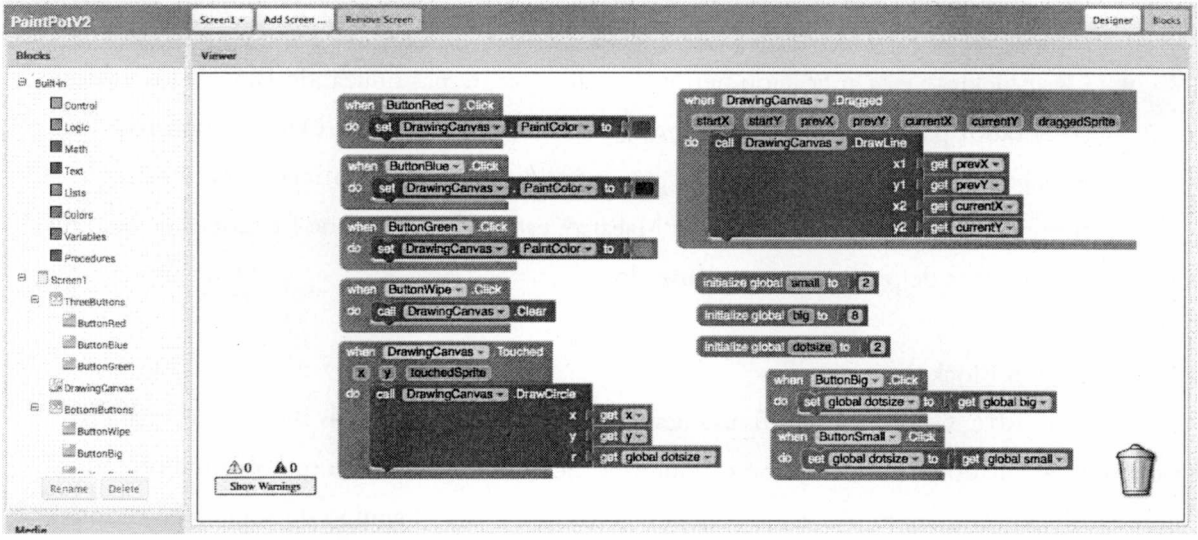

Figura 8 - MIT App Inventor

[23] "App Inventor" 2016. 28 Oct 2016 <http://appinventor.mit.edu/explore/>
[24] "MIT" 2016 <http://web.mit.edu/>

3.2.4 Blockly

Blockly[25] es una librería open source creada por Google, presentada en el año 2012, que permite la creación de editores de programación con bloques como el que se muestra en la Figura 9. Esta librería permite la exportación, a partir de bloques (de código sintácticamente correcto) a JavaScript, Python, PHP y Dart. Además puede ser personalizado para generar código en cualquier otro lenguaje de programación.

Blockly provee por defecto un conjunto básico de bloques para operaciones comunes pero puede ser personalizado agregando más bloques para cumplir con las necesidades del usuario. Estas características hacen que Blockly sea una librería muy popular, usada en cientos de aplicaciones, la mayoría educacionales, entre las que se incluyen:

- Blockly Games[26]: es un proyecto de Google diseñado para alentar a los/as niños/as y jóvenes a programar. Ofrece una serie de juegos educacionales que enseñan a programar. Según Blockly Games, los niños "al finalizar los juegos están listos para usar lenguajes de programación convencionales"[27], dado que en en muchos de los juegos se muestra el código en el lenguaje Javascript.
- Code.org: los desafíos de programación de Code.org están basados en la libreria Blockly. Es un claro ejemplo de la gran utilidad que brinda Blockly, facilitando a los desarrolladores la creación de desafíos, permitiendo adaptar los bloques a las necesidades propias de las actividades.
- OzoBlockly[28]: esta aplicación permite escribir programas utilizando Blockly los cuales son usados para controlar el movimiento y comportamiento de OzoBot, un robot inteligente diseñado especialmente para niños/as. Ozoblocky ofrece 5 niveles de programación, desde Novato hasta Maestro, este último contiene funciones de bajo nivel y conceptos de programación avanzados.

3.2.4.1 Scratch Blocks

En mayo de 2016, Google presentó un nuevo proyecto llamado Scratch Blocks[29], el cual está siendo desarrollado en conjunto con el equipo de Scratch. Esto tiene como objetivo utilizar la librería Blockly en conjunto con los conocimientos que posee el equipo de Scratch en cuanto a diseñar interfaces creativas para jóvenes.

---

[25] "Blockly" 2016. 28 Oct 2016 <https://developers.google.com/blockly/>
[26] "Blockly Games" 2016. 28 Oct 2016 <https://blockly-games.appspot.com/>
[27] "Blockly Games - About" 2016. 28 Oct 2016 <https://blockly-games.appspot.com/about>
[28] OzoBlockly" 2016. 28 Oct 2016 <http://ozoblockly.com/>
[29] "Scratch Blocks" 2016. 28 Oct 2016 <https://scratch.mit.edu/developers>

Scratch Blocks representa el primer paso en un nuevo esfuerzo para desarrollar un set de herramientas que permitan a los desarrolladores crear experiencias de programación de alto nivel para niños/as.

3.2.4.2 Blockly en Android

Google presentó en mayo de 2016 una versión de Blockly para la plataforma Android[30], la cual permite muy fácilmente a los desarrolladores utilizar la programación con bloques en sus aplicaciones Android. Dada su reciente presentación, esta librería aún se encuentra en pleno proceso de desarrollo, por lo cual constantemente aparecen errores los cuales pueden ser reportados para ser arreglados por los desarrolladores. Información sobre Blockly para Android está disponible en https://developers.google.com/blockly/.

Blockly para Android es gratis y de código abierto bajo la Licencia Apache versión 2.0[31]

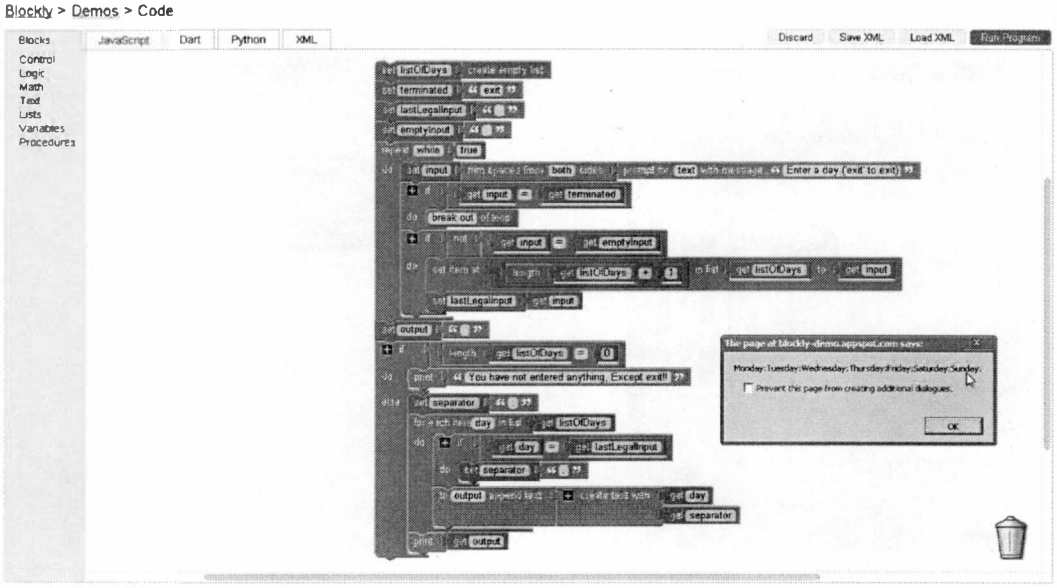

Figura 9 - Interfaz de programación de Blockly

[30] "Blocky for Android" 2016. 28 Oct 2016 <https://github.com/google/blockly-android>
[31] https://www.apache.org/licenses/LICENSE-2.0

3.2.5 Pilas Bloques

Pilas Bloques[32] es una aplicación desarrollada en el marco de la iniciativa Program.AR y la Fundacion Sadosky y con la colaboracion de Huayra[33] lanzada en el año 2015, la cual hace uso de la herramienta Pilas Engine[34].

En la aplicación se proponen desafíos con diversos niveles de dificultad para acercar a los alumnos al mundo de la programación por medio de bloques. Esta aplicación contempla más de 40 actividades diseñadas para dar los primeros pasos en el mundo de la programación. La misma incluye actores y ejemplos prediseñados para que se pueda comenzar a crear variedades de juegos de forma rápida.

Como se puede observar en la Figura 10, la interfaz de Pilas Bloques resulta similar a la de Code.org, ya que el alumno puede programar las soluciones y observar los resultados.

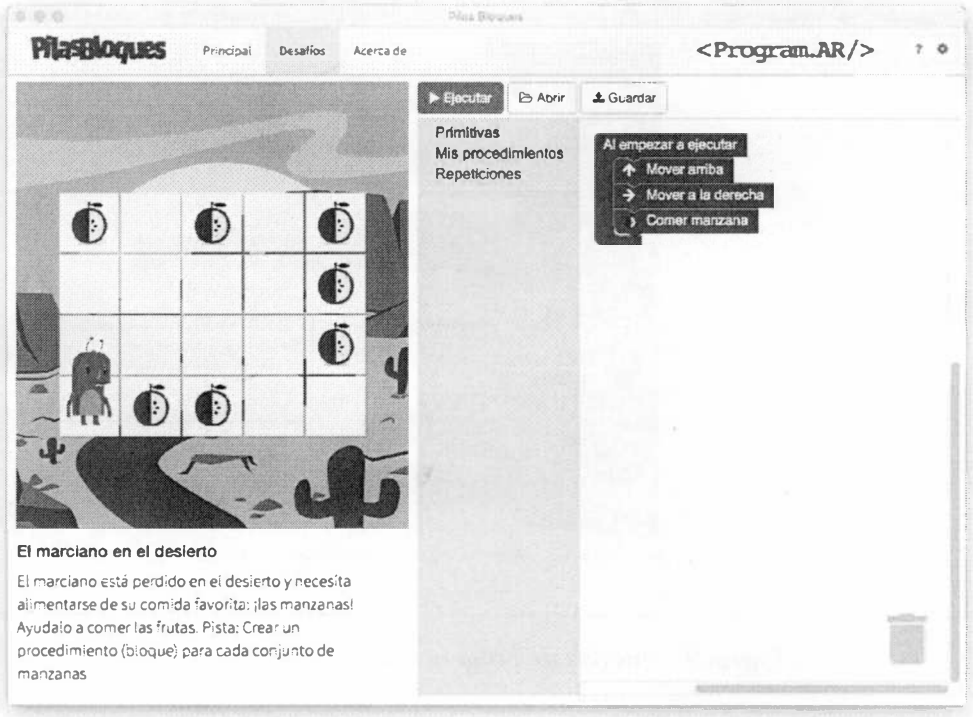

Figura 10 - Interfaz de Pilas Bloques

[32] "Pilas Bloques" 2016 <http://pilasbloques.program.ar/>
[33] "Huayra Linux" 2016 <http://huayra.conectarigualdad.gob.ar/>
[34] "Pilas Engine" 2016 <http://pilas-engine.com.ar/>

3.3 Observaciones

La programación visual basada en bloques se propone como un instrumento didáctico próximo a los/as niños/as y adolescentes que permite acercar la programación como área temática de aprendizaje en las escuelas. Varios de los proyectos que se han descrito en este apartado impulsan la utilización de lenguajes de programación visual basados en bloques para trabajar sobre diferentes contenidos que son de interés entre niños/as y adolescentes.
La programación con bloques en contraposición a la programación con lenguajes de programación textuales evita enfrentarse a una sintaxis rígida, en idioma inglés, a conceptos y herramientas de desarrollo de software complejos, extraños para los/as niños/as, es por ello que que en torno a la programación en bloques se han desarrollado múltiples proyectos mundiales (algunos de los cuales hemos descrito) que suman a millones de usuarios/as en todo el mundo interesados en el aprendizaje de programación a edades tempranas. Es por ello que hemos optado por la programación visual basada en bloques para el desarrollo de DROPSY y de esta manera acercar la programación de robots a niños/as e instituciones escolares.

Capítulo 4

Tecnologías Android aplicadas al desarrollo de DROPSY

4.1 Introducción

Android[35] es un sistema operativo basado en Linux diseñado principalmente para su uso en smartphones y tablets. Actualmente es el sistema operativo móvil más popular del mundo, siendo las ventas de dispositivos Android más que las ventas combinadas de Windows Phone e IOS, de acuerdo a cifras publicadas por los sitios de información de cuotas de mercado NetMarketShare [36] e IDC[37]. Esta popularidad fue, junto con otras razones, lo que nos llevó a elegir Android como plataforma para DROPSY.

4.2 ¿Por qué Android?

Las razones por las que optamos por desarrollar nuestra aplicación para tablets que ejecuten el sistema operativo Android son las siguientes:

1. Portabilidad: las aplicaciones Android nativas son desarrolladas utilizando el lenguaje de programación Java[38] y pueden portarse fácilmente a otros sistemas operativos móviles como Blackberry[39], Symbian y Ubuntu. Además, las aplicaciones Android también pueden portarse fácilmente a Chrome OS.

[35] "Android" 2016. 25 Sep. 2016 <www.android.com>
[36] "NetMarketShare Mobile/Tablet Operating System Market Share 2016. 25 Sep. 2016 <https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1>
[37] "IDC Smartphone OS Market Share" 2016. 28 Oct 2016 <http://www.idc.com/prodserv/smartphone-os-market-share.jsp>
[38] "Java" 2016. 25 Sep. 2016 <www.java.com>
[39] "Blackberry" 2016. 25 Sep. 2016 <global.blackberry.com>

2. Java: es un lenguaje de programación muy poderoso y reconocido, usado en un gran rango de dispositivos y sistemas operativos.
3. Android Studio[40]: es un IDE muy completo y de gran aceptación, basado en el igualmente popular Intellij, que está diseñado específicamente para desarrollar aplicaciones Android. Es rápido, eficiente en el manejo de recursos y fácil de configurar, facilitando en gran medida el desarrollo de las aplicaciones.
4. Cuota de mercado: como se muestra en la Figura 11, con una cuota de mercado de 82,8% (Smartphone OS Market Share, 2015 Q2) en el año 2015, Android supera muy ampliamente a iOS, con 13,9% y Windows Phone con 2,6%. Esta amplia superioridad de Android por sobre el resto de los sistemas operativos para dispositivos móviles es algo que se viene repitiendo año tras año y lo convierte en la opción más conveniente si buscamos desarrollar una aplicación que pueda ser ejecutada por la mayor cantidad de usuarios posibles.
5. Blockly: sin dudas una razón muy importante para la elección de Android fue la reciente presentación de la librería Blockly para Android, la cual facilita en gran medida el desarrollo de aplicaciones nativas basadas en programación con bloques.

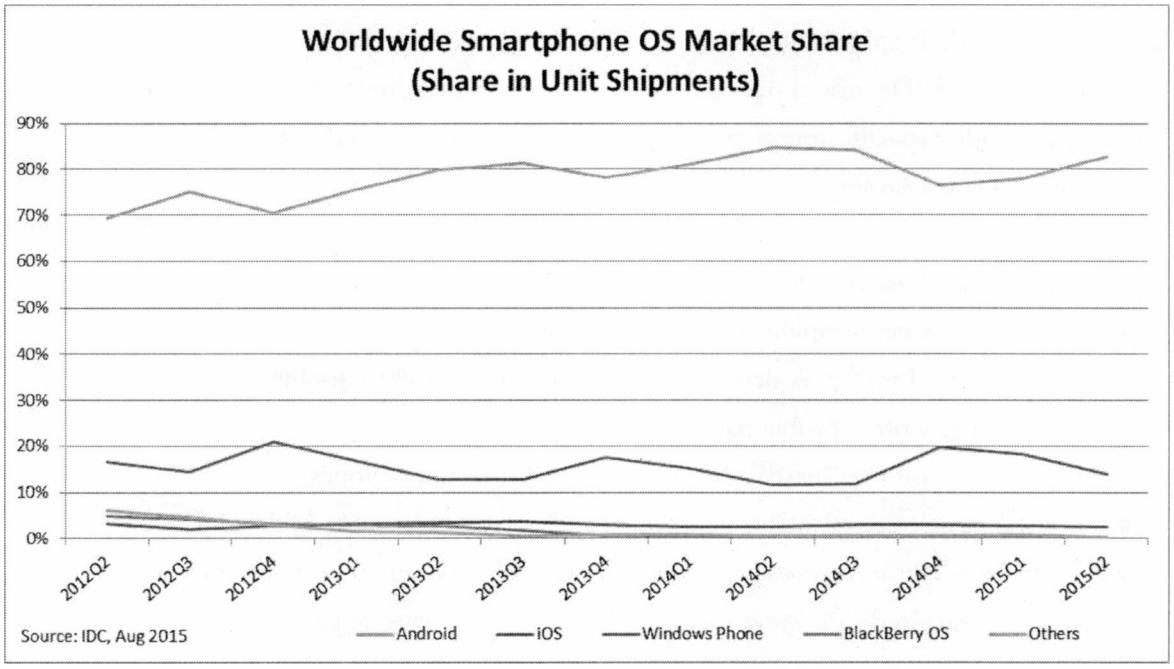

Figura 11 - Dominio del mercado móvil por parte de los principales sistemas operativos móviles

[40] "Android Studio" 2016. 25 Sep. 2016 <developer.android.com/studio/index.html>

4.3 Desarrollo de DROPSY

Teniendo en cuenta las razones antes mencionadas, elegimos Android como sistema operativo sobre el que el se ejecutará nuestra aplicación móvil DROPSY que tendrá como objetivos principales:

- Programar con bloques robots móviles Scribble y Multiplo N6.
- Interactuar con los robots por medio del servidor XRemoteBot.
- Generar código en JavaScript a partir de los bloques, el cual, a partir de su interpretación, lleva a cabo las órdenes que serán enviadas al servidor XRemoteBot.
- Reproducir en la aplicación móvil los movimientos programados de los robots mediante el servicio de streaming de video de XRemoteBot.
- Guardar los programas construidos para su posterior edición y/o ejecución.
- Acceder a los robots de manera ordenada, mediante un sistema de reservas dinámico.

4.3.1 Android Studio

Para el desarrollo de la aplicación se utilizó el IDE Android Studio 2.2.
Android Studio es el IDE oficial para desarrollo de aplicaciones Android. Está basado en IntelliJ IDEA[41] y diseñado específicamente para el desarrollo de aplicaciones Android. Algunas de las características que ofrece son:

- Soporte para *build* basado en Gradle.
- *Refactoring* y arreglos rápidos específicos de Android.
- Herramientas Lint[42] para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones y otros problemas.
- Integración con ProGuard[43] y capacidad para firmar aplicaciones.
- *Wizards* basados en plantillas para crear diseños y componentes Android de uso común.
- Un potente editor de *layouts* que permite arrastrar y soltar componentes de UI y la previsualización de los *layouts* en múltiples configuraciones de pantalla.
- Soporte para construir aplicaciones de Android Wear[44].

[41] "IntelliJ IDEA" 2016. 25 Sep. 2016 <www.jetbrains.com/idea>
[42] "Lint" 2016. 28 Oct. 2016<https://developer.android.com/studio/write/lint.html>
[43] "ProGuard" 2016. 25 Sep. 2016 <proguard.sourceforge.net>
[44] "Android Wear" 2016. 25 Sep. 2016 <www.android.com/wear/>

4.3.2 Gradle

4.3.2.1 Introducción

Gradle[45] es un sistema de automatización de *build* open source basado en las ideas propuestas por Apache Ant[46] y Apache Maven[47], agregando la utilización de un lenguaje basado en Groovy[48] en lugar del lenguaje XML[49] usado en Apache Maven para declarar la configuración del proyecto. Este sistema es presentado como la herramienta de *build* por defecto en Android Studio, pero esta es sólo una de las dos razones por la que la utilizamos en nuestro proyecto, siendo la otra la característica de poseer un lenguaje de dominio específico basado en Groovy para la configuración de los proyectos. Esta característica resulta en scripts de build mucho más cortos y claros que los escritos para Ant o Maven.

4.3.2.2 Configurando Gradle

Dentro de las cosas que debemos decidir al comenzar a desarrollar una aplicación para Android están: la versión de Android SDK queremos usar para compilar nuestra aplicación, la versión mínima de Android SDK que va a ser soportada por la aplicación y la versión de Android SDK a la cual apunta nuestra aplicación. Estas 3 cuestiones las debemos configurar en nuestro script de build Gradle con los parámetros compileSdkVersion, minSdkVersion y targetSdkVersion respectivamente.

- compileSdkVersion: la versión de Android SDK que vamos a usar para compilar la aplicación resulta fundamental ya que esto es lo que nos permite o restringe usar una determinada funcionalidad. Lo recomendado en este caso es siempre utilizar la última SDK disponible, ya que esto nos permite tener los últimos chequeos de compilación, evitar usar APIs deprecadas y tener a disposición las nuevas APIs. Al momento de desarrollar la aplicación la última SDK disponible es la número 24 (Android 7.0 Nougat) por lo que ésta fue la elegida.
- minSdkVersion: la versión mínima de Android SDK que vamos a requerir para ejecutar la aplicación es la que nos va a determinar cuáles son los dispositivos Android que podrán ejecutarla. En este caso elegimos una minSdkVersion de 16 (Android 4.1 Jelly Bean), y de esta manera los dispositivos que posean Android 4.1 o mayor podrán utilizar DROPSY. Elegimos esta versión mínima ya que una de las librerías indispensables para nuestra aplicación posee un minSdkVersion de 16, y como regla general, el

[45] "Gradle" 2016. 25 Sep 2016 <gradle.org>
[46] "Apache Ant" 2016. Sep. 25 2016 <ant.apache.org>
[47] "Apache Maven" 2016. Sep. 25 2016 <maven.apache.org>
[48] "Groovy" 2016. Sep. 25 2016 <www.groovy-lang.org>
[49] "XML" 2016. Sep 25 2016 <www.w3schools.com/xml>

minSdkVersion debe ser igual o mayor al más alto de los minSdkVersion de las librerías que utilicemos.

- targetSdkVersion: el valor recomendable como objetivo para una aplicación es la versión más alta en la que sabemos que la aplicación funciona de manera correcta. En nuestro caso no tuvimos mayores inconvenientes en colocar un targetSdkVersion de 24.

4.3.3 Blockly

4.3.3.1 Introducción

Blockly es la librería más importante de nuestra aplicación pues nos permita cumplir con nuestro objetivo principal: programar robots mediante programación en bloques.
Como ya describimos en la sección anterior (3.2.4.2 Blockly en Android) la librería de Blockly para Android está disponible a partir de mayo de 2016 y nos permite programar en bloques y generar código en lenguaje JavaScript a partir de la programación en bloques. Luego de una breve prueba de concepto, con la cual pudimos probar la facilidad de integración de la librería, la interfaz gráfica y las posibilidades de lograr los desarrollos que teníamos en mente, nos sentimos muy satisfechos y elegimos utilizarla en nuestra aplicación.
El código fuente de Blockly para Android está disponible en GitHub[50], donde también se menciona que "Blockly para Android es una avance de desarrollador del editor de Blockly construido con vistas y fragmentos estándar de Android, ofreciendo una mejor responsividad al tacto y una más fácil integración con aplicaciones Android."

4.3.3.2 Instalación

La instalación de las librerías necesarias para utilizar Blockly consistió en los siguientes pasos:

1. Clonamos el código fuente de Blockly para Android desde GitHub.
2. Importamos a Android Studio el proyecto descargado, el proyecto consta de 4 módulos:
    - blocklydemo: en este módulo se encuentran varias aplicaciones desarrolladas por Google a modo de ejemplo. Estas aplicaciones utilizan las librerías de Blockly para Android para demostrar su funcionalidad.
    - blocklylib-core: es el núcleo de la librería de Blockly para Android. Incluye las clases de modelo, controladores y vistas base.
    - blocklylib-vertical: junto con blocklylib-core constituyen las 2 librerias que necesitamos para nuestra aplicación. Este módulo depende de blocklylib-core e

[50] https://github.com/google/blockly-android

incluye la vista de los bloques apilables verticalmente y clases asociadas a esa vista.

- blocklytest: este módulo contiene todo tipo de tests para comprobar que el código contenido dentro de las librerías funciona de manera correcta.

3. Realizamos un build del proyecto y se generaron las librerías necesarias que importamos en nuestra aplicación:
   - blocklylib-core-release.aar
   - blocklylib-core-debug.aar
   - blocklylib-vertical-release.aar
   - blocklylib-vertical-debug.aar
4. Luego nos dirigimos a nuestra aplicación y ya estamos en condiciones de importar las librerías generadas.

4.3.3.3 Creación del Activity de programación con bloques

Primero y principal debemos crear un Activity que extienda de **AbstractBlocklyActivity**, este Activity va a ser el encargado de mostrar las vistas de programación con bloques (Figura 12), así como también permitirnos interactuar con los mismos, interactuar con algunos botones predefinidos, etc.

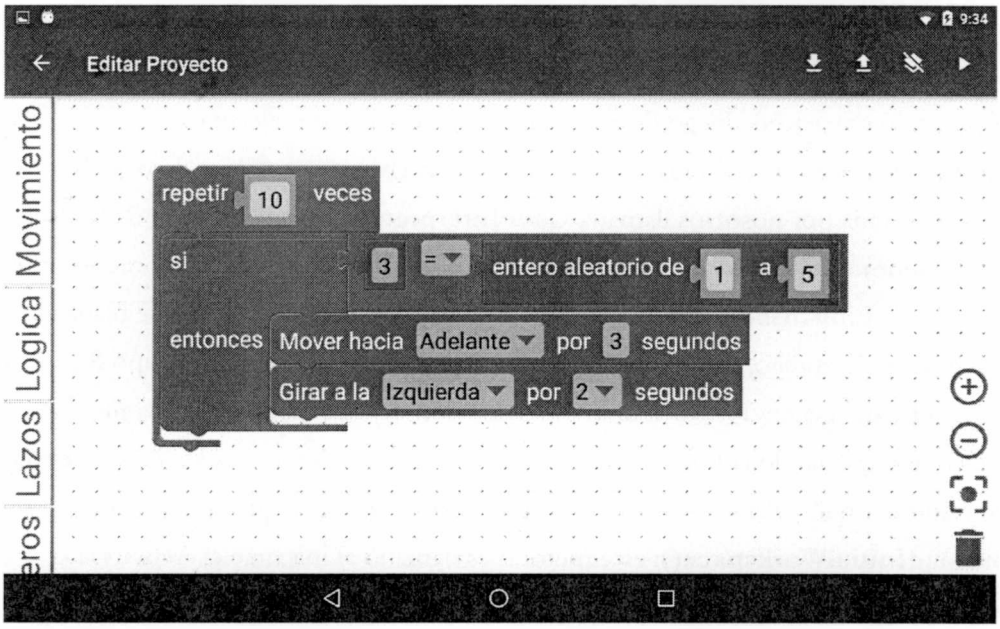

Figura 12 - Interfaz de programación Blockly en la aplicación DROPSY

Además de la creación de este Activity, nuestra tarea consiste en sobrescribir ciertos métodos de AbstractBlocklyActivity que nos interesa personalizar. Estos métodos son los siguientes:

- **onSaveWorkspace**(): este método es llamado cuando el usuario toca el botón guardar dentro de la vista de programación con bloques. En nuestro caso implementamos una base de datos para guardar los proyectos de los usuarios, por lo que necesitamos personalizar este método para poder guardar el proyecto en nuestra base de datos. Básicamente implementamos este método de tal manera que nos permita mostrar un control de UI de diálogo en el que puede ingresarse el nombre del proyecto y que luego de la confirmación por parte del usuario almacena el nombre del proyecto en la base de datos junto con la cantidad total de bloques que posee el proyecto. Los bloques se almacenan en formato XML de manera de poder restaurarse fácilmente cuando se carga el proyecto.
- **onLoadWorkspace**(): este método es llamado cuando el usuario toca el botón cargar dentro de la vista de programación con bloques. Como ya mencionamos, necesitamos que este método interactúe con la base de datos para poder cargar los proyectos desde la misma. Personalizamos este método para que dirija al usuario a una nueva Activity, en la cual el usuario pueda seleccionar el proyecto que desea cargar dentro de todos los proyectos guardados.
- **getCodeGenerationCallback():** este método es el más importante de los que implementamos ya que es el que nos permite indicar cual es el objeto que va a realizar las acciones que necesitemos una vez que sea generado el código fuente a partir de los bloques, lo que sucede luego de que el usuario toca el botón ejecutar.
  Implementamos este método de manera que devuelva una instancia de una clase implementada por nosotros llamada **CodeInterpretation**, que extiende de la clase **CodeGeneratorCallback**. Nuestra clase **CodeInterpretation** debe sobreescribir el método **onFinishCodeGeneration**(), el cual es llamado luego de que se realiza la generación de código, por lo que recibe como parámetro un objeto de tipo String con el código fuente generado. Nuestra implementación del método realiza una interpretación del código utilizando la librería Duktape, que mencionaremos más adelante, y con esto se comunica con el servidor para comunicarle las instrucciones que debe ejecutar el robot.
- **onLoadInitialWorkspace()**: este método es llamado al iniciarse el Activity. Lo implementamos simplemente para verificar si se trata de la carga de un proyecto guardado o de un proyecto nuevo. En caso de que sea un proyecto guardado, obtenemos la información correspondiente al mismo para utilizarla en caso de necesitarla, por ejemplo para mostrar el nombre del proyecto en el ActionBar, o en caso de guardar para sugerir utilizar el mismo nombre que el proyecto ya posee.

4.3.3.4 Creación de bloques personalizados

Como mencionamos anteriormente, Blockly para Android permite la creación de bloques personalizados que pueden ser utilizados en la aplicación[51]. En nuestro caso necesitamos crear 2 nuevos bloques para controlar el movimiento de nuestros robots: un bloque para controlar los movimientos de giro y otro para controlar los movimientos tanto hacia adelante como hacia atrás. Para lograr esto debemos hacer lo siguiente:

1. Modificar el archivo assets/default/toolbox.xml, en el cual se puede especificar cuales son las categorías de bloques que se van a poder utilizar y qué bloques contiene cada una. En este archivo agregamos una nueva categoría que llamamos Movimiento y que contiene nuestros 2 nuevos bloques, como se muestra en el Código 1:

```
<category name="Movimiento" colour="210">
      <block type="movement_turn"></block>
      <block type="movement_move"></block>
</category>
```

Código 1 - Configuración en Blockly de la categoría Movimientos y de los bloques de giro y movimiento

2. Crear un nuevo archivo dentro de /assets/ donde especificaremos en qué consisten nuestros nuevos bloques. En nuestro caso llamamos al archivo "movement_blocks.json".
3. Por último modificamos el archivo que acabamos de crear especificando qué mensaje va a contener nuestro bloque, de qué tipo son sus argumentos y los valores permitidos para los mismos. La configuración resultante para el bloque de movimientos hacia adelante y hacia atrás se muestra en el Código 2.

[51] https://developers.google.com/blockly/guides/create-custom-blocks/define-blocks

```
{"type": "movement_move",
    "message0": "Mover hacia %1 por %2 segundos",
    "args0": [
      {"type": "field_dropdown",
        "name": "direction",
        "options": [
          ["Adelante","forward"],["Atras","backward"]]
      },{
        "type": "field_number",
        "name": "time",
        "value": 0,
        "precision": 1
      }],
    "inputsInline": false,
    "previousStatement": null,
    "nextStatement": null,
    "colour": 290,
    "tooltip": ""}]
```

Código 2 - Configuración en Blockly del bloque de movimiento

4.3.3.5 Configuración de los generadores de código

Como ya mencionamos, el método getCodeGenerationCallback() es llamado cuando finaliza la generación de código a partir de la secuencia de bloques existentes. Para los bloques predeterminados los archivos generadores de código ya están especificados, pero como nosotros agregamos bloques personalizados, debemos especificar cómo se generará el código para los mismos. Para esto debemos hacer lo siguiente:

1. Creamos un nuevo archivo en assets/dropsy/ llamado generators.js
2. Como en nuestra aplicación generamos código JavaScript, editamos el archivo creado especificando cómo generar código JavaScript a partir de los nuevos bloques. Esto puede verse en el Código 3.

```
// Extensions to Blockly's language and JavaScript generator.
Blockly.JavaScript['movement_turn'] = function(block) {
 // Generate JavaScript for moving turning left or right.
   var value = block.getFieldValue('time');
   return 'Robot.' + block.getFieldValue('direction') + '(0,' + value
+ ');\n';
};

Blockly.JavaScript['movement_move'] = function(block) {
 // Generate JavaScript for moving forward or backward.
 var value = block.getFieldValue('time');
 return 'Robot.' + block.getFieldValue('direction') +
     '(0,' + value + ');\n';
};
```

Código 3 - Configuración a JavaScript de generador de código Blockly

3. Por último, modificamos nuestro ProjectActivity para que utilice el nuevo generador de código. Para esto modificamos el valor de la variable estática JAVASCRIPT_GENERATORS como se muestra en el Código 4:

```
private static final List<String> JAVASCRIPT_GENERATORS = Arrays.asList(
            new String[]{ "dropsy/generators.js"}
);
```

Código 4 - Configuración de categoría de bloques en Blockly para generar código JavaScript

4.3.4 Duktape

4.3.4.1 Introducción

Es una librería muy importante dentro de nuestra aplicación, ya que es la que toma nuestro código JavaScript y lo interpreta de tal manera que nos permite comunicarnos con el servidor XRemoteBot para enviarle las instrucciones marcadas por el código generado.

4.3.4.2 Instalación

Para instalar esta librería alcanza con agregar la dependencia al archivo build.gradle de nuestra aplicación como se muestra en el Código 5. Luego Gradle se encarga de descargar la librería.

```
compile 'com.squareup.duktape:duktape-android:0.9.5'
```

Código 5 - Configuración de gradle para incluir la librería Duktape

4.3.4.3 Integración con la aplicación

En nuestro caso vamos a utilizar la librería luego de que es generado el código, por lo tanto lo hacemos en el método onFinishCodeGeneration() de la clase CodeInterpretation.
Para interpretar el código generado basta con crear una instancia de la clase Duktape, a la que luego le enviamos el mensaje evaluate() con el código generado como parámetro. El fragmento de Código 6 muestra el uso de la librería Duktape.

```
Duktape duktape = Duktape.create();
duktape.evaluate(generatedCode);
```

Código 6 - Creación de la instancia de Duktape y evaluación del código generado

Pero como en nuestro caso necesitamos que cuando el código es interpretado utilice clases pertenecientes a nuestro proyecto Android, debemos especificarlas antes de realizar la evaluación utilizando el mensaje bind. Para esto creamos una nueva instancia de la clase RobotImpl, que implementa la interfaz Robot y la enviamos como parámetro en el comando bind. La interfaz Robot define todos los métodos que necesitan ser llamados desde el código JavaScript.
El código resultante se muestra en el Código 6:

```
Duktape duktape = Duktape.create();
RobotImpl robot = new RobotImpl();
duktape.bind("Robot", Robot.class, robot);
duktape.evaluate(generatedCode);
```

Código 7 - Interpretación de código JavaScript utilizando Duktape

La librería Duktape se encarga de interpretar el código JavaScript contenido en la variable generatedCode y, en caso de encontrar alguna referencia a la variable Robot, la redirige a la variable robot que pasamos como parámetro. Con esto básicamente logramos que, al encontrarnos por ejemplo con el envío de una instrucción de movimiento dentro del código JavaScript generado, sea el mismo Duktape el encargado de comunicarse con el servidor para

que lo ejecute el robot. Para poder lograr esto la interfaz Robot define los métodos mostrados en el Código 8.

```
public void left(int speed, int time);
public void right(int speed, int time);
public void forward(int speed, int time);
public void backward(int speed, int time);
```

Código 8 - Métodos implementados por la interfaz Robot

En la Figura 13 se muestra esquemáticamente la interacción entre las distintas librerías del proyecto DROPSY.

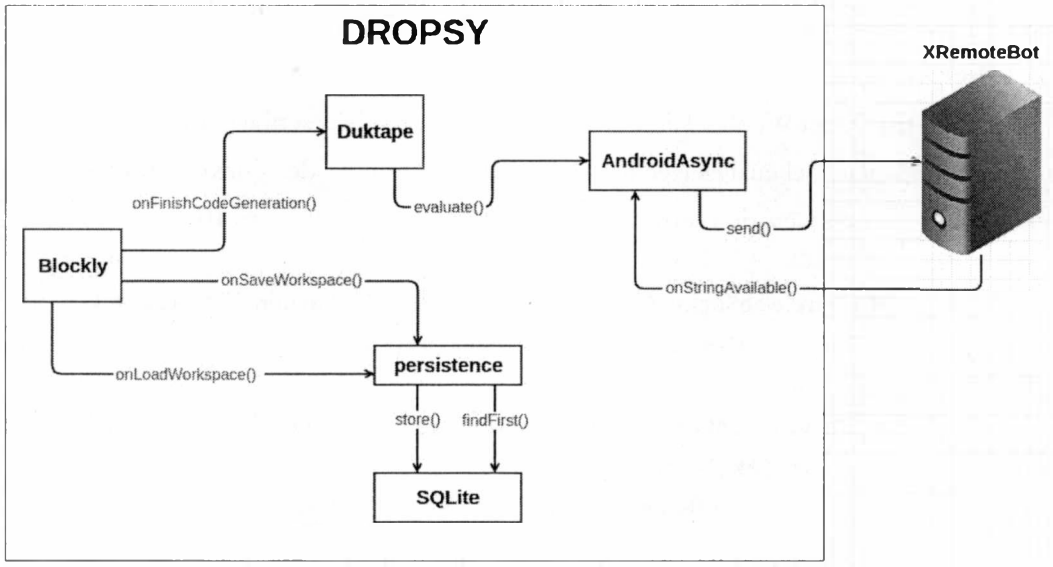

Figura 13 - Interacción entre librerías Android utilizadas en el desarrollo de DROPSY

4.3.5 AndroidAsync

Esta librería[52] fue la que nos permitió comunicarnos con el servidor mediante la utilización de WebSockets, a los cuales podemos conectarnos y enviar mensajes que serán interpretados por el servidor.

[52] https://github.com/koush/AndroidAsync

4.3.5.1 Instalación

Para instalar esta librería alcanza con agregar la dependencia al archivo build.gradle de nuestra aplicación. Luego gradle se encarga de descargar la librería.

```
compile 'com.koushikdutta.async:androidasync:2.1.9'
```

Código 9 - Configuración de gradle para incluir la librería AndroidAsync

4.3.5.2 Utilización

La clase encargada de utilizar la librería AndroidSync es la clase RobotManager, la cual es utilizada para abstraer a los servicios de la comunicación con el servidor.

Primero que nada, la clase RobotManager posee un método connect(String url), encargado de conectarse con el socket que se encuentra en la URL pasada como parámetro. Para esto se realizan los siguientes pasos:

1. Primero se crea una nueva instancia de una clase que hereda de AsyncHttpClient.WebSocketConnectCallback y se sobreescribe el método onCompleted(), el cual es ejecutado al finalizar el intento de conexión, pudiendo el mismo ser satisfactorio o erróneo. Esto se muestra en el Código 10:

```
AsyncHttpClient.WebSocketConnectCallback mWebSocketConnectCallback = new
AsyncHttpClient.WebSocketConnectCallback() {
        @Override
        public void onCompleted(Exception ex, WebSocket webSocket) {
                if (ex != null){
                        //Manejo de excepción
                }
                ws = webSocket; //Guardamos el websocket activo
                //Implementación
        }
```

Código 10 - Fragmento de código del método connect(url) de RobotManager, encargado de la creación de una instancia de WebSocketconnectCallback

2. Se obtiene la instancia activa de la clase AsyncHttpClient llamando a AsyncHttpClient.getDefaultInstance() y se guarda en la variable mAsyncHttpClient, tal como se muestra en el Código 11.

```
AsyncHttpClient mAsyncHttpClient = AsyncHttpClient.getDefaultInstance();
```

Código 11 - Obtención de la instancia activa de AsyncHttpClient

3. Realizamos el intento de conexión llamando al método webSocket() de mAsyncHttpClient como se muestra en el Código 12.

```
mAsyncHttpClient.websocket(url, null, mWebSocketConnectCallback);
```

Código 12 - Creación de WebSocket utilizando AndroidAsync

Una vez que la conexión con el WebSocket está establecida, podemos enviar mensajes al mismo, los cuales serán interpretados por el servidor. En nuestra aplicación utilizamos el formato JSON tanto para el envío como para la recepción de mensajes, suponiendo que la instancia activa de la clase WebSocket está contenida en la variable ws, el método utilizado para enviar mensajes al mismo es el que se muestra en el Código 13.

```
ws.send(json_string);
```

Código 13 - Envío de mensajes al WebSocket utilizando la librería AndroidAsync

Además, es posible configurar una clase que herede de WebSocket.StringCallback, la cual utilizaremos para saber cuando el servidor envie mensajes al WebSocket.
Para esto debemos usar el método setStringCallback() del WebSocket, el cual recibe como parámetro la nueva clase de StringCallBack, sobreescribiendo método onStringAvailable(String s), el cual será llamado cada vez que el servidor envíe un mensaje al mismo.

```
ws.setStringCallback(new WebSocket.StringCallback() {
    @Override
    public void onStringAvailable(String s) {
        //Implementación
    }
});
```

Código 14 - Configuración de WebSocket de la librería AndroidAsync

Este método puede ser llamado directamente en el método onCompleted() del WebSocketCallback, en caso de que la conexión sea exitosa.

Otro método importante es el método setClosedCallback(), el cual permite realizar acciones cuando el WebSocket es cerrado por alguna razón, ya sea por algún problema de red o un cierre intencional desde el lado del servidor. Este método recibe como parámetro una instancia de una clase que implemente la interfaz CompletedCallback, la cual debe implementar el método onCompleted():

```
ws.setClosedCallback(new CompletedCallback() {
    @Override
    public void onCompleted(Exception ex) {
        disconnect();
    }
});
```

Código 15 - Configuración de WebSocket de la librería AndroidAsync

Por último, al finalizar la utilización del WebSocket podemos llamar al método ws.close() para desconectarnos del mismo.

4.3.6 Persistence

Gracias a su mapeo objeto-relacional, esta librería nos facilita la interacción con bases de datos, permitiéndonos crear, modificar, listar, obtener y eliminar objetos en una base de datos de una manera muy simple, sin necesidad de interactuar con queries y cursores.

4.3.6.1 Instalación

Para instalar esta librería primero agregamos la dependencia al archivo build.gradle de nuestra aplicación como se muestra en el Código 16. Luego gradle se encarga de descargar la librería.

```
compile 'com.codeslap:persistence:0.9.24'
```

Código 16 - Configuración de gradle para incluir la librería Persistence

Luego debemos modificar nuestra clase App, que extiende de Application y se encuentra definida en AndroidManifest.xml, agregando el código mostrado en el Código 17. Este código inicializa le indica a la librería cuales son las clases que almacenaremos en la base de datos.

```
DatabaseSpec database = PersistenceConfig.registerSpec(version_db);
database.match(Project.class);
```

Código 17 - Configuración de las clases a utilizar con la librería Persistence

Con el fragmento de código del Código 17 inicializamos la base de datos. El método match() recibe como parámetros las clases para las cuales vamos a utilizar la misma. La librería automáticamente crea las tablas de Sqlite[53] para las clases recibidas como parámetros, las cuales nos permitirán almacenar y manipular objetos pertenecientes a esas clases.

4.3.6.2 Utilización

Una vez configurada la base de datos podemos hacer uso de la misma cuando la necesitemos, para esto se obtiene una implementación de la clase SqlAdapter, por ejemplo como se muestra en el Código 18.

```
SqlAdapter adapter = Persistence.getAdapter(context);
```

Código 18 - Obtención de la instancia de SqlAdapter

Luego podemos utilizar ese adapter para realizar las operaciones que necesitemos, algunas de ellas son:

- Insertar/Actualizar: nos permite guardar un objeto. En caso de que el Id del objeto ya exista, se actualiza. Se muestra un ejemplo en el código 19.

```
Project proyecto = new Project();
adapter.store(proyecto);
```

Código 19 - Inserción de un objeto de la clase Project en la base de datos

- Obtener: nos permite obtener un objeto de una clase según alguna de sus propiedades. Se muestra un ejemplo en el Código 20.

[53] "SQLite" 11 Nov. 2016. <https://sqlite.org/>

```
Project projecto = new Project();
projecto.setNombre("Nuevo")
adapter.findFirst(projecto);
```

Código 20 - Obtención de un objeto de la clase Project desde la base de datos

- Eliminar: nos permite eliminar un objeto de una clase según alguna de sus propiedades o todas. Se muestra un ejemplo en el Código 21.

```
Project projectoAEliminar;
adapter.delete(projectoAEliminar);
```

Código 21 - Eliminación de un objeto de la clase Project de la base de datos

- Listar: nos permite listar todos los objetos de una clase según alguna de sus propiedades, o según su clase. Se muestra un ejemplo en el Código 22.

```
adapter.findAll(new Project());
```

Código 22 - Obtención de todos los objetos de la clase Project almacenados en la base de datos

4.3.7 DROPSY en Google Play

Para un correcto manejo de versiones y disponibilidad ininterrumpida, la aplicación desarrollada fue subida al servicio Google Play, por lo que cualquier usuario puede descargarla desde su tablet utilizando la aplicación Play Store, disponible en todos los dispositivos Android.
El link de la aplicacion es el siguiente:
https://play.google.com/store/apps/details?id=com.fuentesfernandez.dropsy

4.3.8 DROPSY en GitHub

DROPSY es un proyecto de código abierto y su código fuente se puede encontrar en GitHub en el siguiente link:
https://github.com/dropsy-unlp/dropsy-android-client

Capítulo 5

Streaming de video

5.1 Streaming en XRemoteBot

El servidor de XRemoteBot incluye una serie de paquetes que permiten la realización de streaming de video a los clientes que controlan a los robots.

La versión original de XRemoteBot utiliza las siguientes tecnologías:

- avconv[54] para capturar y codificar el video.
- NodeJS[55] para ejecutar el script que transmite el video.
- El paquete ws de npm[56] (dependencia de este último).

El video se codifica utilizando avconv como encoder en formato MPEG-1[57] y luego se envía a un servidor escrito en NodeJS que se encarga de servir el streaming utilizando Websockets como transporte. En la Figura 14 se puede ver cómo interactúan los componentes del streaming.

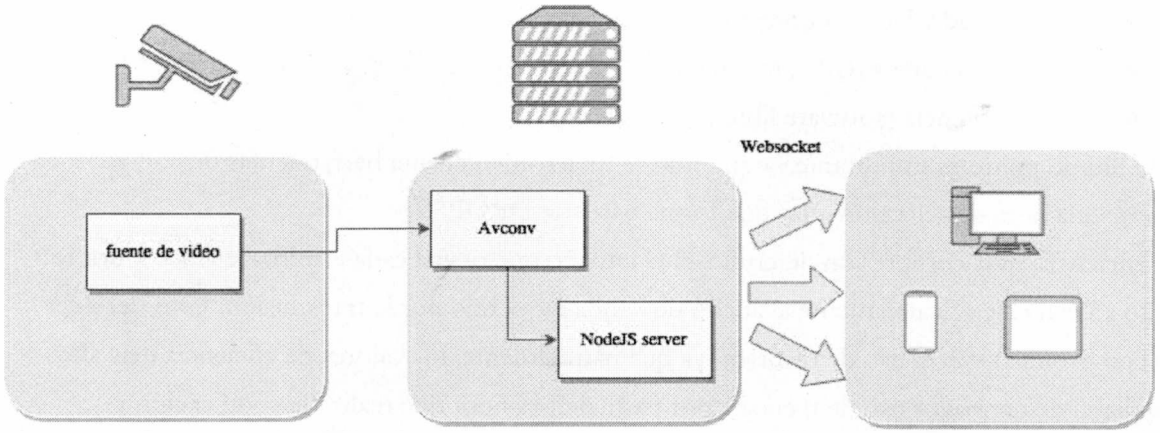

Figura 14 - Diagrama de funcionamiento del streaming en XRemoteBot

[54] "Libav documentation : avconv." 2011. 4 Sep. 2016 <https://libav.org/avconv.html>
[55] "Node.js." 2014. 4 Sep. 2016 <https://nodejs.org/>
[56] "ws - npm." 2014. 4 Sep. 2016 <https://www.npmjs.com/package/ws>
[57] "MPEG-1 Video Coding Standard." 2011. 4 Sep. 2016 <http://coitweb.uncc.edu/~ifan/mpeg.pdf>

Esta configuración resultó muy performante en entornos locales, obteniéndose tiempos de latencia despreciables, en el orden de los milisegundos. Sin embargo al ser expuesto como un servicio remoto y consumido a través de Internet, la solución comenzó a mostrar algunos problemas de estabilidad y rendimiento, debido a las limitaciones de ancho de banda y condiciones de la red subyacente. A estos inconvenientes se suman los problemas de compatibilidad con clientes Android para tablets y celulares, que serán los principales consumidores del servicio.

A continuación se analizarán distintas alternativas para mejorar el funcionamiento del streaming de video del robot; una parte muy importante de conjunto de servicios de DROPSY, entre cuyos objetivos se encuentra el uso de la herramienta por parte de los clientes remotos que acceden a través de Internet.

5.2 Análisis de tecnologías de streaming existentes

5.2.1 Encoders

Actualmente existen solamente unas cuantas herramientas para codificar y transmitir video, ya sea desde fuentes estáticas (archivos de video), o de fuentes en vivo, como por ejemplo una cámara web. Los factores a la hora de considerar una herramienta de *encoding* son varios, entre los que encontramos:

- La eficiencia en la compresión.
- La velocidad y latencia obtenidas.
- El uso de recursos (CPU, memoria).
- Tipo de licencia (software libre / propietario)

Este último punto es importante ya que sólo se tuvieron en cuenta herramientas de código abierto a la hora de seleccionar las que forma parte de DROPSY.

La eficiencia en la compresión determinará el tamaño que ocupa cada cuadro de video y por lo tanto afectará significativamente al ancho de banda requerido por la transmisión. Esto debe ser un compromiso con el uso de recursos, ya que naturalmente, los valores de eficiencia más altos requieren de un mayor uso de recursos por parte del servidor que realiza la codificación y compresión de la señal de video. A su vez, una mayor eficiencia de compresión resultará en tiempos de latencia más altos, como consecuencia del tiempo que toma al procesador llevar a cabo proceso de *encoding*, que puede llegar a ser muy costoso dependiendo del formato y configuraciones.

Por lo tanto, fue de vital importancia seleccionar correctamente la herramienta de *encoding*, así como los codecs[58] que utiliza y los parámetros de configuración para lograr un comportamiento estable, con calidad de imagen aceptable y una latencia dentro de los valores requeridos por DROPSY.

5.2.1.1 FFmpeg

FFmpeg es una colección de librerías y herramientas de código abierto para procesar contenido multimedia como audio, video, subtítulos y metadatos.
Está compuesta por las siguientes librerías:

- libavcodec: provee la implementación de una gran variedad de codecs.
- libavformat: implementa protocolos de streaming , formatos de contenedores[59] y funciones básicas de I/O.
- libavutil: incluye herramientas de hashing, decompresores y otras utilidades generales.
- libavfilter: provee una forma de alterar el video y audio decodificados a través de una serie de filtros.
- libavdevice: provee una capa de abstracción para acceder, capturar y reproducir dispositivos.
- libswresample: implementa rutinas de mezcla y resampling de audio.
- libswscale: implementa rutinas de conversión de color y escalado.

FFmpeg incluye además FFserver; un servidor de streaming para audio y video. Soporta varios feeds simultáneamente, streaming desde archivos y también transmisiones en vivo. Sin embargo al momento de desarrollar esta tesina se encuentra en proceso de ser dado de baja del proyecto FFmpeg y no será incluido en posteriores versiones del aplicativo.

5.2.1.2 Libav

Libav es un proyecto de software libre nacido como una rama de FFmpeg en 2011, que produce librerías y programas para manejar contenido multimedia. Fue creado por ex desarrolladores del proyecto FFmpeg, que decidieron separarse del proyecto original debido a diferencias de management y objetivos: por un lado FFmpeg apostaba por un desarrollo rápido, favoreciendo la incorporación de nuevas características. Libav en cambio buscaba mejorar el estado actual del código y desarrollar mejores APIs.

[58] códec o formato de codificación. Descrito en 5.2.3
[59] "Fundamentos y evolución de la multimedia » contenedor de vídeo." 2013. 4 Sep. 2016 <http://multimedia.uoc.edu/blogs/fem/es/etiqueta/contenedor-de-video/>

Inicialmente esta división pareció favorecer a Libav, que fue incorporado por defecto en algunas de las distribuciones de Linux más importantes (en particular Debian[60]); sin embargo, para la versión 9.0 de Debian se optó por volver a FFmpeg dejando de lado la inclusión de Libav.

5.2.1.3 VLC Media Player como encoder

VLC es un reproductor multimedia y framework de código abierto y multiplataforma que reproduce la mayoría de los contenidos multimedia así como DVD, Audio CDs, VCD y varios protocolos de streaming.

En el apartado de streaming VLC ofrece una potente herramienta por línea de comando[61] que soporta transcoding en múltiples formatos y codecs y que funciona además como servidor; ofreciendo de esta manera una solución integrada para realizar streaming desde y hacia distintas fuentes.

VLC ofrece la posibilidad de configurar un servidor de streaming mediante su UI, de manera intuitiva y sin por ello perder flexibilidad. Además la configuración visual permite extraer una cadena de texto con la que fácilmente podemos trasladar los parámetros utilizados a la línea de comando y de esta manera funcionar en ambientes sin interfaz gráfica.

5.2.2 Protocolos de red

Cuando realizamos streaming de una fuente multimedia, esperamos poder distribuir los contenidos y hacerlos accesibles de manera remota; ya sea libremente o restringido a un público en particular. Además de contar con un formato de codificación y uno de empaquetado, una fuente de video y el hardware que realice el trabajo pesado; también necesitamos de un protocolo de transporte que haga posible la publicación del contenido multimedia.

Describiremos los protocolos de red para streaming más comunes y algunos de reciente aparición, así como una breve reseña de sus características principales. Adicionalmente es importante realizar una breve comparación entre los protocolos de transporte subyacentes previo a la introducción de los empleados específicamente para realizar streaming.

Internet tiene dos protocolos principales en la capa de transporte, uno orientado a la conexión y otro no orientado a la conexión. El protocolo no orientado a la conexión es el UDP y el orientado a la conexión es el TCP.

[60] "Debian" 2016. <https://www.debian.org/index.es.html>

[61] "VLC command-line help - VideoLAN Wiki." 2011. 11 Sep. 2016 <https://wiki.videolan.org/VLC_command-line_help>

- UDP[62] (protocolo de datagramas de usuario): se trata de un protocolo de transporte no orientado a la conexión. Este protocolo proporciona una forma para que las aplicaciones envíen datagramas IP encapsulados sin tener una conexión.
- TCP[63] (protocolo de control de transmisión): se diseñó para proporcionar un flujo de bytes confiable de extremo a extremo a través de una red no confiable. Para lograr esta confiabilidad, implementa mecanismos de retransmisión, buffers y reordenamiento de paquetes.

En cuanto a las características que posee cada uno y que afectarían al momento de realizar streaming podemos mencionar:

- UDP
  - Menos sobrecarga en los encabezados de cada paquete.
  - Menor latencia y velocidad de transmisión.
  - Posibilidad de hacer multicast, ahorrando ancho de banda.
- TCP
  - Mejor control en las transmisiones.
  - Fácilmente adaptable al uso de firewalls.

5.2.2.1 HTTP

Existen varios protocolos que permiten realizar streaming a través de la red. Podríamos pensar en utilizar HTTP[64] por ser el protocolo utilizado por los navegadores web. Sin embargo el uso de HTTP para realizar streaming es relativamente reciente, debido en parte a algunos problemas que fue necesario sortear antes de obtener buenos resultados y también gracias al surgimiento de HTML5[65], que impulsó la maduración de nuevas formas de transmisión basadas en HTTP.
La principal desventaja de utilizar HTTP para realizar streaming es que este último utiliza TCP como protocolo subyacente; que tiene un rendimiento en principio inferior a UDP debido a los controles adicionales que realiza para garantizar la llegada de todos los paquetes intactos y en el orden en que fueron enviados. Sin embargo, en el caso del streaming multimedia, este esfuerzo extra no es tan importante ya que los errores de transmisión y paquetes perdidos pasan casi desapercibidos en una transmisión y en cambio; sí es muy importante la velocidad y el constante flujo de datos para lograr una buena experiencia de usuario.
Por otro lado, el uso de HTTP tiene ciertos beneficios de gran importancia, como por ejemplo una compatibilidad casi universal (sólo requiere de un navegador) e inmunidad al uso de firewalls (todo el tráfico viaja por el puerto 80 en su configuración por defecto).

---

[62] "Transmission Control Protocol" 1981. 29 Oct 2016 <https://tools.ietf.org/html/rfc793>
[63] "RFC 768 (UDP) 1980. 29 Oct 1980 <https://www.ietf.org/rfc/rfc768.txt>
[64] "HTTP: HyperText Transfer Protocol" 1999. 29 Oct 2016 <https://tools.ietf.org/html/rfc2616>
[65] "HTML5 Video - W3Schools." 2012. 24 Sep. 2016 <http://www.w3schools.com/html/html5_video.asp>

5.2.2.2 RTP

El protocolo de trasporte en tiempo real RTP (Real-time Transport Protocol) definido por la RFC 3550[66] provee funciones de transporte para aplicaciones que realizan transmisión de datos en tiempo real, como video audio o información de simulaciones, a través de redes unicast o multicast. Se usa extensivamente en sistemas de comunicación y entretenimiento que incluyen streaming multimedia, tales como telefonía, videoconferencia, servicios de televisión y tecnologías push-to-talk[67].

La capa de transporte de datos de RTP es aumentada por un protocolo de control, RTCP, que permite el monitoreo de la transmisión de datos y provee funciones básicas de control e identificación.

Al momento de iniciar transmisión utilizando RTP se define una sesión por cada flujo multimedia. Una sesión consiste de una dirección IP con un par de puertos para RTP y RTCP. Por ejemplo, los flujos de video y audio utilizan sesiones separadas, permitiendo al receptor omitir alguno de los flujos. Los puertos que forman una sesión son negociados utilizando otros protocolos como RTSP[68] y SIP[69]. Típicamente los flujos de datos se envían utilizando UDP como protocolo subyacente. Sin embargo RTP también ofrece la posibilidad de enviar todo el tráfico a través de TCP intercalando los paquetes de datos y control. Si bien esta última posibilidad aporta flexibilidad; también se pierden algunas de las ventajas propias de UDP y RTP.

5.2.2.3 HLS

HLS es la abreviatura de HTTP Live Streaming[70], un protocolo de streaming adaptativo creado por Apple para comunicarse con dispositivos iOS, Apple TV y Macs ejecutando OSX Snow Leopard o posteriores. HLS puede distribuir archivos en tiempo real o bajo demanda y es la única tecnología disponible para realizar streaming en dispositivos Apple, que conforman un público cada vez más importante para los distribuidores de contenido mediante streaming. Gracias a su creciente popularidad es soportado por gran cantidad de proveedores de servidores de streaming como Adobe, Microsoft, RealNetworks y Wowza.

La tecnología adaptativa[71] de HLS permite que sea utilizado con buenos resultados en una gran variedad de dispositivos; desde los más modestos hasta los de tope de gama. Esta flexibilidad también se traslada a la red sobre la que se realiza la transmisión, pudiendo funcionar en entornos con anchos de banda reducidos.

---

[66] Schulzrinne, H. "RFC 3550 - IETF." 2003. <https://www.ietf.org/rfc/rfc3550.txt>

[67] "Push to Talk" 2016 . 29 Oct 2016 <https://es.wikipedia.org/wiki/Pulsa_y_habla>

[68] Schulzrinne, H. "RFC 2326 "Real Time Streaming Protocol (RTSP)" - IETF." 1998. <https://www.ietf.org/rfc/rfc2326.txt>

[69] Rosenberg, J. "SIP: Session Initiation Protocol (RFC 3261) - IETF." 2002. <https://www.ietf.org/rfc/rfc3261.txt>

[70] "HTTP Live Streaming" 2016. 19 Oct 2016 <https://developer.apple.com/streaming/>

[71] "Adaptative bitrate streaming" 2016. 30 Oct 2016 <https://en.wikipedia.org/wiki/Adaptive_bitrate_streaming>

A pesar del creciente interés en HLS, aún existen dificultades para su adopción fuera del ecosistema Apple. El ejemplo más claro es el caso de Google, que ha incluido soporte para la tecnología de Apple en para su plataforma Android (a partir de 3.0), aunque al día de la fecha siguen existiendo problemas que impiden su normal funcionamiento.

5.2.3 Formatos de codificación y empaquetado

Una transmisión multimedia suele estar compuesta por varios streams, cada uno transportando un tipo de contenido distinto: audio, video o subtítulos. También pueden transportar distintas versiones de un mismo tipo, como por ejemplo subtítulos o pistas de audio en distintos idiomas. Cada stream simple se encuentra codificado en un formato específico mediante un algoritmo llamado codec. Como ejemplos de codecs de vídeo tenemos h264, mp4, mjpeg y como codecs de audio a mp3, aac, ogg. Estos componentes luego se entremezclan o empaquetan en un nuevo formato que funciona como contenedor de los anteriores (mov, avi, mpeg, flv, vob) y se encarga de mantener la sincronía entre los contenidos. Este nuevo stream entremezclado queda listo para guardarse como archivo o bien para ser transmitido. La Figura 15 ilustra este proceso.

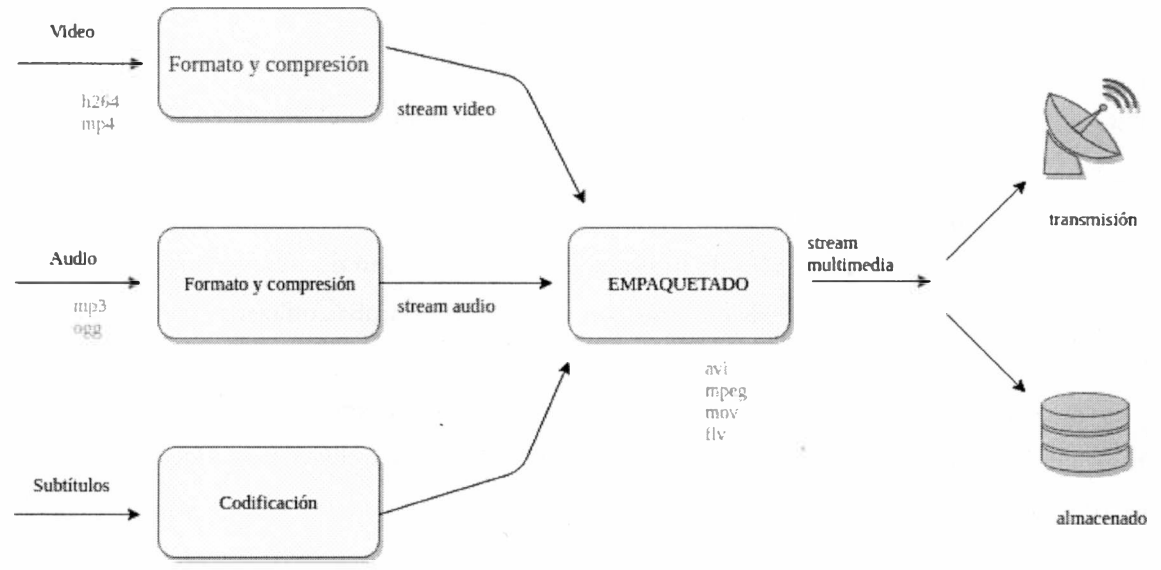

Figura 15 - Proceso de codificación, empaquetado y transmisión

Existe un gran número de codecs y contenedores; algunos recientes y otros que se han popularizado hasta ser utilizados casi universalmente. Mencionaremos sólo los tienen mayor incidencia en el marco de este trabajo.

5.2.3.1 H264

H.264[72] o MPEG-4 AVC, es un estándar de compresión de video orientado a bloques con compensación de movimiento empleado comúnmente en la grabación, compresión y distribución de contenidos de video. Fue desarrollado conjuntamente por el ITU-T Video Coding Experts Group (VCEG)[73] y el ISO/IEC Moving Picture Experts Group (MPEG)[74].
La intención del proyecto H.264/AVC fue la de crear un estándar capaz de proporcionar una buena calidad de imagen con tasas binarias notablemente inferiores a los estándares previos (MPEG-2[75], H.263[76] o MPEG-4 parte 2[77]), sin necesidad de incrementar la complejidad de su diseño.
Otro objetivo adicional fue el de proveer suficiente flexibilidad para permitir que el estándar sea aplicado en una gran variedad de aplicaciones sobre distintas condiciones de red, incluyendo tasas altas y bajas de bits y de resoluciones de video.
H.264 es bien conocido por ser uno de los estándares de codificación de los discos Blu-ray[78]. Además es ampliamente usado por las plataformas de video de Internet como Vimeo, Youtube y iTunes, además de Flash Player de Adobe y Silverlight de Microsoft.
En cuanto a su licencia, MPEG LA permite el uso libre de las tecnologías H.264 para realizar streaming, siempre que sea sin fines de lucro.

5.2.3.2 MP4

MPEG-4 Parte 14[79] o simplemente MP4 es un formato contenedor multimedia comúnmente usado para almacenar audio y video, pero además puede usarse para otros tipos de datos como subtítulos e imágenes estáticas. Como la mayoría de los formatos contenedores modernos, puede ser empleado en el streaming a través de Internet. La única extensión de archivo oficial para MPEG-4 Parte 14 es .mp4, pero existen otras como m4a y m4p, ambas utilizadas para audio únicamente.
Las pistas de audio y video se embeben en los archivos MPEG-4 usando streams privados. Una pista separada se utiliza para incluir los metadatos del stream en el archivo. Los codecs más utilizados son:

---

[72] https://www.itu.int/rec/T-REC-H.264
[73] "Video Coding Experts Group" 2016. 30 Oct 2016 <http://www.itu.int/en/ITU-T/studygroups/2013-2016/16/Pages/video/vceg.aspx>
[74] "Moving Picture Experts Group" 2016. 30 Oct 2016 <http://mpeg.chiariglione.org/>
[75] "MPEG-2 Standard" 2000. 30 Oct 2016 <http://mpeg.chiariglione.org/standards/mpeg-2/video>
[76] "H.263 Video coding for low bit rate communication" 2005. 30 Oct 2016 <https://www.itu.int/rec/T-REC-H.263-200501-I/en>
[77] "MPEG-4 part 2 Video" 1998. 30 Oct 2016 <http://mpeg.chiariglione.org/standards/mpeg-4/video>
[78] "Blu-ray disc" 2012. 30 Oct 2016 <http://www.blu-raydisc.com/en/Technical/TechnicalWhitePapers/General.aspx>
[79] "MPEG-4 Part 14" 2003. 30 Oct 2016 <http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38538>

- Video: MPEG-4 Parte 10 (H264) y MPEG-4 Parte 2
- Audio: Advanced Audio Coding [80](AAC). Además otros que forman parte de MPEG-4 Parte 3, como Audio Lossless Coding (ALS), Scalable Lossless Coding (SLS) MP3.
- Subtítulos: MPEG-4 Timed Text (también conocido como 3GPP Timed Text).

5.2.3.3 FLV

FLV[81] es la abreviatura de Flash video, un formato contenedor utilizado para distribuir contenido multimedia utilizando Adobe Flash Player 6 o superiores. Los contenidos Flash Video pueden ser embebidos dentro de archivos SWF. Existen dos formatos distintos conocidos como Flash Video: FLV y F4v. El audio y video dentro de los archivos FLV se codifican de la misma manera que dentro de los archivos SWF. El formato F4V está basado en el formato base ISO para multimedia y se encuentra disponible a partir de Flash Player 9 actualización 3. Al comienzo de la década del 2000 Flash Video era el estándar de facto para el streaming de video en la web (a través de RTMP). Ejemplos de uso incluyen Hulu, VEVO, Yahoo! Video, metacafe y numerosos proveedores de noticias.
La mayoría de los contenidos FLV se encuentran codificados en los formatos Sorenson Spark[82] o VP6[83]. Las últimas versiones de Flash Player soportan además H.264 para video y HE-AAC para audio.

5.2.3.4 WebM en conjunto con VP8 y VP9

WebM[84] es un formato multimedia abierto y libre con el objetivo principal de ofrecer una alternativa de compresión de video libre de regalías para ser utilizado en los tags HTML5. El desarrollo se encuentra respaldado por Google y el software se distribuye bajo una licencia permisiva similar a la licencia BSD[85]. Está compuesto por el codec de vídeo VP8 (desarrollado originalmente por On2 Technologies) y el codec de audio Vorbis dentro de un contenedor multimedia Matroska. En 2014 fue actualizado para incorporar VP9 y Opus audio.
Fue anunciado bajo el marco de la conferencia Google I/O en mayo de 2010 y cuenta con contribuciones y apoyo oficial de empresas como Mozilla, Opera 5 y Google además de un gran número editores y fabricantes de software y hardware; en un esfuerzo combinado para utilizar VP8 como el formato multimedia estándar en el lenguaje web HTML5. El codec se puede usar en la versión de YouTube HTML5 que trae esa opción activada por defecto.

---

[80]"Advanced Audio Coding (AAC)" 2003. 30 Oct 2016 <http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=38509>
[81]"FLV Flash Video" 2008. 30 Oct 2016 <https://www.adobe.com/content/dam/Adobe/en/devnet/flv/pdfs/video_file_format_spec_v10.pdf>
[82]"Sorenson Spark" 2014. 30 Oct 2016 <http://www.digitalpreservation.gov/formats/fdd/fdd000066.shtml>
[83] "VP6 codec" 2011. 30 Oct 2016 <https://wiki.multimedia.cx/index.php?title=On2_VP6>
[84] "WebM Supporters" 2016. 30 Oct 2016 <https://www.webmproject.org/about/supporters/>
[85] "BSD" 2013. 30 Oct 2016 <https://www.freebsd.org/doc/es/articles/explaining-bsd/article.html>

Los navegadores que ofrecen soporte nativo para WebM son:

- Mozilla Firefox 4 y posteriores
- Opera 10.60 en adelante
- Google Chrome desde la versión 6
- Microsoft Internet Explorer 9 en adelante (requiere componentes WebM MF)
- Microsoft Edge

En cuanto a los reproductores; VLC media player, MPlayer y K-Multimedia Player poseen soporte nativo para WebM. FFmpeg puede codificar y decodificar videos VP8 cuando se lo compila con soporte para libvpx, así como multiplexar y demultiplexar archivos WebM. Android se encuentra habilitado para WebM[86] a partir de su versión 2.3 (Gingerbread). La decodificación utilizando VP9 fue incluida desde la versión 4.4 (KitKat).

5.2.4 Streaming Servers

Las herramientas de captura y codificación de vídeo que hemos visto hasta ahora hacen un gran trabajo en entregarnos un flujo de video/audio listo para ser reproducido con una aplicación cliente compatible o almacenado en el disco para su posterior uso. Sin embargo, en el caso de un flujo de video en vivo, como es nuestro caso, necesitamos poder acceder a la transmisión desde otros clientes remotos, a través de Internet y desde cualquier parte del mundo.
Para lograr este objetivo existen los streaming servers. Su función principal es recibir los pedidos de los clientes, decidir si los atenderá o no y finalmente entregarles las imágenes en tiempo real. En general, un servidor de streaming debería ser capaz de soportar la mayor parte de los protocolos de red; así como otras cuestiones como codecs de audio y video y sistemas operativos. En este caso, lo más importante es la gama de protocolos que soporta, porque esto definirá en qué dispositivos se podrá reproducir el video.

5.2.4.1 ffserver

Como se mencionó en secciones anteriores, FFserver[87] es un servidor de streaming para audio y video, incluido como parte del proyecto FFmpeg. Soporta varios flujos de video en simultáneo, streaming desde archivos y time shifting, lo que significa que es posible retroceder un video en vivo (siempre que se especifique un espacio de almacenamiento temporal lo suficientemente grande).
La configuración de FFserver se realiza por medio de un archivo de configuración que se lee al iniciar el server. Una instancia de FFserver puede recibir archivos pregrabados o streams de

[86] "Android supported media formats" 2016. 30 Oct 2016 <https://developer.android.com/guide/appendix/media-formats.html>
[87] "ffserver" 2016. 30 Oct 2016 <https://trac.ffmpeg.org/wiki/ffserver>

FFmpeg y transmitirlos utilizando alguno de los protocolos soportados (RTP/RSTP/HTTP). Desde el archivo de configuración se determina el puerto en el cual FFserver esperará recibir los streams.

FFserver es una herramienta simple de usar y con un propósito muy claro. Sin embargo es importante mencionar que al momento de escribir este trabajo el proyecto FFmpeg anunció que dejará de incluir FFserver como parte de su paquete de aplicaciones[88]. Asimismo dejará de darle soporte. Entre las causas para dejar de lado la herramienta se menciona la dificultad en el mantenimiento de las APIs internas y algunos problemas de estabilidad.

5.2.4.2 stream-m

Stream-m nace como una solución open source para realizar streaming de video en tiempo real directamente a los navegadores web utilizando el tag *video* de HTML5 y los formatos WebM (con HTTP) o H.264 (sobre RTMP).

La versión actual es un prototipo funcional que presenta las ideas principales del proyecto. El objetivo principal al que apunta el diseño es un bajo consumo de recursos. A pesar de encontrarse en una etapa tan temprana de su desarrollo, funciona de manera estable; y por esta razón se lo incluye en este análisis.

Incluye una interfaz web con un monitor de ancho de banda en tiempo real para detectar congestiones en la red. Al igual que FFserver soporta varios streams independientes y se configura con un archivo de propiedades que se debe especificar al momento de ejecutar el aplicativo. Desde allí se deberá indicar el puerto en el que el servidor esperará recibir los streams, un nombre y un password para cada uno de ellos.

Stream-m es una aplicación Java y como tal funciona en cualquier plataforma que disponga de una jvm[89] instalada. Bastará con ejecutar el servidor y alimentarlo con un stream generado con alguna de las herramientas ya mencionadas.

5.2.4.3 VLC Media Player como servidor

En el apartado de encoders ya se mencionó a VLC como una solución para realizar streaming. En cuanto a su funcionamiento como servidor permite a su vez tomar el rol de cliente de streaming, recibiendo flujos de otros encoders como FFmpeg, o bien, a través de la red desde proveedores remotos para luego volver a retransmitirlos. Como se ve VLC es una herramienta bastante potente.

[88] "FFmpeg" 2016. 2016. 30 Oct 2016 <http://ffmpeg.org/pipermail/ffmpeg-devel/2016-July/196500.html>
[89] "Java JVM" 2016. 30 Oct 2016 <https://docs.oracle.com/javase/specs/jvms/se7/html/jvms-1.html#jvms-1.2>

5.3 Pruebas y comparación de rendimiento

Tras investigar las distintas tecnologías disponibles para realizar streaming de video en el marco del proyecto DROPSY, se realizaron pruebas utilizando distintas herramientas y configuraciones con el propósito de encontrar la combinación que proporcione el mejor rendimiento posible.
En la mayoría de las pruebas la herramienta utilizada para realizar la captura y encoding de video fue la proporcionada por el paquete FFmpeg; con la excepción de las pruebas realizadas íntegramente con VLC como encoder y servidor de streaming.
El total de las pruebas locales realizadas se hicieron sobre una conexión de red 5 mbps de downstream y 1 mbps de upstream. El equipo sobre el que se ejecutaron las mismas cuenta con la siguientes características:

- CPU Intel(R) Core(TM) i5-4210M CPU @ 2.60GHz (4 cores)
- Memoria 12GB DDR3 1600Mhz

5.3.1 Combinar ffmpeg y stream-m

En este conjunto de pruebas se optó por utilizar la herramienta ffmpeg para realizar el encoding de video y stream-m como servidor para publicarlo.
La elección del aplicativo stream-m como candidato se basó en el reciente anuncio por parte del proyecto FFmpeg de retirar FFserver de su paquete de herramientas. Sin embargo, las pruebas realizadas pusieron de manifiesto algunas de las limitaciones de stream-m; que si bien muestra mucho potencial, aún se encuentra en etapas muy tempranas de su desarrollo.
De los protocolos ofrecidos por stream-m, sólo se consideró a WebM para las pruebas. Si bien el server soporta realizar streaming con H264, lo realiza sobre el protocolo RTMP, de escaso soporte en la plataforma Android.
Para preparar el entorno para las pruebas es necesario contar con una instalación del servidor stream-m y su correspondiente archivo de configuración. La Figura 16 muestra el archivo de configuración de referencia del servidor stream-m.

```
#
# Sample configuration file
# Empty lines and lines starting with # and ; are ignored.
# Format: <key> = <value>
# Syntactic elements (words) can be separated by linear whitepace.
#

# http.port
# listening port for the http server
http.port = 8080

# rtmp.port
# listening port for the rtmp server
rtmp.port = 8081

# stream.<streamname>
# if the value is "true" or "1" then the resource will be created
streams.first = true

# stream.<streamname>.password
# determines the password to accept the stream
streams.first.password = secret

# stream.<streamname>.limit
# maximum number of clients for this stream
streams.first.limit = 100
```

Figura 16 - Archivo de configuración de stream-m de referencia

En el archivo de configuración de stream-m es importante definir el puerto en el que stream-m escuchara pedidos (*http.port*), el password de acceso (*streams.<streamname>.password*) y el límite máximo de clientes (*streams.first.limit*). Una vez completados estos parámetros es posible iniciar una instancia del servidor ejecutando el comando mostrado en el Código 23.

```
~ java -jar stream-m.jar server.properties
```

Código 23 - Instanciación del servidor stream-m

Luego, es necesario iniciar el servicio de encoding como se muestra en el Código 24.

```
ffmpeg -f video4linux2 -i /dev/video0 -g 0 -vcodec libvpx -preset ultrafast -f webm
http://localhost:8080/publish/first\?password\=secret
```

Código 24 - Instanciación del servicio de encoding ffmpeg

Parámetros usados en el comando ffmpeg:

- **-f video4linux2**: driver de captura de video.
- **-i /dev/video0**: dispositivo de entrada, en esta caso la webcam.
- **-g 0**: gopsize, número de cuadros entre un frame clave.
- **-vcodec libvpx**: corresponde al codec de video empleado (VP8).
- **-preset ultrafast**: indica al codec que se optimice para brindar la menor latencia posible.
- **-f webm**: el formato de salida del video resultante.
- Por último, la url donde se envía el stream.

El rendimiento obtenido con estos parámetros fue de una imagen de calidad pobre y una latencia de entre 5 y 6 segundos en entornos locales. A través de Internet se lograron rendimientos similares a los locales: es decir que el impacto de la red fue casi despreciable. Obviamente estos valores dependen del estado de la red en el momento de la transmisión y de la conexión de la que disponga el cliente, entre otros factores. El mayor problema de ésta y otras configuraciones similares, más allá de la latencia inicial, fue su inestabilidad a los pocos minutos de iniciar el streaming, con tiempos de latencia que aumentaron hasta más de 10 segundos y en algunos casos pérdida completa de imagen.
La carga de CPU se mantuvo en niveles aceptables como se puede observar en la Figura 17, sobre todo teniendo en cuenta que se trata de un formato bastante intensivo en cuanto a procesamiento.

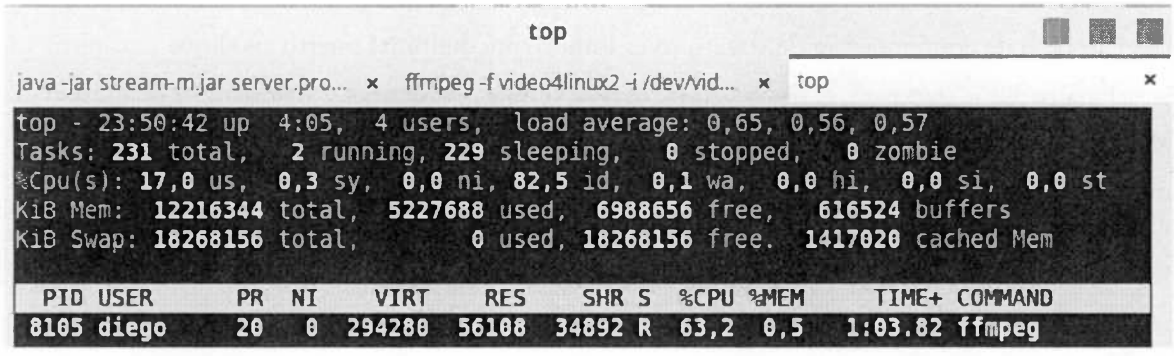

Figura 17 - Salida de comando top, ffmpeg funcionando con WebM

Perfil de rendimiento:

- Calidad de video: pobre, fluido.
- Latencia: 10 segundos en promedio.
- consumo de red: 500/600 kbps.
- consumo de cpu: 63% (de un core) 17% carga total.

Además de estos problemas de estabilidad y performance, se detectaron otros relacionados a la compatibilidad de WebM con ciertos dispositivos. En particular los teléfonos móviles de la marca LG con versiones 4.4 de Android y superiores se mostraron incapaces de reproducir correctamente un stream generado por ffmpeg. Esto se comprobó con nuestra configuración local y también con algunos de los streaming de prueba disponibles en distintos sitios de Internet. Luego de investigar en foros y grupos de ayuda, se descubrió que los móviles afectados poseían un reproductor multimedia propio del fabricante en lugar del reproductor por defecto de Android, siendo ésta presuntamente la causa del problema.

5.3.2 Combinar ffmpeg y ffserver

Esta combinación utiliza ffmpeg como encoder y ffserver como servidor de streaming. La configuración para utilizar ffmpeg junto con ffserver es similar a la de ffmpeg en conjunto con stream.m, sólo que a diferencia de la anterior, los parámetros enviados a ffmpeg se reducen al mínimo y es ffserver quien se encarga de definir las opciones que afecten al encoder y formatos de salida. De igual manera que con las pruebas de stream-m, el protocolo de video es utilizado fue WebM.

Más allá del reciente anuncio de ffmpeg sobre el cese de desarrollo sobre ffserver, este último sigue siendo un serio candidato como servidor de streaming; por su madurez, integración con ffmpeg y en general buen nivel de documentación (sobre todo comparado con la alternativa stream-m).

ffserver soporta numerosos protocolos y configuraciones. Como siempre nos limitaremos a aquellas con potencial para funcionar en Android.

En cuanto a los requisitos para levantar el servidor, solamente necesitamos contar con la instalación de ffmpeg y el archivo de configuración con los parámetros del servidor y del encoder utilizado. La Figura 18 muestra el archivo de configuración de referencia de ffserver.

```
HTTPPort 1234
RTSPPort 1935

<Feed feed1.ffm>
        File /tmp/feed1.ffm
        FileMaxSize 20M
        ACL allow 127.0.0.1
</Feed>

<Stream out.webm>
    Feed feed1.ffm
    Format webm
    Noaudio
    NoDefaults
     # Video settings
   VideoCodec libvpx
   VideoSize 320x240          # Video resolution
   VideoFrameRate 30        # Video FPS
   VideoGopSize 60
   PixelFormat yuv420p
   AVOptionVideo flags +global_header  # Parameters passed to encoder
                                       # (same as ffmpeg command-line parameters)
   AVOptionVideo cpu-used 4
   AVOptionVideo qmin 10
   AVOptionVideo qmax 30
   AVOptionVideo quality good
   AVOptionVideo threads 3
   AVOptionVideo maxrate 600k
   AVOptionVideo bufsize 1200k
   AVOptionAudio flags +global_header
   PreRoll 5
   StartSendOnKey
   VideoBitRate 600            # Video bitrate
</Stream>
```

Figura 18 - Configuración de ffserver de referencia

Los parámetros usados son:

- **VideoSize**: la resolución del video luego de realizar el encoding.
- **VideoFramerate**: cuadros por segundo luego del encoding. No necesariamente igual a los cuadros por segundo entregados por la cámara. Puede ser mayor o menor, en caso de ser mayor existirán cuadros duplicados, en caso de ser menor existirán cuadros descartados.
- **VideoGopSize**: indica cada cuántos cuadros se incluirá una imagen "completa", es decir que contiene toda la información necesaria para mostrarla, independientemente de las restantes. Los cuadros intermedios utilizan algún tipo de predicción o interpolado para mostrarse.
- **PixelFormat**: formato de video que utiliza el dispositivo de captura.
- **cpu-used**: cuanto más bajo sea este valor, más se intensifica la labor de la CPU en el encoding y se alcanzarán cotas mayores de calidad, a costa de una carga también mayor

de procesador. Con valores 4 y 5 se incluyen optimizaciones adicionales que reducen drásticamente el consumo de CPU y también la calidad de la imagen resultante..

- **qmin, qmax**: indican el rango de calidad en el que puede mantenerse el video. Cuanto más estático sea el video, mayor será la calidad en relación al *bit-rate*.
- **quality**: con esta propiedad en *good* se habilita la configuración **cpu-used**, que controla de manera más fina la relación entre CPU y calidad.
- **threads**: número de hilos empleados en la codificación. La configuración recomendadas es n-1 threads donde n es el número de núcleos del procesador.
- **maxrate, bufsize, VideoBitRate**: son parámetros relacionados al control de *bit-rate*; para intentar lograr una tasa más o menos constante de bits.
- **PreRoll**: indica a ffserver que envíe los primeros x segundos del streaming (prebuffer) lo más rápido que la red permita. Luego reduce la velocidad hasta lograr una transmisión en tiempo real.

Comenzamos iniciando ffserver como se muestra en el Código 25.

```
ffserver -f ffserver.conf
```

Código 25 - Instanciación del servidor ffserver

En nuestro caso, el archivo se encontraba en el mismo directorio donde ejecutamos las pruebas. De no ser el caso, podemos especificar la ruta completa. Luego iniciamos el servicio ffmpeg como se muestra en el Código 26, teniendo en cuenta que los parámetros sean consistentes con los definidos en el archivo de configuración.

```
ffmpeg -f video4linux2 -r 23.5 -s 320x240  -i /dev/video0 http://localhost:1234/feed1.ffm
```

Código 26 - Instanciación del servicio de encoding ffmpeg

Las opciones son similares a las utilizadas con stream-m, pero a diferencia de éste, los parámetros propios de encoder se configuran desde ffserver.

- **-r**: es la tasa de cuadros de entrada, es decir la que va a aceptar ffmpeg antes de realizar el encoding. Cada dispositivo de captura tiene un valor óptimo distinto por lo que es necesario ajustarlo según el hardware.
- **-s**: la resolución de entrada que recibe el proceso de encoding, no necesariamente la misma que la del video resultante.
- La url de salida dirige el flujo hacia el servicio ffserver para que éste lo retransmita.

Los resultados obtenidos con esta configuración fueron de una imagen de buena calidad, una latencia de alrededor de 1 segundo en entornos locales y un comportamiento estable.
Los parámetros del archivo de ejemplo fueron el resultado de varias pruebas con distintos valores hasta lograr un comportamiento satisfactorio. A medida que aumentaba el número de pruebas pudimos comprobar el impacto que tuvieron algunos de estos cambios.
En cuanto a la calidad del video, los parámetros más importantes resultaron ser, como era de esperar, los relacionados a la resolución de entrada y salida. Otro valor con gran incidencia fue cpu-used con valores mayores a 3 se aprecia un deterioro notorio en la calidad. En cuanto a la estabilidad y consumo de CPU (ambos bastante ligados en nuestras pruebas) también fue de importancia el valor de cpu-used, ya que con valores 0, 1, 2 el consumo de CPU fue bastante mayor, incluso con algunos problemas de estabilidad que provocaron interrupciones en el flujo de video a los pocos segundos de iniciado como puede observarse en las Figuras 19 y 20. La opción threads, que controla el nivel de paralelismo en el encoding tuvo cierto impacto aunque no muy significativo. Valores más allá del recomendado no presentaron ninguna mejora apreciable.

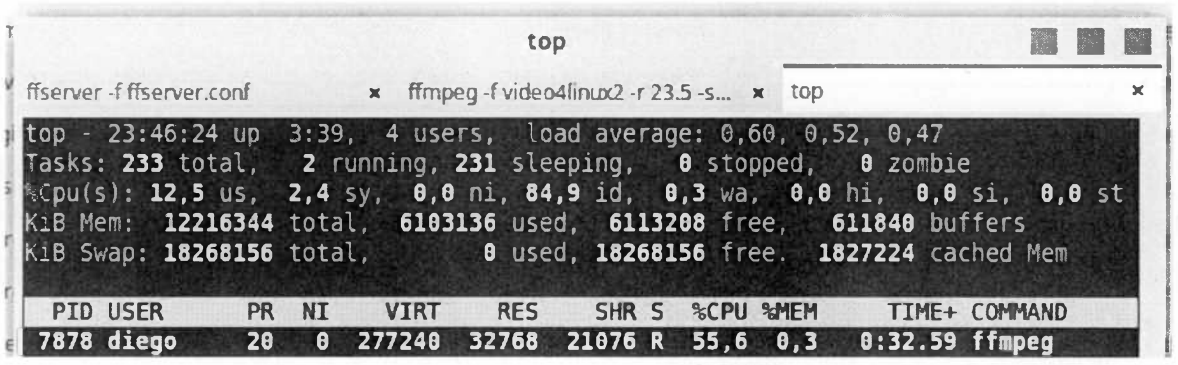

Figura 19 - Salida de comando top, parámetro **cpu-used** en 1

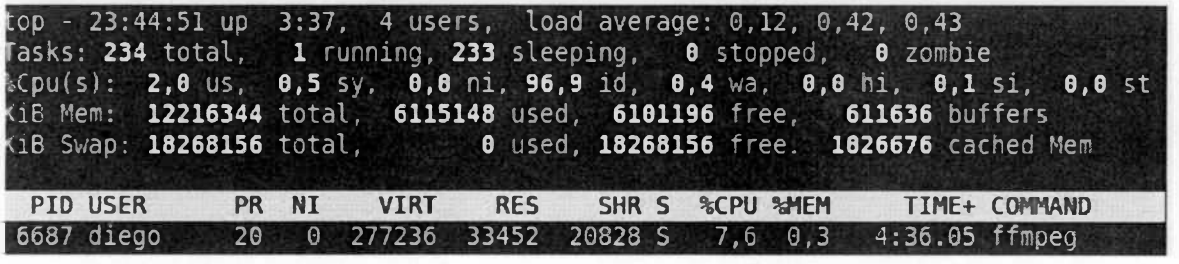

Figura 20 - Salida de comando top, parámetro **cpu-used** en 4

En cuanto a la latencia, fue mucho más difícil lograr un rendimiento predecible ya que son muchos los valores que la afectan. Uno de los más importantes fue la relación entre el frame-rate

de entrada y el de salida. Un valor ligeramente más alto en la salida mejoró notablemente el desempeño en este apartado. De manera similar valores de **cpu-used** más altos lograron cotas de latencia menores. Este último resultado era esperable ya que la CPU dedica menos tiempo a la codificación y compresión y como consecuencia entrega el flujo de video más rápido.
Un factor no mencionado hasta ahora y de gran impacto en el funcionamiento general del streaming fue el nivel de luz del ambiente. Parece algo obvio en principio, pero condiciones de luz óptimas (luz de día) siempre generaron el contexto ideal para las pruebas. La iluminación artificial, de buena intensidad, alcanza para lograr buenos resultados pero lejos de los obtenidos con iluminación natural.
Perfil de rendimiento:

- Calidad de video: muy buena, fluida.
- Latencia: 1 segundo (con iluminación óptima).
- consumo de red: 600/650 kbps.
- consumo de CPU: 17% (de un core) 2% carga total.

5.3.3 VLC en combinación con RTSP

Para este conjunto de pruebas se optó por utilizar VLC para codificar y publicar el streaming utilizando el protocolo RTSP. El proyecto FFmpeg en conjunto con ffserver soporta en teoría la combinación RTSP como protocolo y H264 como formato de empaquetado, sin embargo en nuestras pruebas fue imposible lograr una transmisión estable.
La utilización de VLC presenta el beneficio adicional de ser muy sencillo de instalar y configurar (sólo es necesario contar con los paquetes del reproductor vlc). Para instalar los paquetes necesarios ejecutamos los comandos mostrados en el Código 27.

```
sudo add-apt-repository ppa:mc3man/trusty-media
sudo apt-get update
sudo apt-get install vlc vlc-plugin-*
```

Código 27 - Instalación de VLC

Luego de ejecutar estos comandos en un entorno que posea el gestor de paquete APT, ya estamos en condiciones de comenzar con las pruebas.
Existen diversas maneras de configurar VLC para realizar streaming de video. La más sencilla es a través de su interfaz gráfica y siguiendo los pasos de la guía ofrecida para tal fin como muestra la Figura 21.

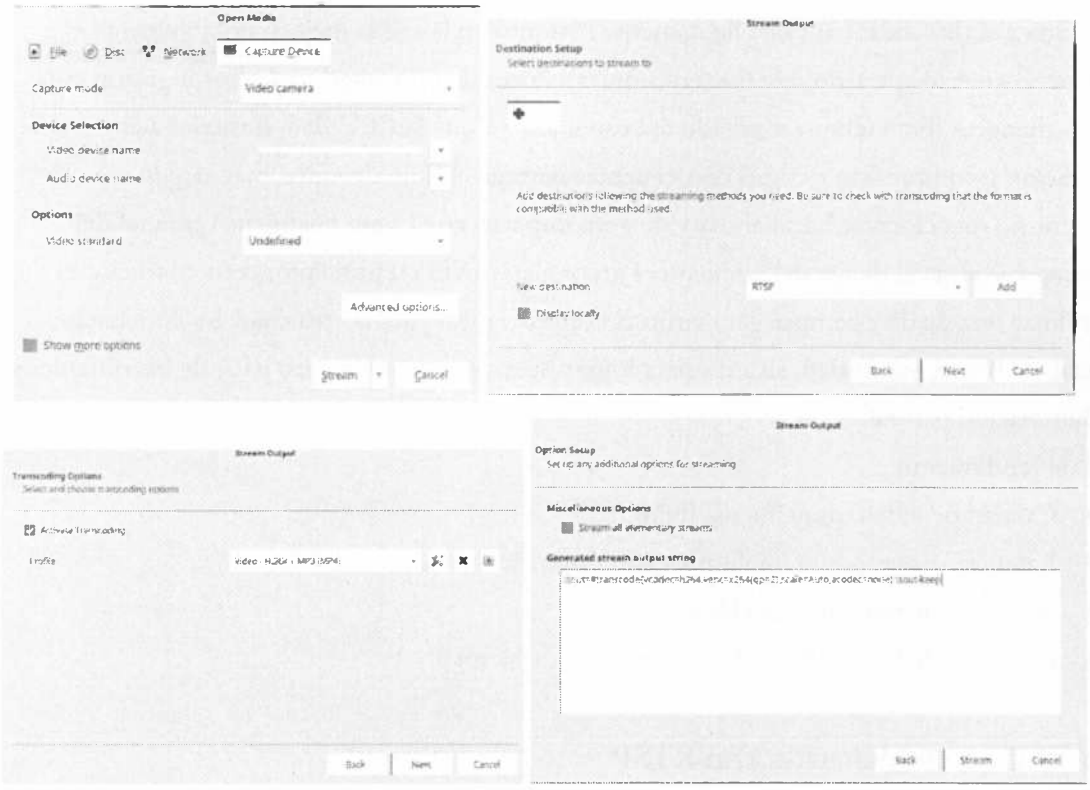

Figura 21 - Configuración del streaming, secciones 1,2,3 y 4

Al finalizar los pasos de la guía de configuración se obtiene un stream listo para publicar. Sin embargo para nuestras pruebas optamos por guardar la cadena de texto generada, para luego utilizarla al momento de iniciar el servicio vlc desde la línea de comandos y con el beneficio adicional de poder realizarle modificaciones posteriores para optimizar su funcionamiento.
Una vez obtenido el stream, lo iniciamos incluyendo algunos parámetros adicionales. A diferencia del paquete FFmpeg, con VLC sólo necesitamos un comando como el mostrado en el Código 28 para iniciar los servicios de encoding y retransmisión:

```
cvlc v4l2:///dev/video0 --sout
#transcode{vcodec=h264,vb=500,venc=x264{profile=baseline,preset=veryfast},scale=Auto,wi
dth=480,height=360,acodec=none}:rtp{sdp=rtsp://:8554/out.mp4}'
```

Código 28 - Instanciación del servicio de encoding y retransmisión de VLC

Los parámetros usados son:

- **v4l2**:///dev/video0: indica el dispositivo de video desde el que se realiza la captura.
- **--sout**: utilizando esta opción especificamos la cadena de entrada con las configuraciones específicas del streaming.

- **vcodec:** indica el códec de video.
- **vb:** la tasa de bits (bitrate)
- **venc:** el conjunto de parámetros específicos del encoder, profile y preset son configuraciones predeterminadas.
- **scale:** la resolución y escala de video.
- **acodec:** el códec de audio, en nuestro caso no se utiliza.
- **rtp:** en esta sección se configuran los parámetros del protocolo RTSP.
- **sdp:** el endpoint en el cual se publicará el streaming, incluyendo el puerto y el protocolo.

Esta configuración mostró muy buenos resultados en nuestras pruebas locales, con una imagen de muy buena calidad, latencia baja (alrededor de 1 segundo) y un funcionamiento sumamente estable. El consumo de recursos no fue un problema en ningún momento de las pruebas, con un consumo de CPU siempre en valores mínimos, como muestra la Figura 22. Esto se debe a que H264 es un codec menos complejo que VP8 (utilizado en las pruebas con WebM). Adicionalmente existe un parámetro de configuración de H264 (-profile:v baseline) que simplifica aún más este proceso para mejorar su rendimiento en dispositivos cliente con niveles más bajos de rendimiento de procesador, como teléfonos y tablets.

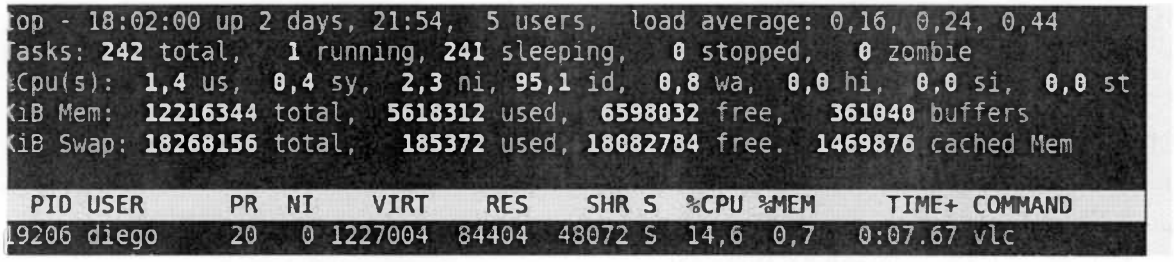

Figura 22 - Salida de comando top, consumo de recursos de VLC durante encoding

El buen rendimiento obtenido con VLC se logró casi sin realizar ajustes adicionales a las opciones del encoder, resolución o tasa de bits, a diferencia de las pruebas con ffmpeg, que requirieron de mucho esfuerzo en este apartado. Sin embargo, una vez que pasamos a las pruebas a través de Internet nos enfrentamos con un importante obstáculo: como mencionamos en la sección específica de RTSP, éste utiliza un puerto TCP para iniciar la negociación y una vez lograda, realiza las transmisiones de cada señal de video y audio sobre un puerto UDP dinámico. Este comportamiento representa un problema cuando el servidor se encuentra detrás de un firewall o de un router configurado con NAT[90] que dificultan y en muchos casos previenen la utilización de puertos no pactados previamente, en este caso los puertos UDP dinámicos de los que se vale RTSP para realizar la transmisión.

[90] "Network Address Translator" 1999. 31 Oct 2016 < https://tools.ietf.org/html/rfc2663>

Existe una alternativa que consiste en enviar todos los flujos RTSP a través de único puerto TCP de manera entrelazada, mediante una configuración específica del servidor de streaming. Sin embargo, esta funcionalidad no se encuentra soportada por Android en la actualidad.

Perfil de rendimiento:

- Calidad de video: muy buena, fluida.
- Latencia: 1 segundo (con iluminación óptima).
- consumo de red: variable entre 400/600 kbps.
- consumo de cpu: 14.6% (de un core) 1.4% carga total.

5.4 Solución elegida

Luego de evaluar las distintas alternativas que fueron puestas a prueba a lo largo de las sección anterior fue relativamente simple elegir una solución que se ajustara a las necesidades de DROPSY.
El conjunto de pruebas realizados utilizando ffmpeg y stream-m, no fue satisfactoria en casi ninguno de sus aspectos. Los problemas notados en estabilidad y calidad de video fueron demasiado importantes como para considerar esta combinación como una solución viable.
Las pruebas realizadas con VLC y el protocolo RTSP fueron satisfactorias en cuanto a métricas de rendimiento, estabilidad y calidad de la señal. Sin embargo y a pesar de estos resultados alentadores, decidimos no considerar su uso en el contexto de DROPSY, ya que su funcionamiento depende de condiciones de red que no siempre será posible garantizar. En particular, es necesario contar con configuraciones especiales en los firewall que permitan acceder a los puertos requeridos por el protocolo RTSP.
En el caso de las pruebas realizadas con ffmpeg en conjunto con ffserver sobre HTTP y utilizando el formato WebM, los resultados en cuanto a calidad de video, utilización de recursos y estabilidad fueron similares a los obtenidos con VLC y RTSP. Más allá del hecho de que alcanzar resultados satisfactorios fue bastante más laborioso que con el ya mencionado VLC y requirió de mucha experimentación con los parámetros de codificación y formatos de video; una vez logrado el objetivo quedó claro que ésta sería la configuración elegida para DROPSY. Esta decisión no se basa solamente en los parámetros cuantitativos mostrados a lo largo de las pruebas sino además en la independencia de la solución con respecto a las redes subyacentes, tanto en relación a su configuración como al ancho de banda del que dispongan.

Capítulo 6

Programando con DROPSY

Con el objetivo de facilitar la adaptación de los/as alumnos/as con el uso de la aplicación, como así también de brindar un recurso que, utilizado en conjunto con la aplicación desarrollada, permita a los/as alumnos/as comprender y experimentar los conceptos básicos de la programación, nos propusimos desarrollar una guía en línea de actividades denominada "Programando con DROPSY".
Si bien se prevé que esta guía se continuará ampliando en el marco del proyecto "Programando con Robots y Software Libre", el desarrollo de una primera versión de la misma nos permitió realizar pruebas que sirvan para dar una primera idea de la eficiencia de la utilización de la misma en conjunto con la aplicación DROPSY.

6.1 Formato elegido

Una de las cuestiones más importantes que tuvimos que determinar fue el formato en el que iba a ser presentada la guía de actividades a ser utilizada por los/as alumnos/as.
Luego de considerar distintos formatos como PDF, presentación de diapositivas y serie de videos, decidimos optar por la elaboración de una página web específica para la guía de actividades ya que consideramos que ésta era la manera más simple y atractiva de presentar el contenido, brindándonos además la posibilidad de poder editar el contenido en cualquier momento y de utilizar un repositorio público de Git para controlar los cambios y versiones. En la Figura 23 se muestra el estilo usado en la página web diseñada.

Figura 23 - Inicio de Programando con DROPSY

6.2 Hosting elegido

A la hora de elegir donde alojar la página web desarrollada, nos encontramos con la opción de hacerlo a través de Github Pages[91], un servicio de GitHub [92]que brinda a los desarrolladores la posibilidad de alojar páginas web en un dominio de GitHub de manera gratuita, con la particular ventaja de que los cambios en las páginas se ven reflejados automáticamente al introducir cambios al repositorio, por lo que basta con realizar un commit y realizar un push al repositorio para observar los nuevos cambios en la página web.
La guía está disponible en: https://dropsy-unlp.github.io/programando-con-dropsy/

6.3 Guía de uso de DROPSY destinada a niños/as y adolescentes

La guía desarrollada cuenta con una breve introducción al proyecto DROPSY, destacando la motivación que llevó a la realización del mismo, como así también una introducción a Programando con DROPSY. El contenido principal de la guía son las 5 unidades que posee, cada una de las cuales cuenta con explicaciones gráficas acerca del uso de los distintos bloques y del propósito de los mismos, actividades en las cuales se utilizan los nuevos bloques y soluciones a las actividades de manera que se pueda verificar la correcta resolución de los mismos.

[91] "Github Pages" 2016. 31 Oct 2016 <https://pages.github.com/>
[92] "Github" 2016. 31 Oct 2016 <https://github.com/>

Las actividades prácticas tienen como objetivo consolidar los conceptos introducidos en cada una de las unidades. En las unidades 2, 3, 4 y 5 se presentan entre 2 y 3 actividades en las cuales el/la alumno/a debe intentar reproducir una figura presentada moviendo el robot. Para cada una de las actividades se define cuales son los bloques permitidos y se les otorga una dificultad.

6.3.1 Primeros pasos con DROPSY

La primera unidad es de carácter introductorio y en la misma se introduce al/la alumno/a en el uso de la aplicación. La misma posee orientaciones acerca de cómo:

- Crear un proyecto
- Guardar un proyecto
- Cargar un proyecto
- Agregado de bloques a un proyecto
- Ejecutar un programa

Consideramos que esta unidad es importante ya que a una persona no familiarizada con las herramientas de programación basadas en bloques puede no resultarle intuitivo el uso de los mismos.

6.3.2 Bloques de movimiento

La segunda unidad de la guía de actividades se centra en los bloques de movimiento del robot. En esta unidad se le da una breve introducción al/la alumno/a acerca de la categoría de bloques de movimiento y se intenta reforzar algunos de los conceptos introducidos en la unidad 1 como son el agregado de bloques a de un proyecto y la ejecución del mismo.
Las actividades propuestas en esta unidad son de dificultad principiante y para resolverlas están disponibles solamente los bloques de movimiento. En la figura 24 se muestra la actividad 1, en la cual el alumno debe reproducir la figura de un cuadrado utilizando el robot.

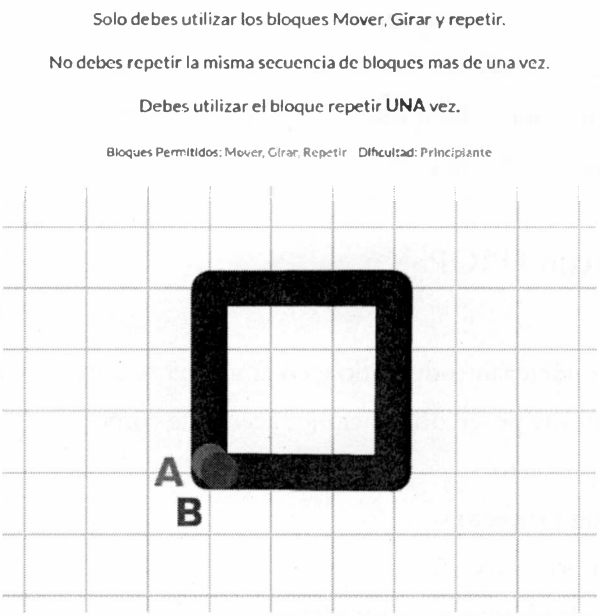

Figura 24 - Ejercicio en Programando con DROPSY

6.3.3 Bloques de iteración

En esta unidad se le da al/la alumno/a una breve introducción al concepto de iteración, permitiéndole entender la utilidad que representa el bloque de repetición en programación. Además se muestra cómo utilizarlo dentro de los proyectos.
La unidad consta además de 3 actividades las cuales los/as alumnos/as deben resolver utilizando el bloque de repetición junto con los bloques de movimiento, sumando de esta manera un nuevo bloque a los 2 ya utilizados en la unidad anterior. En esta unidad, 2 de las actividades son de dificultad principiante y una es de dificultad intermedia ya que consideramos que su resolución requiere un grado más alto de lo comprensión de los conceptos presentados en las primeras 3 unidades. En la Figura 25 se muestra la solución a la actividad 1 de la Unidad 3, en la cual el alumno debe reproducir una figura escalonada utilizando el bloque repetir y los bloques de movimiento.

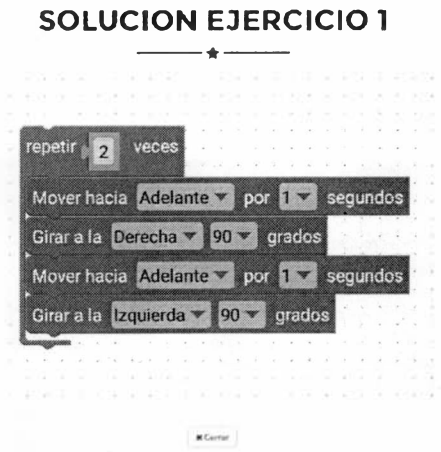

Figura 25 - Solución en Programando con DROPSY

6.3.4 Bloques para definir y usar variables

En esta unidad se introduce al/la alumno/a en el uso de variables, justificando la importancia de las mismas como así también la ~~gran~~ utilidad que representan dentro de los programas. La unidad se separa en 4 secciones las cuales muestran al/la alumno/a como:

- Crear variables
- Renombrar variables
- Modificar variables
- Leer variables

Esta unidad además, finaliza la explicación con un ejemplo completo en el cual se demuestra cómo decrementar el valor de una variable, algo que resulta muy útil durante las actividades, este ejemplo se muestra en la Figura 26. Consideramos necesario dicho ejemplo ya que este concepto es de vital importancia en la resolución de las actividades de la unidad.
La unidad consta de 2 actividades, ambas de dificultad intermedia, en las cuales pueden utilizarse los bloques de movimiento, de repetición y de manejo de variables.

Figura 26 - Ejemplo en Programando con DROPSY

6.3.5 Bloques de selección

En la última unidad se presenta al/la alumno/a el concepto de estructura condicional y se le explica cómo ayudar al robot a tomar decisiones durante la ejecución del programa. Para esto aplicamos conceptos de unidades anteriores, especialmente variables; y cómo combinar ambos para completar las actividades.

Más adelante se presenta el bloque condicional y se le muestra al/la alumno/a cómo utilizarlo en un proyecto en conjunto con los otros bloques.

Luego se incluyen actividades para aplicar los conceptos aprendidos en esta unidad y además para integrar lo visto en las unidades anteriores. La dificultad varía de intermedia a avanzada ya que consideramos que en este punto el/la alumno/a está en condiciones de trabajar con todos lo bloques de DROPSY pudiendo resolver actividades de mayor complejidad.

Capítulo 7

Pruebas de DROPSY con niños/as, adolescentes y adultos

En este capítulo se describirán las pruebas realizadas con DROPSY que nos permitieron hacer una primera evaluación de la herramienta y de la guía de actividades propuesta.

7.1 Metodología de evaluación de DROPSY

La guía "Programando con DROPSY" propone ser la herramienta que acompañe al/la alumno/a durante el proceso de aprendizaje de programación con DROPSY. La guia esta en su primera versión, por lo que puede contener errores, omisiones y supuestos que se irán corrigiendo en futuras versiones. De la misma manera es importante exponer la herramienta a un uso más general para pulir ciertos detalles y mejorar la experiencia de usuario; para acercarla aún más a los/as niños/as y adolescentes que son el foco principal de este trabajo. Es por esto que incluimos una breve guía con los criterios de evaluación que serán aplicados en el transcurso de las pruebas a los/as alumnos/as que participen de ellas.

La metodología de evaluación utilizó como instrumento un formulario, el cual fue completado para cada uno de los participantes. El formulario utilizado sigue los siguientes criterios:

- **Nivel de comprensión de los conceptos introducidos en cada unidad**.
  Preferentemente ofrecer una breve reseña de lo observado en cada una de las unidades y criterios; utilizando como guía las siguientes preguntas:
  - ¿Entendió para qué sirve un bloque determinado?
  - ¿Logró encontrar la funcionalidad explicada en la interfaz de la aplicación?
  - ¿Requirió de explicaciones adicionales por parte del tutor?
  - ¿Tuvo que releer varias veces la unidad para comprender los conceptos?
- **Completitud y corrección de las actividades planteadas.**
  Este aspecto evalúa el desempeño del alumno a la hora de resolver las actividades planteadas.
  - ¿Pudo completar todos las actividades?

- ¿Aplicó los conceptos explicados en la unidad?
- ¿Tuvo dificultades para comprender el enunciado?
- ¿Requirió de la ayuda del tutor para resolver algúna actividad?
- ¿Las soluciones propuestas por el alumno coinciden con las respuestas incluidas en la guía?
- Tiempo dedicado a la resolución de las actividades de cada unidad.

- **Nivel de entusiasmo del participante.**
  Este es un criterio muy subjetivo y como tal puede ser interpretado de muchas maneras. Sin embargo fue incluido a modo de retroalimentación para mejoras de la herramienta y además para obtener información en cuanto al interés que pueda despertar la temática en los alumnos.
  - ¿Se lo notó entusiasmado en el transcurso de la prueba?
  - ¿Manifestó algún descontento respecto de la herramienta o de la guía de actividades?
  - ¿Hizo alguna sugerencia de mejora o mencionó alguna funcionalidad que le gustaría haber visto en la aplicación?
  - ¿Completó la totalidad de la guía?

Algunos datos adicionales para registrar durante cada prueba:

- Tiempo total dedicado a la prueba.
- Tiempo dedicado a la sección teórica de cada unidad.
- Tiempo total dedicado a las actividades de cada unidad.
- Edad del participante.
- ¿Poseía algún tipo de conocimiento sobre programación previo a la prueba?

La identidad de cada participante y otros datos personales quedarán excluidos de los resultados de las pruebas y de cualquier otra parte del trabajo. De todas maneras no está entre los objetivos de estas pruebas el de evaluar a los participantes en sí, sino a la herramienta y la guía de actividades; con el fin de recabar información que nos ayude a mejorarla de cara a un uso educativo de la misma.

7.2 Pruebas de uso, retroalimentación y cambios

Una vez definida la metodología de evaluación y establecidas las metas, comenzamos con las pruebas de utilización de DROPSY, centrándonos en conseguir la mayor cantidad de información posible para mejorar la herramienta y guías de uso. Si bien no esperamos que los/as alumnos/as aprendan los conceptos necesarios para programar en una sola sesión de uso de la aplicación; sí creemos que es posible despertar en ellos/as un gran interés por la programación, apoyándonos en la novedad que representa controlar un robot real desde un dispositivo cotidiano como una tablet o un teléfono celular inteligente, en lugar del acercamiento a través de un lenguaje de programación textual, como los usados en la industria de desarrollo de software.

Para la realización de las pruebas optamos por separar a los participantes en 3 grupos bien definidos que abarquen personas de todas las edades: niños entre 8 y 13 años, adolescentes entre 14 y 18 años y adultos mayores de 19 años. Esta división no es arbitraria, sino que refleja nuestra intención de que la herramienta pueda ser utilizada por personas de todas las edades.
Durante las pruebas en cada uno de los rangos de edades surgen objetivos un poco distintos:

- Niños (8 a 12): Principalmente se busca observar el interés de los niños por la programación con bloques y por el control del robot. Además se busca observar si la guía es fácil de entender y si pueden hacer un correcto uso de la aplicación para manejar los bloques.
- Adolescentes (13 a 18): se buscó evaluar tanto el avance en la guía de actividades como la calidad de las actividades, observando si las mismas son capaces de introducir los conceptos presentados en las unidades y si se presentan con una dificultad incremental.
- Mayores (19 en adelante): los/as alumnos/as pertenecientes a este rango son capaces de explicar las dificultades que encontraron a lo largo de la guía, proponiendo con esto mejoras a la misma. Además es posible evaluar la calidad de las actividades propuestas.

Las pruebas se realizaron basándonos en los lineamientos definidos en los criterios de evaluación (comunes para los tres rangos de edad) y los resultados fueron volcados en los formularios elaboradas para tal fin. Para las mismas se montó un servidor corriendo una versión adaptada de XRemoteBot, el cual se encargó de comunicarse con un robot Multiplo N6 que se encontraba a pocos metros tanto del servidor como de los/as alumnos/as. En esta ocasión, los autores de esta tesina actuaron como tutores, encargados de contestar dudas e inquietudes de los alumnos.
Las pruebas fueron divididas en etapas luego de cada una de las cuales se redactó un breve informe con las conclusiones y mejoras puntuales que surgieron luego de analizar la información recolectada en cada una y se realizaron mejoras para intentar solucionar los inconvenientes encontrados.

7.3 Resultados de las pruebas

Se realizaron pruebas en alumnos de entre 12 y 26 años de edad de forma individual o en pareja. Cada una de las pruebas tuvo una duración de entre 1 y 2 horas, en las cuales cada uno los/as alumnos/as tuvieron distintos desempeños. Desafortunadamente en el tiempo que se poseía no se pudo concretar una segunda sesión con ninguno de los/as alumnos/as, por lo que el avance de cada uno en la guia se ve limitado por una única sesión.
Durante las pruebas se registraron los resultados de las mismas utilizando los criterios de evaluación planteados en la sección 7.1 y luego, teniendo en cuenta los resultados obtenidos, se evaluaron mejoras tanto en la aplicación como en la guia "Programando con DROPSY".
Los resultados de las pruebas están divididos en distintas etapas que describen brevemente los perfiles de los/as alumnos/as evaluados como así también los cambios realizados al finalizar las mismas.

7.3.1 Etapa 1: primeras pruebas

En esta etapa se realizaron las pruebas a los dos primeros alumnos, luego de las cuales se realizaron cambios teniendo en cuenta la retroalimentación obtenida.

Primer alumno: Agustín
Nuestro primer alumno, mostrado en la Figura 27 y 28, es un adolescente de 14 años de edad llamado Agustín que, si bien no poseía conocimiento previos en programación, si posee experiencia utilizando dispositivos electrónicos y en particular el sistema operativo Android. En aproximadamente una hora el alumno logró completar correctamente las unidades 1, 2 y 3.

- Nivel de comprensión de los conceptos introducidos en cada unidad:
    - Unidad 1: esta unidad es de carácter introductorio y el alumno no tuvo inconvenientes.
    - Unidad 2: el alumno comprendió los bloques de movimiento y los aplicó correctamente.
    - Unidad 3: en esta unidad el alumno necesitó releer las explicaciones a la hora de resolver las actividades y con esto logró entender la utilidad de los bloques de repetición. Se notó un poco de confusión al haber bloques en la aplicación que luego no eran explicados ni necesitados, se decidió eliminar de la aplicación todos los bloques que no eran necesarios.
    - Unidad 4: nuevamente el alumno leyó varias veces las explicaciones para terminar de comprender los conceptos introducidos. Esta vez se necesitó además una breve explicación del ejemplo presentado al final de la unidad para un entendimiento pleno del mismo.
    - Unidad 5: no se evaluó por falta de tiempo.
- Completitud y corrección de las actividades planteadas:
    - Unidad 1: esta unidad no presenta actividades.
    - Unidad 2: el alumno pudo resolver las 3 actividades planteados en menos de 10 minutos, necesitando un par de intentos tanto para la actividad 1 como la actividad 3, probando en el robot los intentos fallidos y pudiendo a partir del movimiento de este reconocer los errores cometidos y solucionándolos en el código. Las soluciones obtenidas fueron las esperados. Dada la facilidad para resolver las actividades se decidió cambiar algunos para que tengan una dificultad un poco mayor y evitar que sean repetitivos.
    - Unidad 3: una vez más el alumno necesito reiterados intentos fallidos para dar con la solución a las actividades, probando cada intento en el robot. El alumno pudo aplicar el concepto de repetición para evitar el reuso de bloques en los programas. Las soluciones obtenidas fueron las esperadas. Se notó que la tercer actividad era muy difícil y repetitiva por lo que se decidió cambiarla.

- Unidad 4: el alumno no pudo resolver ninguna de las actividades presentadas ya que estaba cansado. Intentó resolver la primer actividad pero no lo logró. Se consideró que esta actividad poseía una dificultad demasiado alta y se decidió cambiarla.
- Unidad 5: no se evaluó por falta de tiempo.

- Nivel de entusiasmo del participante:
  - Unidad 1: el alumno sólo necesitó alrededor de 5 minutos para leer y comprender los conceptos introducidos en esta unidad.
  - Unidad 2: en los 10 minutos que necesitó para completar la unidad se lo notó entretenido, el hecho de tener el robot cerca y poder probar cada intento en el mismo es algo que sin dudas entretuvo y entusiasmo al alumno.
  - Unidad 3: el alumno necesitó 30 minutos para completar la unidad, aunque al finalizar la misma ya estaba visiblemente agotado.
  - Unidad 4: el alumno, ya notablemente cansado, decidió parar.
  - Unidad 5: no se evaluó.

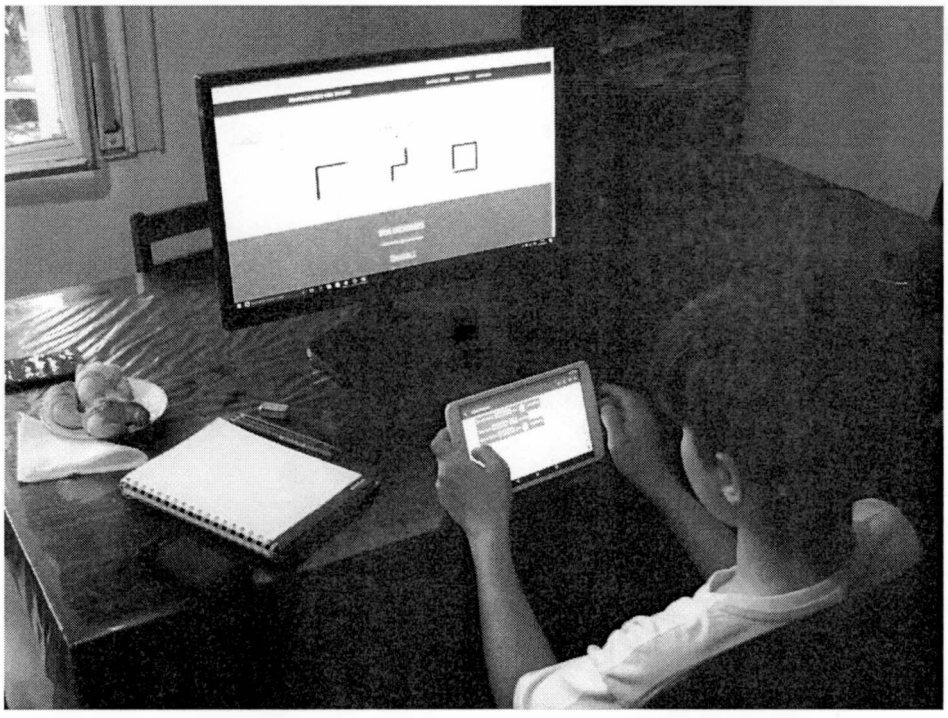

Figura 27 - Primer alumno: Agustín

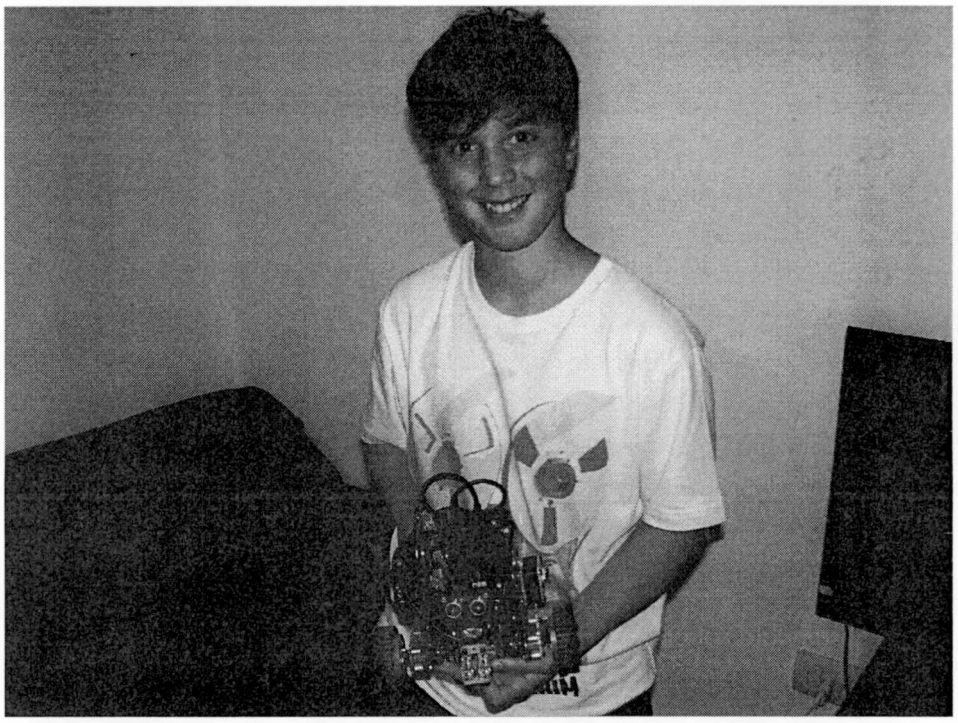

Figura 28 - Primer alumno: Agustín

Segundo alumno: Mauro

Nuestro segundo alumno, mostrado en la Figura 29, es un hombre de 26 años de edad llamado Mauro que, al igual que el alumno anterior, no poseía conocimiento previos en programación pero si posee experiencia utilizando dispositivos electrónicos y en particular con el sistema operativo Android. En aproximadamente dos horas el alumno logró completar correctamente las unidades 1, 2, 3 y 4.

- Nivel de comprensión de los conceptos introducidos en cada unidad:
    - Unidad 1: esta unidad es de carácter introductorio y el alumno no tuvo inconvenientes.
    - Unidad 2: el alumno comprendió los bloques de movimiento y los aplicó correctamente.
    - Unidad 3: en esta unidad el alumno necesito releer las explicaciones al momento de resolver las actividades y con esto logró entender la utilidad de los bloques de repetición.
    - Unidad 4: nuevamente el alumno necesitó releer en reiteradas ocasiones las explicaciones para entender completamente los conceptos introducidos. Además se le explicó brevemente el ejemplo mostrado en la guia. El alumno presentó dudas en cuanto a los nombres elegidos para las variables, creyendo en un principio que era necesario usar un nombre determinado para las mismas, se

decidió corregir la guia para usar nombres más genéricos que no generen esta confusión.

- Unidad 5: el alumno logró comprender satisfactoriamente la utilidad del bloque Si-Entonces luego de releer la unidad retiradas ocasiones. El alumno notó que no había explicacion de los comparadores lógicos "And-Or", se decidió incluir una breve explicación a la guia.

- Completitud y corrección de las actividades planteadas:
  - Unidad 1: esta unidad no presenta actividades.
  - Unidad 2: el alumno sólo necesitó 5 minutos para completar correctamente todas las actividades.
  - Unidad 3: el alumno completó correctamente todas las actividades, sin embargo se presentaron dudas en cuanto a los enunciados y a las condiciones para resolver los mismos las cuales el tutor contestó. Se decidió cambiar las actividades 1 y 3 por otras en los cuales la consigna sea más clara.
  - Unidad 4: el alumno completó correctamente las actividades 2 y 3. Se decidió que no haga la actividad 1 ya que había notado mucha confusión para resolverla por parte del alumno anterior.
  - Unidad 5: el alumno pudo resolver correctamente la actividad 1 pero se notó una dificultad muy alta para el mismo. Se decidió cambiar tanto la actividad 1 como la 2 por otras actividades más accesibles y que fuercen la utilización de los conceptos introducidos en la unidad de una manera más clara.
- Nivel de entusiasmo del participante:
  - Unidad 1: el alumno se interesó por probar la aplicación cuando vio de qué se trataba mientras se realizaban las pruebas a Agustín. Esto nos da una idea de los atractivo que resulta la experiencia para aquellos alumnos que no poseen experiencia previa en la programación.
  - Unidad 2: el alumno se mostró muy incentivado y entusiasmado con poder probar el código desarrollado utilizando el robot.
  - Unidad 3: la dificultad incremental de las actividades y la complejización de las mismas en este caso fue algo que atrajo aún más al alumno.
  - Unidad 4: una vez más se lo notó interesado en comprender los conceptos introducidos y en poder afrontar la dificultad de las actividades
  - Unidad 5: el alumno seguía demostrando interés por aprender los conceptos introducidos incluso rondando las 2 horas de prueba.

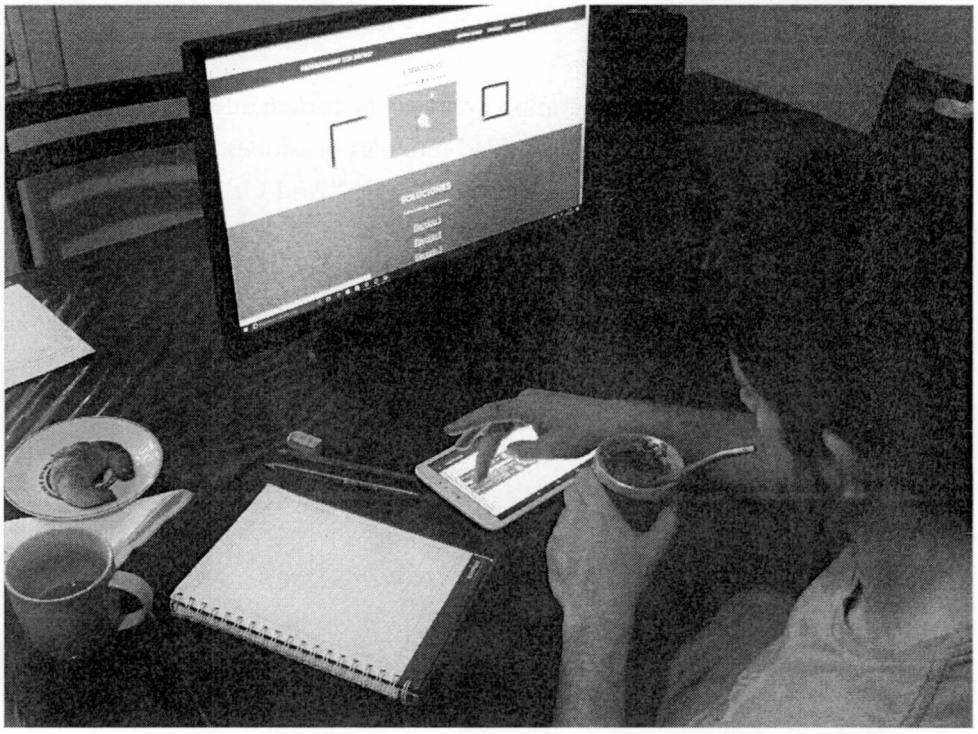

Figura 29 - Segundo alumno: Mauro

Cambios realizados luego de la etapa 1:
Se decidió realizar los siguientes cambios en la aplicación y en la guía al finalizar la primer etapa de pruebas:

1. Eliminación en la aplicación DROPSY de todos aquellos bloques que no se explican en la guia o se utilizan en las actividades.
2. Mejorar las actividades de la unidad 2 para incrementar la dificultad de las mismas ya que las existentes eran notablemente repetitivas y muy fáciles de resolver.
3. Cambiar la actividad 1 y 3 de la unidad 3 por otras con dificultad más baja y que fuercen la correcta utilización del bloque repetir.
4. Cambiar la actividad 1 de la unidad 4 por otra con dificultad más baja y que fuerce la correcta utilización de variables para resolverse.
5. Mejorar la explicación de variables en la guia de tal manera que se utilicen nombres genéricos para las mismas con el fin de no confundir al alumno, haciéndole creer que los nombres son determinantes para el funcionamiento del programa.
6. Agregar explicación de comparadores lógicos en la unidad 5 de la guia.
7. Cambiar las actividades 1 y 2 de la unidad 5 por otras mejor planteadas y que fuercen la consolidación de los conceptos introducidos.

7.3.2 Etapa 2: probando las mejoras

Luego de introducir los cambios planteados en la etapa 1 se continuó con las pruebas, teniendo como alumnos/as en esta nueva etapa a una adolescente de 16 años y un adulto de 26.

Tercer alumno: Abril
Nuestra tercera alumna, mostrada en la Figura 30 y 31, es una adolescente de 16 años de edad llamada Abril que, al igual que el resto de los participantes anteriores no poseía conocimientos previos en programación. En aproximadamente una hora Abril logró completar correctamente las unidades 1, 2 y 3.

- Nivel de comprensión de los conceptos introducidos en cada unidad:
    - Unidad 1: esta unidad es de carácter introductorio y la alumna no tuvo inconvenientes.
    - Unidad 2: la alumna comprendió los bloques de movimiento y los aplicó correctamente.
    - Unidad 3: en esta unidad la alumna requirió una breve explicación por parte del tutor para terminar de entender la utilidad del bloque de repetición que se explicó en la unidad.
    - Unidad 4: en este caso la alumna leyó más de una vez la explicación para terminar de entender los conceptos, además el tutor brindó una breve explicación utilizando como ejemplo el presentado al final de la unidad.
    - Unidad 5: no se evaluó por falta de tiempo.
- Completitud y corrección de las actividades planteadas:
    - Unidad 1: esta unidad no presenta actividades.
    - Unidad 2: la alumna pudo resolver las 3 actividades planteados en menos de 5 minutos, necesitando un único intento para resolver tanto la actividad 1 como la actividad 2 y sólo 2 intentos para la actividad 3. Los cambios introducidos en estas actividades demostraron ser buenos, incrementando la dificultad con respecto a las que había anteriormente y quitando repetitividad a los mismos. Al intentar repetidamente ejecutar programas nos encontramos con la presencia de un bug en la aplicación Android que no permitía reservar Robots.
    - Unidad 3: la alumna necesitó algunos intentos para resolver correctamente la actividad 1, resolvió la actividad 2 fácilmente pero no pudo resolver correctamente la actividad 3 y decidió continuar con la unidad 4. Se consideró que lo mejor era evaluar el desempeño de los demás alumnos con la actividad 3 para comprobar si era necesario o no cambiarla.
    - Unidad 4: la alumna logró resolver correctamente la actividad 1.
    - Unidad 5: no se evaluó por falta de tiempo.
- Nivel de entusiasmo del participante:

- Unidad 1: la alumna sólo necesitó alrededor de 5 minutos para leer y comprender los conceptos introducidos en esta unidad.
- Unidad 2: la alumna sólo demoró 5 minutos en completar la unidad y rápidamente se familiarizó con el uso de la aplicación, el hecho de poder probar los programas utilizando el robot una vez más fue un factor para motivar a la alumna.
- Unidad 3: la alumna estuvo alrededor de 20 minutos trabajando en las actividades de esta unidad, al llegar a la última actividad estaba notablemente agotada y ésta probablemente fue la razón que la llevó a no poder completarla.
- Unidad 4: la alumna ya se mostraba cansada y decidió parar luego de resolver la actividad 1.
- Unidad 5: no se evaluó.

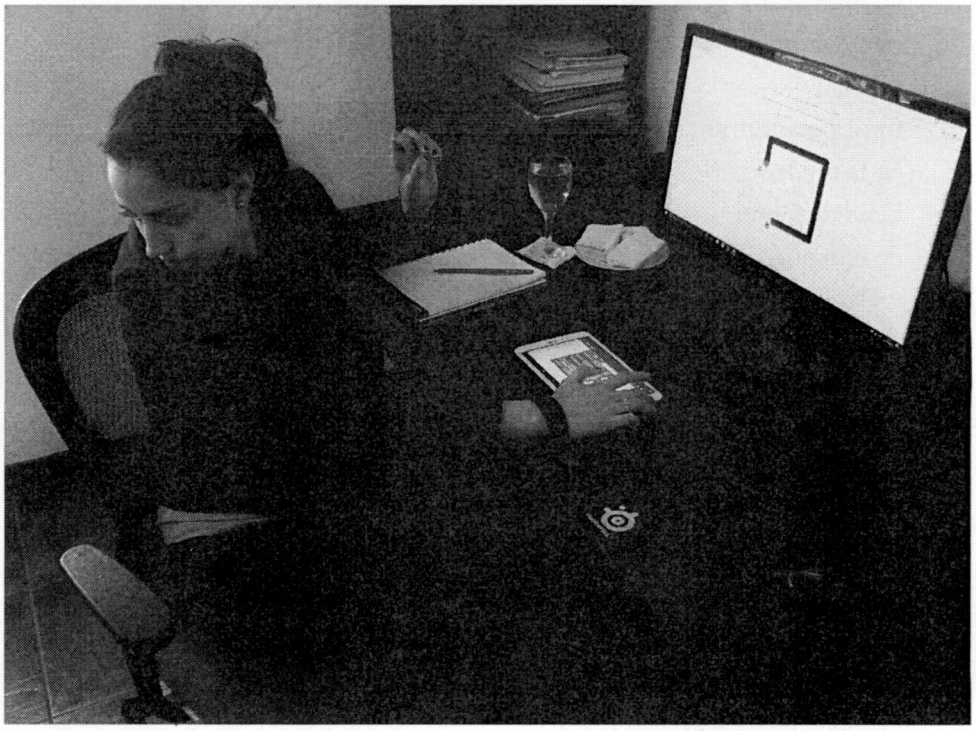

Figura 30 - Tercer alumno: Abril

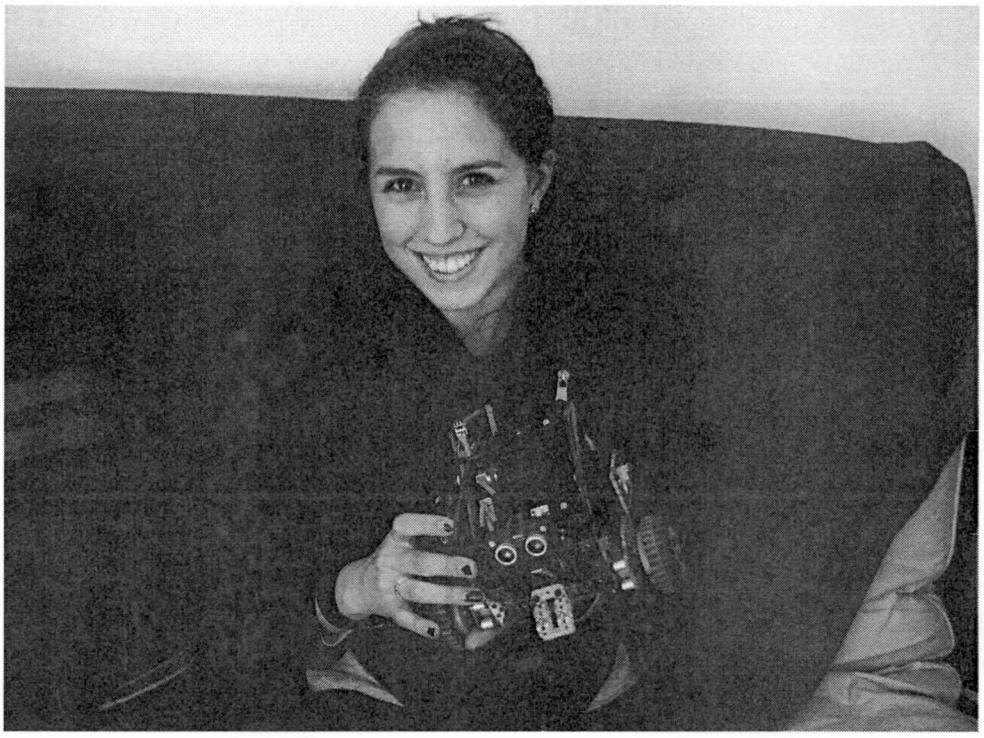

Figura 31 - Tercer alumno: Abril

Cuarto alumno: Germán

Nuestro cuarto alumno, mostrado en la Figura 32, es un hombre de 26 años de edad llamado Germán que, al igual que el resto de los participantes anteriores no poseía conocimientos previos en programación. En aproximadamente dos horas Germán logró completar correctamente las unidades 1, 2, 3 y 4.

- Nivel de comprensión de los conceptos introducidos en cada unidad:
  - Unidad 1: esta unidad es de carácter introductorio y el alumno no tuvo inconvenientes. Destacó la falta de tildes en algunas partes de la guia.
  - Unidad 2: el alumno comprendió los bloques de movimiento rápidamente y los aplicó correctamente.
  - Unidad 3: una vez más el alumno no tuvo mayores inconvenientes y pudo aplicar los conceptos explicados correctamente.
  - Unidad 4: en este caso el alumno leyó más de una vez la explicación para terminar de entender los conceptos, además el tutor brindó una breve explicación utilizando como ejemplo el presentado al final de la unidad.
  - Unidad 5: nuevamente el alumno necesitó leer varias veces la unidad y se le debió hacer una explicación para ayudar el entendimiento de los conceptos introducidos.
- Completitud y corrección de las actividades planteadas:
  - Unidad 1: esta unidad no presenta actividades.

- Unidad 2: el alumno pudo resolver las 3 actividades planteadas en menos de 10 minutos, necesitando un único intento para resolver cada actividad.
- Unidad 3: nuevamente el alumno sólo necesitó 10 minutos para resolver todas las actividades, pudiendo resolver cada actividad en el primer intento. Al no tener mayores inconvenientes con la actividad 3, se decidió dejarla en la guia.
- Unidad 4: el alumno necesitó alrededor de 1 hora para resolver todas las actividades, siendo la primer actividad la de mayor dificultad pero una vez resuelta pudo resolver los otros 2 sin mayores inconvenientes.
- Unidad 5: el alumno intentó resolver la primer actividad pero no pudo continuar la prueba por otros compromisos.

- Nivel de entusiasmo del participante:
  - Unidad 1: el alumno comenzó las pruebas con mucha predisposición y conociendo el avance que habían logrado el resto de los alumnos lo que lo incentivó aún más a, minimamente, llegar hasta el mismo punto que los demás.
  - Unidad 2: el alumno completó la unidad fácilmente lo que lo incentivó a continuar para descubrir qué nuevos desafíos habría en la guia.
  - Unidad 3: una vez más el alumno completó la unidad fácilmente, esto lo motivó a continuar ya que claramente un progreso tan rápido demostraba que poseía cierta facilidad para incorporar los conceptos introducidos.
  - Unidad 4: en esta unidad el alumno necesito un esfuerzo mucho mayor a las unidades anteriores pero nuevamente mostró mucha predisposición ya que le parecían interesantes los conceptos que se introducían.
  - Unidad 5: el alumno intento lo más que pudo pero debió abandonar por tener otros compromisos, sin embargo nunca dejó de buscar soluciones y de intentar entender los conceptos introducidos en la unidad.

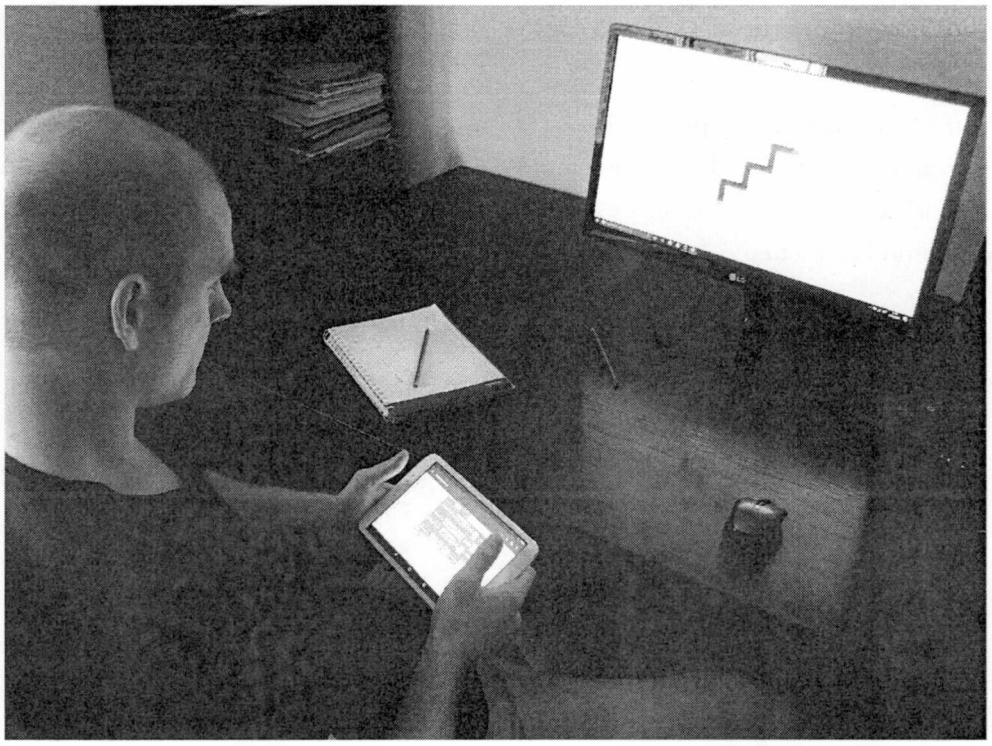

Figura 32 - Cuarto alumno: German

Cambios realizados luego de la etapa 2:
Se decidió realizar los siguientes cambios en la aplicación y en la guía al finalizar la primer etapa de pruebas:

1. Resolver el bug encontrado durante las pruebas que no permitía reservar el robot a utilizar.
2. Corregir errores gramaticales a lo largo de la guia.
3. Se decidió mejorar la interfaz de la guia, mostrando las capturas de pantalla dentro de un mock de dispositivo para fortalecer el atractivo visual del sitio.
4. Se cambió el manejo de robots durante la ejecución de código, en el caso de que haya solo un robot disponible se reservara ese robot sin consultar al usuario, con esto se logra agilizar la ejecución de código localmente.
5. Se decidió cambiar la actividad 3 de la unidad 2 por otra que necesite un poco más de ingenio.

7.3.3 Etapa 3: última prueba

Luego de introducir los cambios planteados en la etapa 2 se continuó con las pruebas, teniendo como últimas alumnas a Sofía y Julieta, dos niñas de 12 y 13 años respectivamente, las mismas se distinguieron del resto ya que llevaron a cabo la prueba las dos al mismo tiempo, compartiendo tanto la guía como la tablet en la que ejecutaron la aplicación DROPSY.

Quinta y sexta alumna: Sofía y Julieta
Sofía y Julieta, mostradas en la Figura 33, 34 y 35, son dos niñas de 12 y 13 años de edad respectivamente que, al igual que el resto de los participantes anteriores no poseían conocimientos previos en programación. En aproximadamente 80 minutos lograron completar correctamente las unidades 1, 2, 3 y parte de la 4.

- Nivel de comprensión de los conceptos introducidos en cada unidad:
    - Unidad 1: esta unidad es de carácter introductorio y las alumnas no tuvieron inconvenientes.
    - Unidad 2: las alumnas no tuvieron dificultades en entender los conceptos de movimiento introducidos en la unidad.
    - Unidad 3: en esta unidad las alumnas necesitaron una breve explicación para terminar de comprender el concepto de repetición ya que no lograban encontrarle la utilidad, pero luego de eso no tuvieron inconvenientes.
    - Unidad 4: en esta unidad se requirió un poco más de atención por parte del tutor ya que con sus tempranas edades les costó prestar atención a los textos, lo cual es fundamental en las últimas unidades. Se les dio una breve explicación del ejemplo planteado al final de la unidad. Luego de esto lograron entender la utilidad de las variables.
    - Unidad 5: no se evaluó.
- Completitud y corrección de las actividades planteadas:
    - Unidad 1: esta unidad no presenta actividades.
    - Unidad 2: las alumnas pudieron resolver todas las actividades en alrededor de 10 minutos. Solo necesitaron más de un intento para la actividad 2, pudiendo resolver tanto la actividad 1 como la nuevo actividad 3 en el primer intento.
    - Unidad 3: en alrededor de 25 minutos lograron resolver todos las actividades de la unidad, incluso la actividad 3 la cual tiene una dificultad más elevada.
    - Unidad 4: las alumnas dedicaron alrededor de 30 minutos a esta unidad, pudiendo en ese tiempo resolver la actividad 1 correctamente.
    - Unidad 5: no se evaluó.
- Nivel de entusiasmo del participante:
    - Unidad 1: el entusiasmo es sin dudas algo que incrementó notablemente en esta prueba y se debe al hecho de que las alumnas hicieron la prueba en pareja, esto

hizo que les resulte más divertido y entretenido ya que podían conversar sobre la prueba mientras la realizaban.

- Unidad 2: al incrementar la dificultad de las pruebas las alumnas seguían muy entusiasmadas pero mucho más concentradas, debatiendo las posibles resoluciones de las actividades entre ellas. Si bien la presencia del robot en la sala es atractivo para todos los alumnos, para estas alumnas de muy temprana edad resultó todavía más atractivo y las mismas se mostraron muy eufóricas al probar exitosamente programas en el robot.
- Unidad 3: en esta unidad las alumnas demostraron que el atractivo de este tipo de actividades incrementa al realizarlo en compañía ya que en más de una ocasión necesitaron concentrarse, debatir e intercambiar ideas para alcanzar las soluciones a las actividades, algo que hubiese sido mucho más difícil de lograr en caso de tomar la prueba individualmente.
- Unidad 4: las alumnas no perdieron el entusiasmo y continuaban muy entretenidas por las pruebas pero lamentablemente debieron parar porque tenían otros compromisos.
- Unidad 5: no se evaluó.

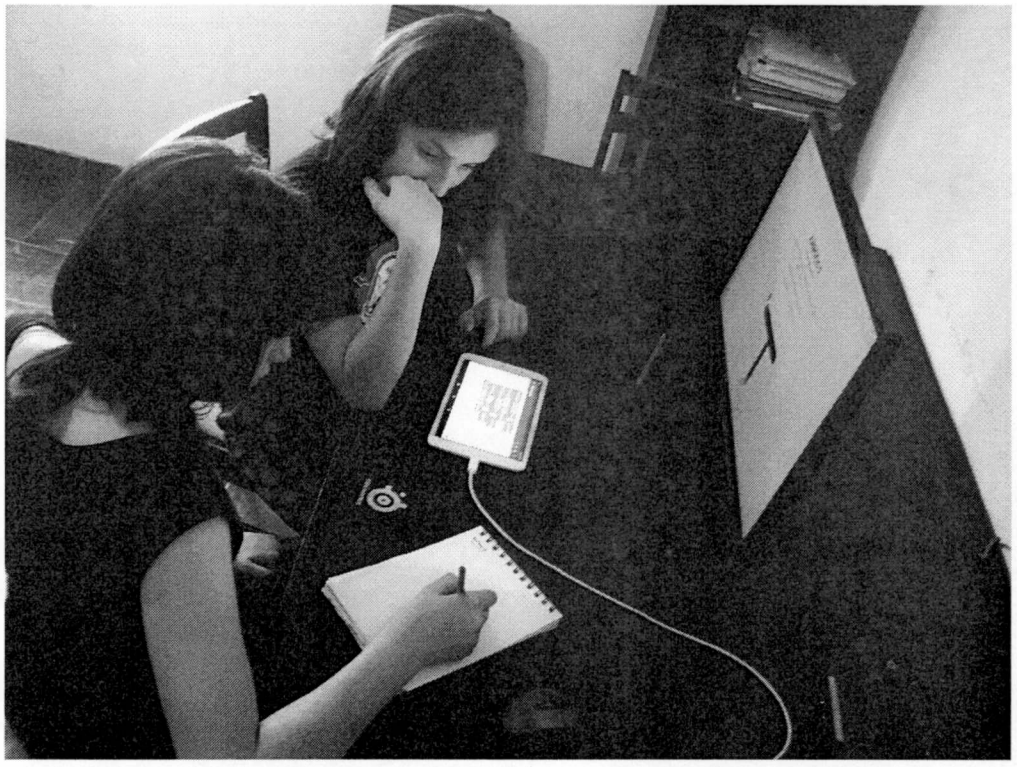

Figura 33 - Quinta y sexta alumnas: Sofía y Julieta

Figura 34 - Quinta alumna: Sofía

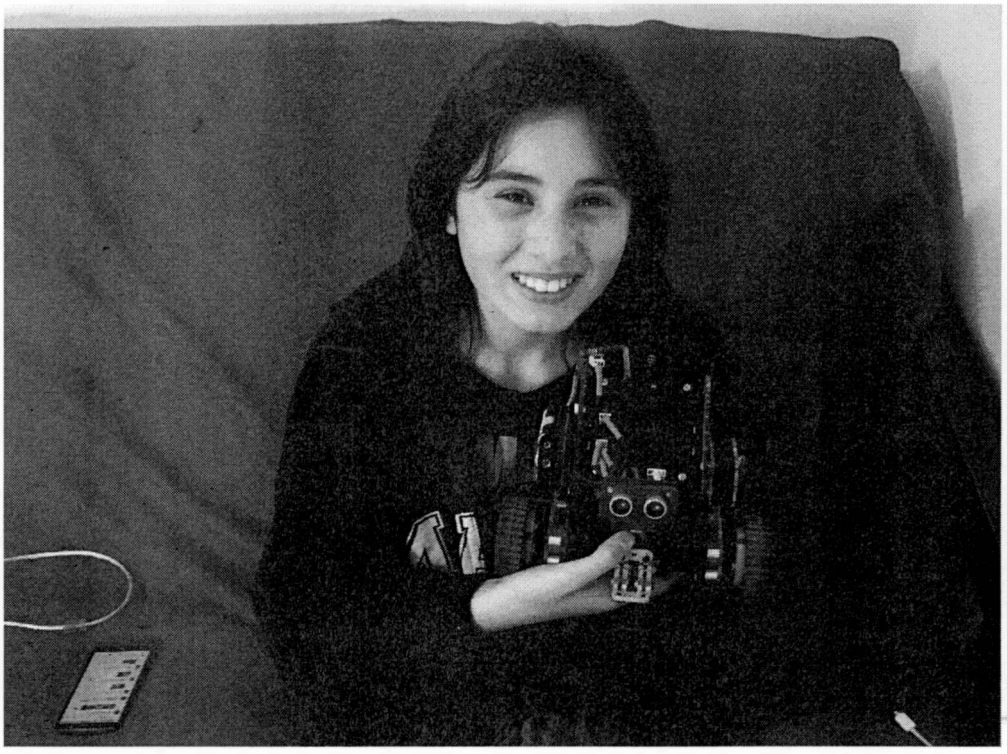

Figura 35 - Sexta alumna: Julieta

7.4 Conclusiones sobre las pruebas

Al finalizar la tercer etapa de prueba se dieron por concluidas las pruebas, las cuales fueron consideradas exitosas ya que permitieron, en mayor o menor medida, introducir a personas completamente ajenas a la programación en los conceptos básicos de la misma. a partir de las cuales podemos hacer las siguientes menciones:

1. Se trabajó con un número reducido de alumnos/as por limitaciones en el tiempo de finalización de esta tesina, pero aunque las pruebas individuales o de a pares no se ajustan por completo a las posibles aplicaciones que tendrá la aplicación en un futuro, creemos que la realización de las mismas fue muy útiles ya que nos permitió encontrar problemas tanto en la aplicación desarrollada como en la guía de actividades los cuales, una vez resueltos, mejoran la experiencia de los/as alumnos/as indistintamente del entorno donde se utilice el sistema.
2. Se observó un gran entusiasmo por parte de los/as alumnos/as por poder corroborar utilizando el robot la correcta utilización de los conceptos introducidos en cada unidad. También, se mostró una gran rapidez de los/as alumnos/as para familiarizarse con el manejo de bloques y con la aplicación. Consideramos que el hecho de que la aplicación está desarrollada para ser ejecutada en tablets y, por lo tanto, pueda ser utilizada sin necesidad de usar periféricos, es un factor importante en esta facilidad de aprendizaje.
3. Como era de esperarse, debido a su alta complejidad, algunas unidades de la guia no pudieron ser probadas tanto como otras. Esto nos permite profundizar en la idea de que la guía de actividades propuesta es una primera versión, debiendo la misma probarse extensamente y corregirse antes de que pueda considerarse completada. Es por esto que se está programando una prueba más masiva y de mayor duración con niños de escuelas primarias y secundarias en el marco del proyecto "Programando con Robots y Software Libre".

Capítulo 8

Conclusiones

Esta tesina extiende y complementa la herramienta XremoteBot, desarrollada en el marco de la tesis de Fernando López en el año 2015, y concluye con el desarrollo de la plataforma DROPSY como una herramienta dedicada a enseñanza de programación en escuelas primarias y secundarias.
El trabajo incluye la aplicación cliente móvil DROPSY como herramienta central y punto de contacto entre el usuario y los robots, adaptaciones del servidor XRemoteBot que permitan integrarlo a los nuevos desarrollos y optimizarlo para incrementar su aplicabilidad y, la nueva solución de streaming enfocada en el cliente móvil DROPSY y en general a cualquier cliente en Internet.
También se elaboró la guía de actividades "Programando con DROPSY" a la que consideramos parte fundamental del proyecto DROPSY, ya que representa un punto de partida interesante para la utilización del mismo y permitirá que la herramienta sea difundida y compartida.

Como DROPSY se conforma como una plataforma completa y no una serie de desarrollos separados, fue necesario poner a prueba de alguna manera el funcionamiento en conjunto de todas las herramientas, obtener resultados medibles y realizar los ajustes necesarios que posibiliten luego la transición a un uso más general de la plataforma. Es por este motivo que se realizaron una serie de pruebas de utilización de la herramienta en un contexto informal, pero no por eso menos valioso, que nos proporcionaron información muy valiosa y la retroalimentación necesaria para mejorar DROPSY y por qué no, pensar en nuevas funcionalidades que le den aún más valor. Fue durante el transcurso de estas pruebas que pudimos descubrir algunas situaciones que derivaron en mejoras; pero además nos dieron un nuevo punto de vista respecto a la manera en que los/as alumnos/as y participantes interactuaron con DROPSY y los robots:

- Contar con un robot físico que convierte nuestros programas en acciones en el mundo real fue definitivamente un gran estímulo, como pudimos comprobar con la totalidad de los participantes que manifestaron su entusiasmo al ver al robot realizar las acciones que ellos mismos habían definido.
- Utilizar una tablet o un celular para interactuar con DROPSY significó ventajas adicionales, permitiendo al alumno/a conservar la movilidad y acercarse al robot cuanto desee, todo ello sin perder contacto con la aplicación. Además facilitó el empleo de un

dispositivo adicional como una notebook para acceder a la guia "Programando con Dropsy" mientras se realizaban las actividades en el cliente móvil.

- Organizar los grupos de trabajo de a pares aumentó en gran medida el entusiasmo de los/as alumnos/as, convirtiendo cada actividad en un desafío y motivándolos/as para seguir avanzando. Además facilitó la comprensión de cada concepto introducido, al abrir un espacio de debate e intercambio de ideas y fomentando el trabajo en equipo y la colaboración para alcanzar un objetivo común.

Todas estas observaciones sumadas a nuestras experiencias personales y a los antecedentes mencionados a lo largo del presente trabajo nos convencieron del potencial de DROPSY como herramienta didáctica y en general del uso de robots educativos para acercar la enseñanza de la programación a la escuela.

Referencias Bibliográficas

[1] López, F. E. M. (2015). XRemoteBot: un servicio para programar robots en forma remota. Recuperado de http://sedici.unlp.edu.ar/handle/10915/51032

[2] Berners-Lee, T. (2013). Programming ability is the new digital divide. Computerworld, Inc. Recuperado de http://www.computerworld.com/article/2494661/app-development/programming-ability-is-the-new-digital-divide--says-berners-lee.html

[3] Resnick, M. (2008). Sowing the Seeds for a More Creative Society. Learning & Leading with Technology. Recuperado de http://web.media.mit.edu/~mres/papers/Learning-Leading-final.pdf

[4] Díaz, J., Banchoff Tzancoff, C., Queiruga, C. y Martín, E. (2014). Experiencias de la Facultad de Informática en la Enseñanza de Programación en Escuelas con Software Libre. Congreso Iberoamericano de Ciencia, Tecnología, Innovación y Educación. Recuperado de http://www.oei.es/congreso2014/memoriactei/1426.pdf

[5] Harel, I. y Papert, S. (Eds) (1991). Constructionism. Ablex Publishing Corporation. Norwood, NJ. Recuperado de http://www.papert.org/articles/SituatingConstructionism.html

[6] Gobierno Argentino. Consejo Federal de Educación (2015). Prioridades y metas del ministerio de educación de la nación y los ministerios jurisdiccionales en el marco de la intensificación del uso de TIC en las escuelas para la mejora de los procesos de enseñanza y aprendizaje. Anexo resolución CFE N° 244/15. Recuperado de http://www.me.gov.ar/consejo/resoluciones/res15/244-15_01.pdf

[7] Wing, J. M. (2006). Computational Thinking. Carnegie Mellon School of Computer Science. Recuperado de https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf

[8] Clements, D. H., Battista, M. T. y Sarama, J. (2001). Logo and Geometry. Journal for Research in Mathematics Education. Recuperado de http://www.jstor.org/stable/749924

[9] Clements, D. H. y Swaminathan, S. (1995). Technology and School Change New Lamps for Old?. Childhood Education, Volume 71. Recuperado de http://www.tandfonline.com/doi/abs/10.1080/00094056.1995.10522619

[10] Papert, S. (1999). Ghost in the Machine: Seymour Papert on How Computers Fundamentally Change the Way Kids Learn. ZineZone. Recuperado de http://www.papert.org/articles/GhostInTheMachine.html

[11] Heese B. (2014). Why Teach Computer Science in High Schools?. The Huffington Post. Recuperado de http://www.huffingtonpost.com/brian-heese/why-teach-computer-science_b_4350315.html

[12] Naughton, J. (2012). A manifesto for teaching computer science in the 21st century. Guardian News and Media Limited. Londres. Recuperado de https://www.theguardian.com/education/2012/mar/31/manifesto-teaching-ict-education-minister