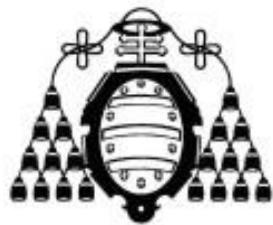


UNIVERSIDAD DE OVIEDO



ESCUELA DE INGENIERÍA INFORMÁTICA

PROYECTO FIN DE CARRERA

PROGRAMACIÓN DEL ROBOT ZOWI PARA SU MANEJO Y USO EN
AULAS DE EDUCACIÓN INFANTIL

DIRECTOR: María del Puerto Paule Ruíz

CODIRECTOR: Claudio López Ardura

PAULE RUIZ MARIA PUERTO - 09402054E	Firmado digitalmente por PAULE RUIZ MARIA PUERTO - 09402054E Fecha: 2017.11.23 10:06:17 +01'00'
--	--

**Vº Bº del Director del
Proyecto**

AUTOR: Marco Martínez Ávila

Agradecimientos

A ti. Por tu ayuda. Pero especialmente, por todo lo demás.

Resumen

Este Trabajo Fin de Máster consiste en la **personalización del robot Zowi de BQ, así como en el desarrollo de una aplicación Android, para su uso y manejo en las aulas de Educación Infantil**. Se pretende conseguir un prototipo que, en una etapa inmediatamente posterior, será utilizado en la práctica por una maestra de Gijón. Por lo tanto, algunos de los requisitos del Proyecto se han establecido teniendo en cuenta sus necesidades.

El trabajo realizado se puede dividir en dos secciones bien diferenciadas, siendo la primera crucial de cara a conseguir un verdadero aprovechamiento del Proyecto: **el diseño y contextualización de las actividades docentes**. Esta etapa consiste en, **con la ayuda de profesionales del sector, decidir cómo serán los ejercicios** que los niños tendrán que resolver: qué habilidades potenciar, regular el grado de dificultad o decidir las temáticas a tratar son algunos de los principales aspectos a tener en cuenta.

La importancia de este análisis radica en que el sistema, independientemente de las tecnologías o los recursos utilizados, debe **satisfacer la necesidad básica implícita en el ámbito en el que se hará uso: enseñar**.

La segunda sección consiste en el desarrollo del software y modificación del hardware necesarios para llevar a cabo el objetivo propuesto. Para ello, **se dispondrá tanto del robot Zowi -conjunto de sensores y actuadores basados en Arduino-, como de una aplicación Android** que será ejecutada en una tablet. Los alumnos podrán acceder al catálogo de ejercicios e interactuar con el robot, fomentando así el proceso de aprendizaje al contar con recursos atípicos y llamativos.

La principal necesidad a satisfacer es la **escasa utilidad didáctica del software proporcionado por la empresa española**. Este permite conocer el producto en su totalidad, interactuar con él y practicar algunos juegos, pero carece de las características necesarias para que los niños aprendan. **No motiva el aprendizaje**, y más allá de una grata sorpresa inicial, se aburren de él con relativa facilidad e impide que se aproveche todo el potencial del conjunto.

Palabras Clave

Zowi, Robótica, Aplicación Android, Uso en aulas de Educación Infantil, Aprendizaje, Tecnología como medio

Índice general

1. Memoria del Proyecto	13
1.1. La robótica como medio, no como fin	13
1.1.1. La robótica en el ámbito de la Educación	14
1.2. Resumen de la Motivación, Objetivos y Alcance del Proyecto	15
1.2.1. Motivación	15
1.2.2. Objetivos y alcance	15
1.3. Resumen de las secciones del Proyecto	17
1.3.1. Introducción	17
1.3.2. Análisis técnico de Zowi y la aplicación Android	17
1.3.3. Planificación del Proyecto y resumen de presupuestos	17
1.3.4. Estudio y análisis del software de fábrica	17
1.3.5. Análisis de ZowiApprende	17
1.3.6. La nueva ZowiApprende, aprendiendo con Zowi	18
1.3.7. Ampliaciones del hardware y software de Zowi	18
1.3.8. Personalización de las piezas de Zowi mediante impresión 3D	18
1.3.9. Diseño de ZowiApprende	18
1.3.10. Pruebas y manuales	18
1.3.11. Conclusiones	19
2. Introducción	21
2.1. Justificación del Proyecto: ¿está Zowi a la altura?	21
2.2. Objetivos y requisitos del Proyecto	22
2.3. Estudio de la situación actual	23
2.3.1. Bee-Bot	23
2.3.2. Codi-Oruga	24
2.3.3. Pleo V2 Reborn	24
2.3.4. Emiglio	25
2.3.5. ScratchJr	26
2.3.6. Otras tecnologías utilizadas en el contexto escolar	27
2.3.7. ¿Se adapta Zowi a las exigencias de la competencia?	28
2.3.8. Tabla comparativa de robots	29
3. Análisis técnico de Zowi y la aplicación Android	31
3.1. Zowi, el robot open source	31
3.1.1. Robot de código abierto	31
3.1.2. Placa base	32

3.1.3. Sensores	33
3.1.4. Actuadores	34
3.1.5. Pines digitales y analógicos	34
3.1.6. Carcasas 3D	35
3.1.7. Lenguaje de programación Arduino y Bitblock	36
3.2. Aplicación Android	38
3.2.1. Lenguaje de programación Java	38
3.3. Ampliaciones sobre el hardware	38
4. Planificación del Proyecto y Resumen de Presupuestos	41
4.1. Planificación	41
4.1.1. Procesos previos al desarrollo	46
4.1.2. Estudios previos	46
4.1.3. Desarrollo	46
4.1.4. Pruebas	46
4.2. Resumen del presupuesto	47
5. Estudio y análisis del software de fábrica	49
5.1. Zowi y su comportamiento por defecto, la base de todo su potencial	49
5.1.1. Matriz de LEDs	49
5.1.2. Mecanismo para mover las piernas y algoritmo para caminar	52
5.1.3. Algoritmo para caminar	54
5.2. ZowiApp. ¿Está la aplicación a la altura del robot?	57
5.2.1. Comunicación de Zowi con la aplicación Android	58
5.2.2. Recepción de comandos en Zowi	60
6. Análisis de ZowiApprende	63
6.1. Definición del sistema	63
6.1.1. Aplicación Android	63
6.1.2. Programa Arduino	64
6.2. Requisitos	64
6.2.1. Requisitos ZowiApprende	64
6.2.2. Requisitos Zowi	64
6.3. Identificación de actores del sistema	64
6.4. Identificación de los subsistemas	65
6.4.1. Descripción de subsistemas	65
6.4.2. Descripción de las interfaces entre subsistemas	66
6.5. Especificación de casos de uso y escenarios	67
6.6. Análisis de interfaces de usuario	70
6.6.1. Pantalla de selección de tipo de juego	70
6.6.2. Juego libre	71
6.6.3. Juego dirigido	71
6.6.4. Actividad de cuadrícula	72
6.6.5. Actividad de bloques lógicos	73
6.7. Diagramas de navegabilidad	74
6.8. Definición del plan de pruebas	74

6.8.1. Pruebas unitarias	74
6.8.2. Pruebas de integración, de sistema y de usabilidad	75
7. La nueva ZowiApprende, aprendiendo con Zowi	77
7.1. Nuevo contenido, pensado para enseñar	77
7.2. Aleatoriedad de contenido	78
7.3. Sección de juego libre	79
7.4. Sección de juego dirigido	79
7.4.1. Columnas	81
7.4.2. Arrastrar al contenedor	81
7.4.3. Semillas	82
7.4.4. Pirámide de alimentación	82
7.4.5. ¡Los ojos de Zowi!	83
7.4.6. Operaciones matemáticas	83
7.4.7. Memory	84
7.4.8. ¡Música, maestro!	85
7.4.9. Puzzle	86
7.4.10. ¡Guía a Zowi a su casa!	86
7.4.11. Cuadrícula de colores	86
7.4.12. Cuadrícula	87
7.4.13. Ventanas emergentes	87
8. Ampliaciones del software de Zowi	89
8.1. <i>setup</i> y <i>loop</i> de Arduino	89
8.2. Bocas, gestos, movimientos y bailes	89
8.3. Funciones para corregir actividades	90
8.4. Guiar a Zowi a su casa mediante el sensor de ultrasonidos	90
8.5. Sonar	91
8.6. Ritmo de la música	91
8.7. Girar y caminar recto	92
8.8. Operaciones en la boca	92
8.9. Tabla resumen de comandos	95
8.10. Nuevo mecanismo para caminar	95
8.10.1. Guía de electroimanes	96
8.10.2. Uso de acelerómetro y giroscopio	96
8.10.3. MPU 6050	100
8.10.4. MPU 9250: uso de acelerómetro, giroscopio y magnetómetro	103
8.10.5. Transformación de los valores a ángulos Euler	107
8.10.6. Creación del nuevo algoritmo para caminar	109
8.10.7. Sensor de infrarrojos... En la cabeza	111
8.10.8. Implementación definitiva	112
9. Diseño de ZowiApprende	115
9.1. Diagramas de paquetes	115
9.1.1. Aplicación Android	116
9.1.2. Código Arduino	117

9.2. Diagramas de clases	119
9.3. Diagramas de componentes	126
9.3.1. Componentes del sistema	126
9.3.2. Componentes de la aplicación Android	127
9.3.3. Componentes del código Arduino	131
9.4. Diagramas de secuencia	134
9.5. Diagrama de despliegue	138
9.5.1. Tablet	138
9.5.2. Zowi	138
10. Desarrollo de las pruebas	141
10.1. Pruebas de ZowiApprende	141
10.1.1. Pruebas del menú principal	141
10.1.2. Pruebas de la sección de juego libre	143
10.1.3. Pruebas de la sección de juego dirigido	145
11. Implementación del sistema	153
11.1. Alternativas tecnológicas y software utilizado	153
11.1.1. Sistema operativo	153
11.1.2. Lenguaje de programación	153
11.1.3. IDE	154
11.1.4. Otro software	154
11.2. Manual de instalación	155
11.3. Problemas y obstáculos enfrentados	157
11.3.1. Carga de contenido	157
11.3.2. Evento TouchEvent en imágenes	159
11.3.3. Carga de imágenes asíncrona con Picasso	159
11.3.4. Configuración de IMUs	160
11.3.5. Hilos bluetooth	161
11.3.6. Personalización de las piezas de Zowi mediante impresión 3D	162
11.3.7. Adaptación a requisitos de edades tan tempranas	164
11.3.8. Mejora del algoritmo de caminar	164
11.4. Objetivos no alcanzados	167
11.4.1. Mejora del algoritmo de caminar	167
12. Conclusiones	169
12.1. Posibles mejoras	169
12.1.1. Más actividades de juego libre	169
12.1.2. Uso de imágenes libres	169
12.1.3. Optimización en la carga y manejo de imágenes	170
12.1.4. Registros de la actividad de los alumnos y uso de perfiles de usuario	170
12.1.5. Soporte a sistemas operativos modernos	171
12.1.6. Pruebas en colegios	172
12.1.7. Subida de ZowiApprende a la Play Store	172
12.2. Reflexión final	172

Índice de figuras

2.1.	Forma del Bee-Bot.	23
2.2.	Botones disponibles en la parte superior del Bee-Bot.	24
2.3.	Cuerpo del robot Codi-Oruga.	24
2.4.	Codi-Oruga junto a las placas que determinan sus movimientos.	25
2.5.	Pleo V2 Reborn, el robot con forma de dinosaurio.	25
2.6.	Emiglio, el robot mayordomo.	26
2.7.	Pantalla de la aplicación ScratchJr para Android.	26
2.8.	Pizarra Digital Interactiva (PDI).	27
3.1.	Arduino UNO con conectores hembra.	32
3.2.	Placa similar a la de Zowi con conectores macho.	32
3.3.	Conector Dupont, cuyo uso es necesario para la conexión de jumpers a la placa de Zowi.	33
3.4.	Botones disponibles en la espalda de Zowi.	33
3.5.	Posibles personalizaciones de la cabeza de Zowi.	35
3.6.	Diseño de un gorro con forma de gota para Zowi.	36
3.7.	Código de ejemplo “programado” con Bitbloq -con su equivalente en Arduino más adelante-.	37
3.8.	Sección Gamepad de ZowiApp.	38
4.1.	Primera parte del diagrama de Gantt	42
4.2.	Segunda parte del diagrama de Gantt	43
4.3.	Tercera parte del diagrama de Gantt	44
4.4.	Cuarta parte del diagrama de Gantt	45
5.1.	Formas mostradas en la matriz de LEDs de Zowi.	50
5.2.	Posición de los servos desde el perfil de Zowi.	52
5.3.	Posición de reposo de Zowi.	53
5.4.	Esquema de funcionamiento de los servos de la cadera	53
5.5.	Esquema de funcionamiento de los servos de los pies	54
5.6.	Seno con amplitud 1 y período 2π	55
5.7.	Funciones sinusoidales correspondientes a las caderas y a los pies.	56
5.8.	Ángulo -valor de la función sinusoidal- que provoca el movimiento de Zowi hacia delante o hacia atrás.	57
5.9.	Valores absolutos de los servos en el algoritmo de caminar.	58
5.10.	Flujo de comandos Bluetooth enviados a Zowi esde ZowiApp.	59

5.11.	Código de ejemplo escrito por Arduino en el puerto serie.	59
6.1.	Agrupación de los casos de uso de ZowiApprende.	68
6.2.	Pantalla de selección de tipo de juego.	71
6.3.	Pantalla de selección de actividad en juego libre.	71
6.4.	Pantalla de selección de tema en juego dirigido.	72
6.5.	Pantalla de selección de actividad en juego libre.	72
6.6.	Actividad de la cuadrícula (modo sencillo).	73
6.7.	Actividad de la cuadrícula (modo difícil).	73
6.8.	Actividad de bloques lógicos.	74
6.9.	Diagrama de navegabilidad de ZowiApprende.	74
7.1.	Menú principal de la nueva ZowiApp.	78
7.2.	Actividad de la cuadrícula de colores.	78
7.3.	Actividad de la cuadrícula de colores con distinto contenido.	79
7.4.	Menú de la sección de juego libre.	79
7.5.	Menú de la sección de juego dirigido.	80
7.6.	Actividad de las columnas.	81
7.7.	Actividad de arrastrar a los contenedores inferiores.	81
7.8.	Actividad de arrastrar a los contenedores inferiores.	82
7.9.	Actividad de las semillas.	82
7.10.	Actividad de la pirámide alimenticia.	83
7.11.	Actividad de los ojos de Zowi.	83
7.12.	Actividad de las operaciones (modo sencillo).	84
7.13.	Actividad de las operaciones (modo difícil).	84
7.14.	Actividad del Memory con las imágenes de cara.	85
7.15.	Actividad del Memory con las imágenes volteadas.	85
7.16.	Actividad de los dictados musicales.	85
7.17.	Actividad del puzzle.	86
7.18.	Actividad en la que se guía a Zowi a su casa.	86
7.19.	Actividad de la cuadrícula.	87
7.20.	<i>Progress Dialog</i> que se muestra al buscar a Zowi usando el Bluetooth.	87
7.21.	<i>Alert Dialog</i> que se muestra cuando no se puede conectar con Zowi.	88
7.22.	Feedback recibido al completar con éxito una actividad.	88
7.23.	Ventana mostrada en caso de excepción por <i>NullPointerException</i>	88
8.1.	Esquema de las mediciones de un acelerómetro.	97
8.2.	Direcciones y sentidos del acelerómetro en un smartphone (los ejes y su sentido podrían cambiar).	98
8.3.	<i>Drift</i> en las medidas del giroscopio de un MPU6050.	101
8.4.	<i>Drift</i> en las medidas del giroscopio de un MPU6050 inmóvil en una superficie plana.	102
8.5.	Valores devueltos por el giroscopio del MPU6050 sin calibrar.	102
8.6.	Datos devueltos por el giroscopio del MPU6050 calibrado.	103
8.7.	Datos devueltos por un acelerómetro al girar lentamente sobre el eje Z.	103
8.8.	Valores obtenidos por el magnetómetro desplazados con respecto al (0,0).	105

8.9.	Valores obtenidos por el magnetómetro con forma ligeramente elipsoidal.	105
8.10.	Valores obtenidos por el magnetómetro no calibrado proyectados en los 3 planos del sistema de coordenadas cartesianas.	106
8.11.	Valores obtenidos por el magnetómetro calibrado proyectados en los 3 planos del sistema de coordenadas cartesianas.	107
8.12.	Valores obtenidos por el magnetómetro calibrado.	107
8.13.	Esquema explicativo de los ángulos yaw, pitch y roll.	108
8.14.	Dirección y sentido de los ejes del acelerómetro y el giroscopio en el MPU9250.108	
8.15.	Dirección y sentido de los ejes del magnetómetro en el MPU9250.	109
9.1.	Diagrama de paquetes de la aplicación Android.	115
9.2.	Diagrama de paquetes del código Arduino.	116
9.3.	Esquema de las clases abstractas.	119
9.4.	Esquema de las clases utilizadas en todo el Proyecto.	120
9.5.	Comienzo de la aplicación y selección de actividad.	121
9.6.	Cinco de los doce tipos de actividades.	122
9.7.	Cinco de los doce tipos de actividades.	123
9.8.	Dos de los doce tipos de actividades.	124
9.9.	Simplificación de las clases Arduino implementadas en Zowi.	125
9.10.	Diagrama de componentes del sistema.	126
9.11.	Diagrama de componentes de la aplicación Android.	128
9.12.	Diagrama de componentes del código Arduino.	132
9.13.	Conexión con Zowi por medio de un <i>receiver</i>	135
9.14.	Conexión con Zowi una vez almacenada su dirección.	136
9.15.	Envío de comandos desde ZowiApprende.	137
9.16.	Desconexión de Zowi.	137
9.17.	Diagrama de despliegue de ZowiApprende.	138
10.1.	Menú principal de ZowiApprende con el código ColorADD.	142
11.1.	Habilitación de los “Orígenes desconocidos” en Android.	156
11.2.	“ZowiApprende.apk” en un explorador de archivos.	156
11.3.	Instalación de una aplicación externa al Google Play Store.	157
11.4.	IMU 9DOF Octopus.	161
11.5.	Nuevas orejas de Zowi.	163
11.6.	Soporte para el sensor dentro de las orejas de Zowi.	164
11.7.	Gorra y disposición de los sensores de infrarrojos.	165
11.8.	Gorra y disposición de los sensores de infrarrojos.	165
12.1.	Pantalla de selección de usuario de ZowiApprende.	170
12.2.	Pantalla de selección de más usuarios.	171
12.3.	Popup para añadir más usuarios.	171

Capítulo 1

Memoria del Proyecto

1.1. La robótica como medio, no como fin

La robótica es un ámbito de la tecnología que lleva años desarrollándose y evolucionando, **consistiendo en el diseño, la construcción y el uso de robots**. No hace mucho tiempo que este concepto sólo era imaginable a través de la gran pantalla u obras literarias, pero el avance tanto de las capacidades de ingeniería, como una mayor experiencia y conocimientos al respecto, han provocado que **actualmente prácticamente todos tengamos la posibilidad de interactuar con robots en nuestro día a día**.

Los sectores en los que estos dispositivos bípedos, cuadrúpedos, con ruedas o incluso con hélices pueden verse siendo utilizados son cada vez más diversos. **Un laboratorio fue el punto de partida** cuando apenas se disponían de medios para llevar a cabo un proyecto funcional, y **siempre se ha hecho mucho hincapié en todo aquello que el humano no puede conseguir**: trabajos extremos, guerras, rescates en condiciones demasiado duras para nosotros, etc. Este punto de vista se ha visto claramente reflejado en los cines, donde producciones como Blade Runner muestran esta faceta del objetivo final de la tecnología.

Independientemente de la moralidad de estos actos, y sin entrar en temas tan complejos como la inteligencia artificial, **la realidad del siglo XXI es bastante más simple** y, en numerosos casos, más cotidiana. **Un campo tan complejo y ajeno como la robótica se ha ido acercando al usuario**, pasando de requerir conocimientos técnicos amplísimos o grandes sumas de dinero, a ser accesible por todos, en prácticamente todo: la cocina, la limpieza, la domótica, los drones...

Los robots han pasado a formar parte de nuestra vida, para bien o para mal. E indudablemente, este acercamiento al gran público dota a este sector de un potencial mucho mayor, lo que unido a la competencia existente en el mercado y al abaratamiento de costes resulta en multitud de proyectos de incuestionable interés. No sólo eso, sino que se ha producido **un cambio de paradigma** que, aunque a priori pueda parecer minúsculo e incluso confuso, tiene una importancia mayúscula: **la robótica como medio, en lugar de como fin**.

En la primera etapa de investigación del sector se evalúan posibilidades, se establecen límites, se explora, se experimenta, se descubre y se llevan a cabo tareas por primera vez. La robótica es aquí un fin último, ya que el objetivo no es otro sino saber más sobre ella.

Pero igual de llamativo es el hecho de que la robótica sea un medio. Un utensilio, **una**

herramienta en nuestras manos que nos permita conseguir un objetivo distinto de forma satisfactoria. En los ejemplos expuestos anteriormente, se han mencionado los trabajos extremos, la guerra o la limpieza, pero este Trabajo Fin de Máster tiene como contexto un ámbito que debería considerarse de más importancia aún: la Educación. El uso de robots -medio- con el objetivo -fin- de aprender.

1.1.1. La robótica en el ámbito de la Educación

La Educación se ha llevado a cabo prácticamente desde que el ser humano tiene uso de razón, ya sea en las escuelas o mediante el aprendizaje cotidiano y basado en nuestras propias experiencias.

La robótica en este ámbito consiste, por su parte, en un **recurso que pone en marcha técnicas metodológicas de renovación pedagógica aplicables en todas las etapas educativas.** Etapas en las que es fundamental valorar no sólo las características evolutivas estudiadas por autores tan clásicos como Piaget o Vigostky, sino también los gustos, curiosidades e intereses de los niños. **Adaptarse a sus necesidades** para poder ofrecerles una respuesta individualizada que se ajuste a sus hitos madurativos se convierte en un requisito imprescindible.

Este hecho tiene especial importancia en las etapas tempranas del aprendizaje, donde se comienzan a formar cualidades personales tan características e intransferibles como los valores o el comportamiento cívico; y es en la Educación Infantil donde se desarrolla todo lo relacionado con este Proyecto. Cabe destacar que **todos estos aspectos también pueden verse reflejados en la normativa vigente**, si bien no se va a exponer de forma demasiado extensa por no ser considerados aconsejables en una introducción clara y comprensible.

Partiendo de la Ley Orgánica para la Mejora de la Calidad Educativa (LOMCE), se puede corroborar que la finalidad de la Educación Infantil es contribuir al desarrollo integral de los niños, favoreciendo así el ámbito físico, social, cognitivo y afectivo de los mismos. Sin embargo, para ello es **labor ineludible de los docentes crear experiencias motivadoras**, en las que los protagonistas del proceso de aprendizaje puedan desarrollar las capacidades correspondientes a esa etapa. Cabe destacar, por ejemplo, que sean capaces de “desarrollar habilidades comunicativas en diferentes lenguajes y formas de expresión”.

Asimismo, el trabajo de dichas capacidades se podrá abordar desde las tres Áreas de experiencia y desarrollo infantil establecidas en el Real Decreto 1630/2006, por el que se establecen las **enseñanzas mínimas del segundo ciclo de Educación Infantil**: “Conocimiento de sí mismo y autonomía personal”, “Conocimiento del entorno” y “Lenguajes: expresión y comunicación”.

En este orden de ideas, es importante subrayar que **a partir del tercer área se impulsará el trabajo de la robótica** dentro del centro escolar. Además, este ámbito de desarrollo pretende **facilitar la relación del niño con el entorno** gracias a las múltiples vías de comunicación que tiene a su disposición, como el lenguaje verbal, artístico o el corporal, sin olvidar el lenguaje audiovisual y las nuevas tecnologías de la información y la comunicación. Por todo esto, es necesario destacar la **importancia que desde hace un tiempo están adquiriendo las TIC/TAC**: Tecnologías de la Información y la Comunicación aplicadas como nuevas Tecnologías de Aprendizaje y Conocimiento (TAC).

En otras palabras, el uso de la tecnología como recurso pedagógico.

Al hilo de lo expuesto, autores como Pere Marqués destacan en sus estudios **una amplia variedad de software educativo a partir del cual los alumnos pueden aprender jugando**. Son muchas las páginas web, blogs, cortos de animación y similares que despiertan sus sentidos y les ayudan a desarrollar destrezas perceptivas, motrices, comunicativas o sociales, mientras ponen en marcha valores tan importantes como el respeto o la igualdad.

La Educación está evolucionando, y lo hace constantemente. La compañía de los niños en las aulas ya no son sólo los juguetes, ya que **ahora tienen a su disposición pizarras interactivas, tablets o robots; elementos que permiten enfocar el objetivo educativo desde un ángulo distinto**, ofreciendo un amplio abanico de posibilidades que -bien usado- puede fomentar enormemente todas las cualidades ya descritas.

1.2. Resumen de la Motivación, Objetivos y Alcance del Proyecto

1.2.1. Motivación

La tecnología es una gran desconocida para muchos, una especie de misterio donde la cara visible -ordenadores, smartphones, robots- casi nunca desvela toda la complejidad -o sencillez- que hay detrás. La tecnología puede ser una caja negra cuyo uso se limite a una o dos tareas, o **un conjunto sincronizado y personalizable de mecanismos que funcionan en común para conseguir multitud de objetivos variados**.

Esta última perspectiva es aquella que un ingeniero debe ser capaz de explotar. Ser capaz de ver más allá de la apariencia, más allá de las funciones por defecto, para encontrar utilidades y poder sacarles todo su jugo a lo que otras personas ven como simples “aparatos”.

La robótica, por su parte, y entendida siempre como un medio, **es un aliciente muy adecuado para renovar aquello que se ha convertido en cotidiano**, en aburrido, aquello que ya no llama la atención. La robótica es, todavía, algo nuevo, un soplo de aire fresco que usado convenientemente puede contribuir a que muchas personas -en el contexto de este TFM, niños- aprendan.

Constituye, en definitiva, **un pequeño grano de arena que pretende mejorar a las personas**, conseguir que tengan un criterio propio y una personalidad digna del mundo en el que vivimos. Con componentes baratos -unas piezas impresas en 3D, una placa Arduino y varios actuadores y sensores- se pueden lograr grandes desarrollos, renovando un sector considerado por algunos anticuado y destacando así “la grandeza de las pequeñas cosas”.

1.2.2. Objetivos y alcance

Los conceptos de “robótica” y “Educación” se concretan en este Proyecto para dar lugar a un desarrollo conciso, cuyos requisitos, objetivos y alcance están bien definidos.

Se tomará como base el robot Zowi, ensamblado y distribuido por la empresa española BQ, para crear una aplicación Android que niños de entre 3 y

5 años puedan utilizar en las aulas. El flujo de uso habitual será acceder a distintas actividades docentes especialmente diseñadas para este TFM en una tablet, conectándose esta con el robot y permitiendo interactuar con él.

La aplicación estará dividida en **dos grandes secciones**:

- **Juego libre:** sección en la que se dispondrá de **ejercicios de temática variada**, que no tienen por qué seguir unas pautas definidas.
- **Juego dirigido:** sección en la que se organizarán los **ejercicios por temas**. La selección de los últimos se ha llevado a cabo **contextualizando la aplicación en el primer trimestre del curso escolar de Educación Infantil**. Esta etapa incluye temáticas que favorecen la acogida de los alumnos con unidades motivadoras y cotidianas.

Toda la interfaz estará adaptada a los requisitos establecidos por la edad de las personas que harán uso de la misma, consistiendo en líneas sencillas, botones grandes y primando la funcionalidad a los efectos visuales más complejos.

Como punto determinante a mencionar, **la nueva ZowiApp** (ZowiApprende, de aquí en adelante) **constituirá un prototipo funcional cuyo alcance llegará hasta la etapa previa a la publicación en el Google Play Store y al despliegue en las aulas**. Deberá cumplir los requisitos expuestos más adelante, y su funcionamiento deberá adecuarse a lo estipulado: **rendimiento y cantidad de actividades suficientes como para comenzar el período de pruebas reales en colegio**.

1.3. Resumen de las secciones del Proyecto

1.3.1. Introducción

En esta sección se expondrá, más detalladamente y con contenido más técnico, qué aporta Zowi como robot al ámbito de la educación. Además, se llevará a cabo un estudio del arte en el que se contextualizará en la situación actual de la robótica en los colegios de Educación Infantil.

No sólo eso, sino que **cobrarán especial importancia las alternativas existentes en el mercado**, que se compararán con lo que aporta Zowi recién sacado de la caja, y con los añadidos desarrollados en este Proyecto. De esta forma, se pondrá tener una idea de clara de cuáles son los pros y contras de este robot bípedo.

1.3.2. Análisis técnico de Zowi y la aplicación Android

Este apartado recogerá las bases técnicas y teóricas del funcionamiento de Zowi, centrándose más en aspectos como los componentes, los sensores, su placa base Arduino y lo que lo hace funcionar bajo la carcasa.

También se estudiará la aplicación Android proporcionada por BQ y la propia de este TFM diseñada para su manejo, así como la comunicación que tiene lugar entre ambas partes.

Esta sección es un buen punto de partida para comprender qué se ha construido exactamente sobre la infraestructura que puede comprar cualquier particular, permitiendo así que el lector diferencie entre funciones por defecto y nuevas características.

1.3.3. Planificación del Proyecto y resumen de presupuestos

Un proyecto no consiste únicamente -y en muchos casos, ni siquiera es lo más importante- en desarrollos y análisis técnicos. La organización, planificación y gestión previa y paralela al mismo es determinante en su finalización con éxito.

Por tanto, en esta sección se detallará aspectos evaluados y definidos que marcarán las pautas a seguir: **tareas a realizar, plazos previsto, hitos marcados... Así como un presupuesto** que sentaría las bases de los costes de este TFM.

1.3.4. Estudio y análisis del software de fábrica

Tras conocer detalladamente los aspectos técnicos tanto de Zowi como de su relación con la aplicación Android, se hará un recorrido por las funciones principales del primero. El objeto es **comprender y evaluar el comportamiento de fábrica**, de forma que se pueda valorar todo aquello que se construya sobre esta base.

1.3.5. Análisis de ZowiApprende

Una vez evaluado el software disponible de fábrica, **se hará un análisis de la nueva funcionalidad que se pretende implementar**, del código que se creará y de otros aspectos como los requisitos o los casos de uso.

1.3.6. La nueva ZowiApprende, aprendiendo con Zowi

Apartado correspondiente a la **nueva aplicación Android** diseñada para el manejo de Zowi. Como se ha comentado varias veces con anterioridad, **el objetivo es que los niños aprendan con ella** -y que se diviertan haciéndolo-.

La aplicación de BQ ofrece muchas posibilidades interesantes, pero carece del factor educativo que se pretende aprovechar en las aulas.

1.3.7. Ampliaciones del hardware y software de Zowi

Teniendo en cuenta la plataforma Arduino en la que se basa este robot, se llevarán a cabo ampliaciones tanto de hardware -nuevos sensores y/o actuadores- como de software -código que le permitirá adaptarse al comportamiento de la aplicación.

En este capítulo se concretará todos **aquellos aspectos relacionados con estos añadidos que supongan un cambio** con respecto al conjunto disponible recién sacado el producto de la caja.

1.3.8. Personalización de las piezas de Zowi mediante impresión 3D

La empresa española no sólo es la encargada de imprimir las piezas que componen a Zowi, sino que **proporciona todos los archivos fuente para su uso y modificación por parte de terceros**.

Este hecho abre las puertas a la personalización por parte de aquellos usuarios con conocimientos en el campo, siendo posible modificar la carcasa original para adaptarla las necesidades existentes.

En este Proyecto **se valorará la inclusión de complementos 3D, así como la elaboración de una nueva cabeza que permite el correcto acomodamiento de nuevos sensores en su interior.**

1.3.9. Diseño de ZowiApprende

Estudio detallado de la arquitectura final del sistema, plasmando todos los esquemas, diagramas y descripciones que sean necesarios para definir a nivel técnico qué se ha llevado a cabo a lo largo del ciclo de vida del Proyecto.

1.3.10. Pruebas y manuales

No menos importantes que el desarrollo troncal son las pruebas, ya que determinan si un proyecto ha conseguido alcanzar satisfactoriamente los límites que se habían establecido con anterioridad. Por lo tanto, **se detallará todo el proceso especificando tanto los diversos test, sus resultados y los comentarios aportados por una profesional del sector.**

También se proporcionará una **guía básica o tutorial de instalación**, para acercar el manejo del robot a aquellas personas menos familiarizadas con el ámbito tecnológico.

1.3.11. Conclusiones

Se expondrá una reflexión final sobre Zowi y, sobre todo, su utilidad y contribución en el sector de la Educación.

Este apartado se considera de especial importancia ya que condensará los datos, los conocimientos y la experiencia obtenidos a lo largo de varios meses, plasmando de forma comprensible las ventajas e inconvenientes encontrados en la totalidad del Proyecto.

Capítulo 2

Introducción

2.1. Justificación del Proyecto: ¿está Zowi a la altura?

Ya se ha explicado la importancia de la robótica en la educación, así como definido **la herramienta protagonista de este Proyecto: Zowi**.

Puede resultar curioso, no obstante, que **cualquier particular pueda adquirir este robot** en cadenas comerciales tan extendidas como Media Markt. No sólo eso, sino que la propia **BQ** -distribuidora del producto- **proporciona a través de Google Play Store una aplicación Android** que permite interactuar con él y hacer uso de prácticamente todas sus funciones.

¿Qué necesidad hay, entonces, de desarrollar otra aplicación? ¿Para qué personalizar a Zowi con nuevos sensores si viene dotado de fábrica con multitud de características llamativas? La respuesta es muy directa: **lo que hay no es suficiente**.

El software de la empresa española permite hacer que Zowi camine con distintos grados de velocidad, que en la matriz de LEDs que hace de boca se iluminen distintas sonrisas, que baile o que encadene movimientos a escoger por el usuario. Además, su cerebro -la placa base Arduino- tiene precargado un programa que permite detectar sonidos u obstáculos.

E indudablemente, estas características están muy bien y son de agradecer, ya que si no el partido que se le podría sacar a Zowi sería ínfimo; sin embargo, **no son actividades que contribuyan al aprendizaje del alumno**, sino meros pasatiempos destinados a saber qué se puede hacer con él. La experiencia con niños en las aulas ha demostrado que, **tras una etapa de sorpresa inicial, pronto dejan de sentir interés por este pequeño compañero de juegos**. No consigue engancharlos como para seguir utilizándolo, hecho motivado por la falta de contenido real con el que llevar a cabo tareas de aprendizaje. No deja de ser un juego que pronto se vuelve aburrido.

Con este Proyecto se pretende dar un giro de tuerca, **mejorar la experiencia de los niños proporcionándoles algo que se adecue a la evolución de sus conocimientos**. Y sobre todo, se pretende crear un desarrollo que les permita **aprender dentro de la temática del colegio**, un desarrollo que se integre con los contenidos y los temas impartidos en Educación Infantil y que no constituya un simple juego. La idea era, es y será divertirse aprendiendo, **utilizando la robótica como gancho para atraer su atención y haciendo que Zowi les ayude a mejorar**.

2.2. Objetivos y requisitos del Proyecto

Si bien en el apartado anterior ha analizado cuáles son la motivación y el fin -a nivel genérico- a conseguir, en este **se detallan de una forma más esquemática y precisa aquellos objetivos que marcan los pequeños hitos del TFM.**

Para definirlos, se ha tenido en cuenta que el producto final será utilizado por una maestra en situaciones reales y, por tanto, parte de los requisitos indispensables se han contemplado de cara a satisfacer sus necesidades.

- **Uso de Zowi:** aunque pueda parecer evidente, antes de decidir el uso definitivo de Zowi se valoraron otras alternativas -descritas más adelante-. No obstante, se concluyó que la utilización de este robot, y no otro, era lo más adecuado para la maestra.
- **Mejora del algoritmo para caminar:** Zowi cuenta por defecto con 3 velocidades a la hora de caminar: lenta, normal y rápida; y aunque esta función cumple con su cometido la mayoría de las ocasiones, se queda por debajo de lo esperado si se quiere ir un paso más allá.

Como se detallará en la sección [Algoritmo para caminar](#), **los motores que permiten los movimientos para caminar no son del todo precisos**, lo que implica que el robot se va torciendo poco a poco a medida que avanza. Este es un inconveniente que, dada una de las actividades sugeridas por la maestra, merece ser analizado y resuelto.

- **Ampliación de sensores:** se evaluarán las posibilidades de añadir sensores a la placa Arduino, **incorporando así nuevas funciones a las ya existentes por defecto**. Esta necesidad se debe a la intención de utilizar a Zowi en una actividad que requiere más precisión a la hora de caminar que la que nos proporciona por defecto.
- **Creación de una actividad en la que Zowi camine por una cuadrícula:** los dos anteriores objetivos tienen como finalidad la consecución de este. **La maestra requiere hacer uso de una cuadrícula de 3x3 ó 4x4 en una actividad**, donde el robot se sitúa en una de las celdas y su destino, en otra.

Para resolver el ejercicio, **el niño deberá ser capaz de indicar los pasos que tiene que seguir el robot**, evitando posibles obstáculos y aprendiendo conceptos como la orientación y la percepción de la distancia.

- **Aplicación Android con estética y estructura sencillas:** aunque inicialmente la idea de un desarrollador ajeno a este ámbito suele ser llevar a la práctica aquello que primero visualiza en su cabeza, la programación de una aplicación para niños choca con este concepto.

No se puede alardear de efectos visuales, ni pensar en listas desplegables o menús sobrecargados. La interfaz deberá ser simple, con botones y textos grandes, muchas imágenes y poco nivel de profundidad -pocos clics para llegar a una determinada actividad-.

- **Compatibilidad con Android 4.4:** la fragmentación es un problema conocido y recurrente en el sistema operativo de Google, pero no por ello deja de ser importante. **La maestra dispone de una tablet BQ Aquaris M10 con pantalla HD y Android 4.4 KitKat.** Por lo tanto, y dado que es con este dispositivo con el que se va a interactuar con Zowi, la aplicación será compatible con esta versión de sistema y superiores.

2.3. Estudio de la situación actual

Aunque Zowi constituye una buena alternativa en este ámbito, no es la única opción disponible en el mercado actual. Diferentes empresas llevan años presentado productos y comercializando robots educativos para niños, donde cada uno cuenta con distintas características y desarrolla diferentes competencias en los alumnos.

Partiendo de esta base, se ha llevado a cabo un estudio donde **se evalúan algunos de estos dispositivos tecnológicos**, comparando sus componentes, su estética y su rendimiento con el protagonista de este Proyecto.

2.3.1. Bee-Bot

Bee-Bot es un robot con forma de abeja, **cuyo principal objetivo es potenciar la lateralidad, la coordinación y la orientación espacial** en las aulas de Educación Infantil.



Figura 2.1: Forma del Bee-Bot.

Dispone de varios botones en la parte superior, como se puede ver en la Figura 2.2. Mediante una sucesión de pulsaciones en los mismos se programa el número de pasos, pudiendo combinar cuatro sentidos -adelante, atrás, izquierda y derecha-.

Se suele utilizar en conjunto con una cuadrícula situada en el suelo que sirve de guía, donde cada celda constituye un color, letra, número... Que los niños deben identificar para ubicar en ella al robot, como se puede ver en el vídeo [Robot infantil BEE-BOT. Aprendiendo letras y programar.](#)



Figura 2.2: Botones disponibles en la parte superior del Bee-Bot.

2.3.2. Codi-Oruga

Como su propio nombre indica, la forma de este robot se asemeja a la de una oruga. Consiste en una pieza central -la cabeza-, a la que se le conectan hasta ocho secciones distintas que hacen las veces de cuerpo.



Figura 2.3: Cuerpo del robot Codi-Oruga.

Cada una de estas provoca que Codi-Oruga se mueva en una dirección distinta; es decir, **cambiando su orden es posible modificar el recorrido del mismo** en función de las indicaciones situadas encima de cada pieza. De esta forma, **los niños trabajan habilidades como la creatividad y la improvisación**, dependiendo de ellos el trayecto del robot.

Cabe destacar que, además de lo mencionado, cuenta con dos placas que indican el inicio y el final del recorrido.

2.3.3. Pleo V2 Reborn

La más completa de las alternativas disponibles, **simula el comportamiento de un animal con la forma de un dinosaurio**.



Figura 2.4: Codi-Oruga junto a las placas que determinan sus movimientos.



Figura 2.5: Pleo V2 Reborn, el robot con forma de dinosaurio.

Cuenta con multitud de sensores y mecanismos de inteligencia artificial que le permiten tanto interactuar con el entorno como aprender. Los niños pueden acariciarlo, alimentarlo, cuidarlo o jugar con él, siendo cada acción importante de cara al desarrollo de su personalidad; a medida que pasa el tiempo, va creciendo con unas cualidades u otras, convirtiendo cada interacción en una experiencia única.

Las funciones del Pleo V2 son muy extensas, por lo que si se quiere profundizar más en cada una de ellas se puede visitar la página web www.juguetronica.com/pleo-v2-reborn.

En lo referente a los alumnos, este robot impulsa el conocimiento de las necesidades básicas, las emociones, las relaciones afectivas y el desarrollo de la resiliencia, a partir de la que son capaces de dar respuesta a los pequeños conflictos que viven en su día a día.

2.3.4. Emiglio

Emiglio es un robot humanoide que hace las funciones de un mayordomo, y aunque su aplicación es más bien lúdica, se trata de un recurso muy motivador con el que también es posible aprender.

Se puede controlar mediante un brazalete que el alumno se coloca en la muñeca, y dispone de distintas voces con las que puede repetir aquello que se le diga.

De esta manera, se trabajan contenidos como el esquema corporal, las relaciones interpersonales y se pone en marcha el juego simbólico, donde cada niño asume un rol elegido libremente.



Figura 2.6: Emiglio, el robot mayordomo.

2.3.5. ScratchJr

Aunque ScratchJr no es un robot ni un dispositivo electrónico como tal, tiene una estrecha relación con uno de los aspectos fundamentales de Zowi: su programación con Bitbloq.

Este es un software, o una especie de framework, que **permite crear historias interactivas de forma sencilla e intuitiva**. Los niños pueden arrastrar distintos elementos a la pantalla principal, para a continuación escoger una sucesión de acciones que llevarán a cabo. Por ejemplo, es posible configurar los pasos y los diálogos que tendrán lugar entre varios animales añadidos, así como escoger el entorno en el que tendrá lugar esta interacción.



Figura 2.7: Pantalla de la aplicación ScratchJr para Android.

Un punto a su favor es su amplia disponibilidad, ya que se puede utilizar desde los sistemas operativos Android, iOS y Windows.

2.3.6. Otras tecnologías utilizadas en el contexto escolar

Además de la robótica, **existen otras tecnologías modernas** que son aplicadas a día de hoy en todas las etapas de la Educación.

La televisión, el proyector o el cañón, por ejemplo, son herramientas fundamentales para potenciar la comunicación visual. No obstante, existe **un recurso tecnológico todavía más llamativo y muy utilizado en las aulas de Infantil: la pizarra digital interactiva (PDI)**.



Figura 2.8: Pizarra Digital Interactiva (PDI).

Con ella, se pueden trabajar actividades rutinarias como consultar el tiempo, repasar la asistencia a clase, elaborar dibujos, etc, aunque su uso no sólo se limita al propio dispositivo. **Se puede complementar la experiencia con aplicaciones de terceros**, como Notebook, con la que se pueden entrenar destrezas lógico-matemáticas; Las Regletas Digitales, para componer y descomponer cifras; o los avances proporcionados por la realidad aumentada, al dotar de vida a los dibujos de láminas elaboradas previamente.

Con un fin similar se utilizan las tablets infantiles, que si bien persiguen un objetivo parecido al de las PDIs, fomentan el trabajo autónomo de manera individual o en pequeños equipos.

Al hilo de lo expuesto, **aplicaciones como Voki o Simbaloo inciden en la alfabetización digitalizadora**. La primera permite crear avatares correspondientes a cada integrante de la clase, mientras que la segunda convierte el escritorio del ordenador en un tablero de enlaces directos. Así, el acceso a recursos como páginas con cuentos, canciones, visitas a museos virtuales, etc, es más cómodo, rápido e intuitivo.

Para acabar, cabe destacar un aspecto tan importante como **la atención a la diversidad, denominador común en las aulas**, donde es necesario proporcionar una respuesta adecuada a las características de cada alumno. Para ello, se puede hacer uso de opciones como las mostradas a continuación:

- **Programas como el Pequeabecedario:** dirigido al alumnado con discapacidad auditiva, impulsando la **adquisición de vocabulario mediante el apoyo de imágenes y la traducción al lenguaje de signos**.

- Técnicas como el implante coclear, audífonos, amplificadores de mesa, lupas, la máquina Perkins, el Pilot traductor...
- **Tiflotecnología:** se utiliza para describir las TIC adaptadas a la discapacidad visual, ya que **permite acceder a la información y a recursos como libros de texto en igualdad de condiciones.** También posibilita su guardado de forma sencilla y manejable.

2.3.7. ¿Se adapta Zowi a las exigencias de la competencia?

Robots

Tras hacer un breve repaso a algunas de las alternativas disponibles en el mercado, merece la pena comparar al protagonista de este Proyecto con sus competidores directos.

Cada producto tienen unos puntos fuertes y destaca en algunos aspectos concretos, si bien se considera que la mayoría proporcionan una funcionalidad demasiado escasa. **Bee-Bot** dispone de pocas opciones para ser usado a medio o largo plazo, ya que **su uso se limita a programar sus pasos y hacerlo moverse en una dirección un otra.** Bien es cierto que lo poco que hace lo hace muy bien, pero en este caso resulta **imprescindible contar con un educador que explote al máximo esta característica.** El precio, que ronda los 80 €, tampoco lo convierte en un reclamo demasiado atractivo.

La situación en lo referente a **Emiglio** es similar, pudiendo este robot incluso ser considerado como clasista. A pesar de que siempre se pueden aprovechar los juguetes para aprender, los **120 € de precio oficial y sus escasas posibilidades** hacen difícil que salga bien parado si se hace un estudio de mercado relativamente conciso.

Sin embargo, los dos productos restantes muestran un comportamiento y rendimiento por encima de la media, ofreciendo contenido y experiencias realmente útiles para los niños.

En primer lugar, **Codi-Oruga permite un aprovechamiento mayor gracias a su modularidad y los cambios que esto conlleva en su recorrido.** Las luces y los sonidos son lo de menos, pero combinar las secciones de su cuerpo puede ofrecer actividades muy interesantes si se saben aplicar adecuadamente. Su precio ronda los 60 €, el más barato del elenco.

En segundo y último lugar, **el Pleo V2 Reborn supone un salto cualitativo demasiado acentuado como para que la comparativa sea justa.** Su precio, de unos 450 €, no hace sino remarcar la importancia de esta afirmación.

Más que un robot similar al resto de los aquí tratados, **constituye un sistema de inteligencia artificial capaz de aprender a partir de las acciones de los usuarios.** Reacciona a estímulos externos como las caricias, la alimentación por medio de hojas de plástico, la temperatura o emociones como el aburrimiento. Estas capacidades amplían enormemente el abanico de aptitudes que pueden ser tratadas en las aulas, y sin duda **se sitúa como la alternativa más completa en términos generales.** No obstante, su elevado coste hace que sólo sea barajable en casos concretos, siendo prohibitivo para el gran público.

Zowi, por su parte, se abre un hueco y se posiciona en un punto destacable, ya que **se considera el robot más equilibrado tanto en funciones, como en relación**

calidad-precio.

Se puede encontrar entre los 80 y los 90 €, y dado su valor, ofrece una cantidad más que aceptable de sensores y actuadores. Este hecho abre un amplio abanico de posibilidades, siendo posible aprovecharlos para crear actividades lúdicas entretenidas y motivadoras, como se verá a lo largo de este escrito.

A pesar de eso, lo que sin duda constituye **su mayor ventaja es su placa base Arduino y lo que esto implica:** la posibilidad de añadir piezas nuevas y de personalizar tanto el software como el hardware, haciendo que su potencial dependa de la imaginación de los desarrolladores y los maestros.

Software de programación visual

En lo que respecta al software **ScratchJr**, si bien el concepto es similar al de Bitbloq, sólo tienen como denominador común que suponen una abstracción de las líneas de código que se programan por detrás.

El primero está enfocado en la creación de historias interactivas, y en ningún momento se hace hincapié en las estructuras clásicas como condicionales o bucles. El segundo, por su parte, y aunque facilita la tarea de escritura de código, **requiere de cierta comprensión a la hora de colocar los bloques y decidir qué se pretende conseguir.**

Aunque ambos representan lo mejor de los frameworks de programación para niños, la alternativa de BQ se sitúa como una opción más avanzada, pensada para usuarios de más edad y con más conocimiento técnico.

2.3.8. Tabla comparativa de robots

	Resumen	Precio (€)
Bee-Bot	Muy pocas funciones bien realizadas	80
Codi-Oruga	Barato y modular	60
Emiglio	Caro y con pocas funciones útiles	120
Pleo V2 Reborn	Muy caro, complejo y con mucha calidad	450
Zowi	Asequible y con numerosas funciones	90

Tabla 2.1: Tabla comparativa de las alternativas robóticas analizadas.

Capítulo 3

Análisis técnico de Zowi y la aplicación Android

3.1. Zowi, el robot open source

Aunque por fuera tenga una apariencia amigable y bonita, el interior de Zowi es tan complejo como cualquier otro dispositivo tecnológico. **Un conjunto sincronizado de piezas y sensores le permite llevar a cabo sus funciones básicas** y, a su vez, su estudio permite ahondar en su verdadero potencial.

De este modo, saber qué puede hacer un servo, un sensor de ultrasonidos o una matriz de LEDs puede desembocar -como ha sido- en la creación de diversas actividades que aprovechen su funcionamiento.

A continuación se hace un breve repaso sobre estos conceptos, aclarando cuál es su utilidad exacta con el software de fábrica proporcionado por BQ.

3.1.1. Robot de código abierto

Este término, muy conocido por su anglicismo *open source*, implica que **cualquier persona puede descargar el código de Zowi, modificarlo y distribuirlo gratuitamente**, sin la necesidad de pagar derechos de autor.

Aquí radica la que es, posiblemente, su mayor ventaja y lo que ofrece mayor potencial en el futuro. El equipo de desarrollo de la firma española puede crear un código o tener unas ideas realmente buenas, pero **el hecho de que una comunidad entera de makers dedique su tiempo a investigar y contribuir es lo que realmente hace que Zowi pueda mejorar hasta límites increíbles**.

De hecho, este concepto se ha aprovechado en este TFM, ya que se ha podido acceder a todos los archivos fuente, ficheros con información 3D sobre las carcasas e incluso adquirir piezas compatibles.

Son muchos los usuarios con conocimientos técnicos que podrían hacer lo mismo, por lo que **es probable que la calidad de Zowi siga mejorando con el tiempo y el esfuerzo de particulares**.

3.1.2. Placa base

No es la primera vez en este escrito que se destaca **la importancia de Arduino en el resultado final**, ya que constituye el eje central del robot protagonista: su cerebro.

La placa base del robot hace de intermediaria entre todos los componentes, permitiendo cargar tanto el programa por defecto como el código personalizado. Cualquier documentación, sensor o proyecto que se encuentre para Arduino -y los hay a miles en Internet- se puede utilizar en este caso, lo que implica que hay una cantidad ingente de información aprovechable. Concretamente, se trata de **una versión personalizada de la archiconocida plataforma**, desarrollada por la propia firma española.

Existen, no obstante, diferencias importantes que pueden complicar ligeramente la personalización por parte del usuario.

- **Entradas/salidas macho:** si bien **un Arduino convencional dispone de entradas y salidas hembra**, ideales para conectar directamente un jumper, no es el caso de esta placa base. Por lo tanto, hacer nuevas incorporaciones al elenco inicial requiere algo más trabajo, bien buscando los componentes ideales -algo más complicados de encontrar-, o con soluciones caseras usando conectores Dupont.



Figura 3.1: Arduino UNO con conectores hembra.

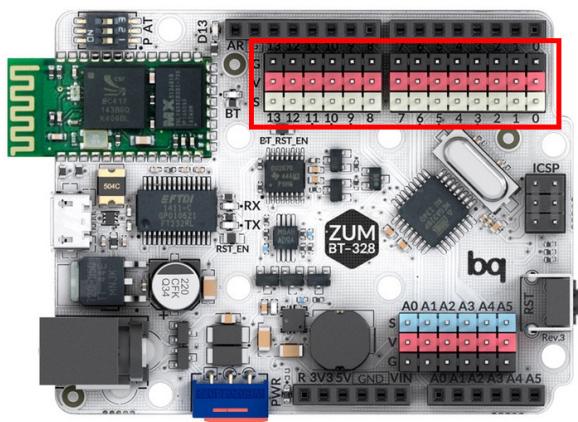


Figura 3.2: Placa similar a la de Zowi con conectores macho.

- **Conectores distintos:** un Arduino Uno -la versión básica más extendida de esta plataforma- dispone de un USB tipo B y clavija de alimentación, mientras que **la**



Figura 3.3: Conector Dupont, cuyo uso es necesario para la conexión de jumpers a la placa de Zowi.

placa de BQ cuenta con un micro USB y clavija para la batería. Salta a la vista que está más enfocado a la portabilidad que su análoga.

- **Botones:** Zowi incluye dos botones de acción, A y B, y uno de encendido.



Figura 3.4: Botones disponibles en la espalda de Zowi.

- **Más funcionalidades:** esta placa amplía el pack básico proporcionado por Arduino Uno con un zumbador, conexión bluetooth integrada, un micrófono...

En general, y a pesar de proporcionar más funciones por defecto que un Arduino básico, **esta placa es algo más compleja de personalizar, ya que el uso de los kits de la primera plataforma no es inmediato.** A pesar de ello, en Internet se pueden encontrar páginas como [Elecfreaks](#), con piezas perfectas para ser conectadas de forma inmediata.

3.1.3. Sensores

Si la placa base es el cerebro, los sensores son -valga la redundancia- los sentidos. Permiten que Zowi esté al tanto de lo que ocurre a su alrededor, permitiéndole interactuar con su entorno y reaccionar a sus estímulos.

- **Sensores ultrasonidos:** los ojos de Zowi, que le permiten detectar obstáculos.

- **Micrófono:** no posibilita el reconocimiento de voz, pero **se usa para detectar sonidos o golpes en la carcasa.** De este modo, se pueden programar acciones como que el robot reaccione ante una palmada, o empiece a caminar cuando sea tocado.

3.1.4. Actuadores

Las manos y los pies son a los actuadores lo que los sentidos a los sensores. Estos elementos son **los encargados de realizar acciones -no de detectarlas-**, como caminar o emitir sonidos. Mientras los sensores permiten recibir estímulos, los actuadores posibilitan reaccionar a ellos.

- **Servos:** los servos son **motores eléctricos que giran entre un ángulo máximo y uno mínimo**, permitiendo controlar por software en qué cifra exacta se ubican. A diferencia de los motores convencionales, estos no giran de forma continua a una determinada velocidad angular, sino que su posición es siempre controlable desde la placa base.

La cadera de Zowi está formada por dos servos, uno a cada lado. La articulación de los pies, por su parte, sigue la misma estructura. El movimiento sincronizado de los cuatro motores -que se estudiará [más adelante](#) en profundidad- provoca que el robot camine.

- **Matriz de luces LED:** conjunto de luces LED de 5 filas por 6 columnas que **forma la boca del robot.** Cada uno de los 30 puntos se puede iluminar de forma independiente, siendo posible representar números, letras o cualquier forma a imaginación del usuario -siempre que la escasa resolución lo permita-.
- **Zumbador:** elemento que permite **emitir sonidos.**

3.1.5. Pines digitales y analógicos

Como todo Arduino, las capacidades del protagonista incluyen pines analógicos y digitales, **a los que van conectados todos los componentes del conjunto.** Pueden ser configurados -con algunas excepciones- como entradas (para recibir datos de sensores) o salidas (para enviar instrucciones a actuadores).

- **Pines digitales:** su rango de trabajo se limita a **dos valores, 0 ó 1 -LOW o HIGH-**. De este modo, funcionen como entrada o como salida, sólo va a haber dos opciones posibles.
- **Pines analógicos:** al funcionar como entradas, estos pines **leen valores de tensión entre 0 y 5 V.** Cuentan con una resolución de 10 bits, lo que significa que pueden diferenciar un total de 1024 voltajes distintos ($2^{10} = 1024$). De este modo, la variación de tensión necesaria para que haya un salto en la cifra leída es de unos 5 mV ($5V/1024 = 5mV$).

Por su parte, el funcionamiento como salidas, también conocido como PWM, presenta alguna particularidad. Este modo produce intervalos a 0 intercalados con intervalos

los a 1, cuya duración depende del valor que le proporcionemos al pin -comprendido entre 0 y 255-.

Están distribuidos de la siguiente manera:

- **D0 y D1:** no disponibles.
- **D2, D3, D4 y D5:** ocupados por los 4 servos que mueven las piernas.
- **D6 y D7:** correspondientes a los botones A y B, respectivamente.
- **D8 y D9:** ocupados por el sensor de ultrasonidos.
- **D10:** correspondiente al zumbador.
- **A0, A1, A2, A3, A4 y A5:** libres.
- **A6:** correspondiente al micrófono.

Lo realmente destacable en toda la distribución es **la falta de conexión en los 6 primeros pines analógicos**. En ellos podremos conectar cualquier sensor o actuador a la placa base para, a continuación, cargar un programa personalizado en la misma y hacer uso de ellos.

Y es en este hecho en el que reside parte de la grandeza del producto. La experiencia pudiendo añadir componentes nuevos a cambio de quitar los originales le restaría interés al conjunto; pero afortunadamente, es perfectamente posible ampliar en lugar de sustituir, como se describirá más adelante.

3.1.6. Carcasas 3D

El mundo de la impresión 3D no ha parado de crecer en los últimos años, y productos como este no hacen sino reafirmar el hecho de que está aquí para quedarse.

Todas las piezas de plástico de Zowi están impresas de esta forma, desde la cabeza hasta los pies. Puede que estos últimos no tengan demasiada importancia, pero **la cara del robot es lo primero que llama la atención**, y ante lo que se puede sentir más empatía. Por lo tanto, su personalización puede resultar conveniente en función de los usos que se le vayan a dar, como describe BQ en su blog: diwo.bq.com/zowi-personalizar-impresion-3d.



Figura 3.5: Posibles personalizaciones de la cabeza de Zowi.

El aspecto físico, no obstante, no lo es todo. **Y es que el interior de la pieza frontal está pensado para albergar los componentes de fábrica, pero no para incluir otros nuevos.** ¿Qué ocurre si se quiere añadir un diodo LED para dotarlo de luz en los ojos? ¿O si se le quiere proporcionar la capacidad de detectar la luminosidad en el ambiente?

Siempre se puede recurrir a soluciones caseras, como el celo, la cinta de doble cara o un taladro. No obstante, no son recursos ideales si lo que se busca es un resultado profesional. Para esto, y siempre sujetos a la disponibilidad de una impresora 3D, **se puede modificar la plantilla básica para añadir huecos, agujeros, repisas o soportes** en el interior del protagonista.

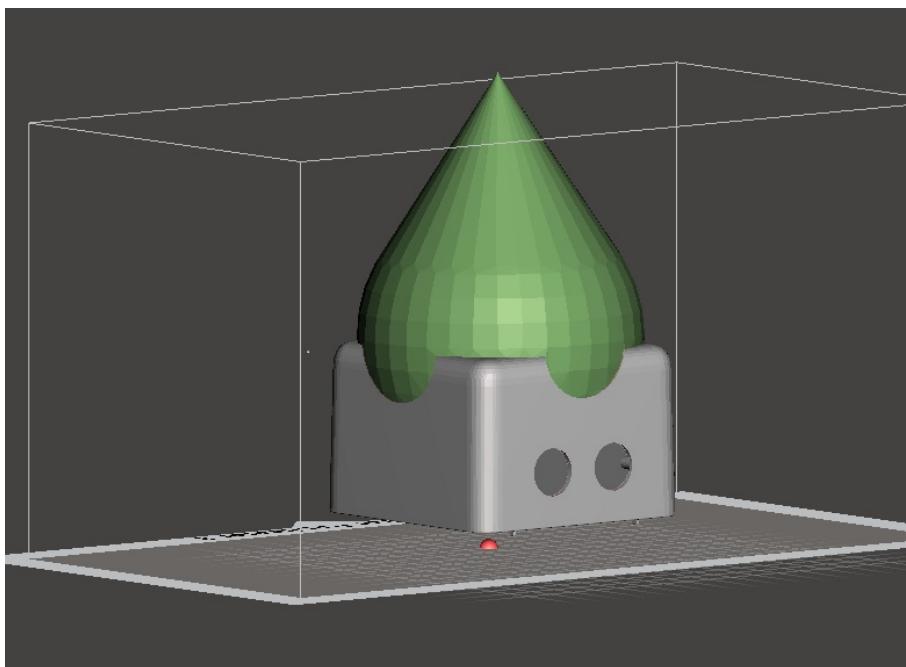


Figura 3.6: Diseño de un gorro con forma de gota para Zowi.

3.1.7. Lenguaje de programación Arduino y Bitblock

A la hora de programar la placa -o Arduino- de Zowi, hay dos importantes vertientes que deben ser tenidas en cuenta.

- **Bitbloq:** es un **mecanismo de programación por bloques** que permite crear programas de forma muy visual, arrastrando -valga la redundancia- bloques para colocarlos en el lugar que se considere adecuado.

No deja de ser una simplificación o una especie de **framework que opera sobre el propio código**; es decir, aunque se puedan conseguir sentencias, condicionales o bucles de modo muy sencillo, por detrás realmente se están asignando valores a variables, definiendo *ifs* o iterando en *whiles*.

Aunque dedicándole tiempo este sistema tiene mucha potencia, **está más orientado a que los niños se inicien en el mundo de la programación** y la informática, más que a crear programas completos desde cero. Para eso, se cuenta con la siguiente alternativa.

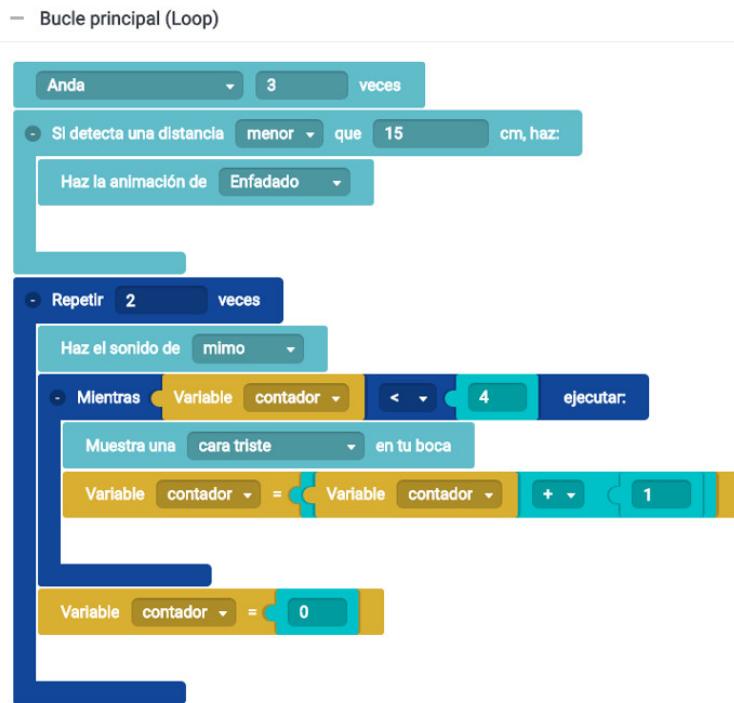


Figura 3.7: Código de ejemplo “programado” con Bitbloq -con su equivalente en Arduino más adelante-.

- **Lenguaje Arduino:** si se quiere tener un control completo del código y sus consecuencias, se puede tratar a Zowi como el Arduino que es y **programarlo directamente como se haría en caso de contar únicamente con la placa**. Para ello, se utiliza un lenguaje basado en C++, que permite el manejo de aspectos como la gestión de la memoria RAM del dispositivo.

```

1 void loop() {
2     zowi.walk(3, 1000);
3     if (zowi.getDistance() < 15)
4         zowi.playGesture(angry);
5
6     for (int i=0; i<2; i++) {
7         zowi.playSound(caress);
8         while (contador < 4) {
9             zowi.putMouth(sad);
10            contador = contador + 1;
11        }
12        contador = 0
13    }
14 }
```

Todos los archivos fuente, incluidas las librerías utilizadas, se pueden descargar del GitHub de BQ, github.com/bq/zowi/tree/master/Zowi_mold/scr. Constituyen un punto de partida excelente para estudiar los algoritmos utilizados y las posibles mejoras.

3.2. Aplicación Android

A pesar de que Zowi puede trabajar de forma autónoma, interactuando con los botones de los que dispone, **sólo se expresa todo su potencial si se controla en tiempo real en función de los deseos del usuario.**

El más claro ejemplo es la aplicación que BQ pone a disposición del usuario en el Google Play Store, de la que se ha hablado en [Justificación del Proyecto: ¿está Zowi a la altura?](#). Esta se conecta con el robot a través de Bluetooth, momento en el que se pone en marcha un modo especial que recibe comandos enviados desde el móvil o la tablet.

De esta forma, se le puede hacer caminar, bailar, encadenar movimientos, jugar a pintar las bocas mostradas en la matriz de LEDs, y algunas otras **tareas que pueden resultar muy útiles para comprobar cómo afecta la causalidad a las decisiones tomadas en la app.**

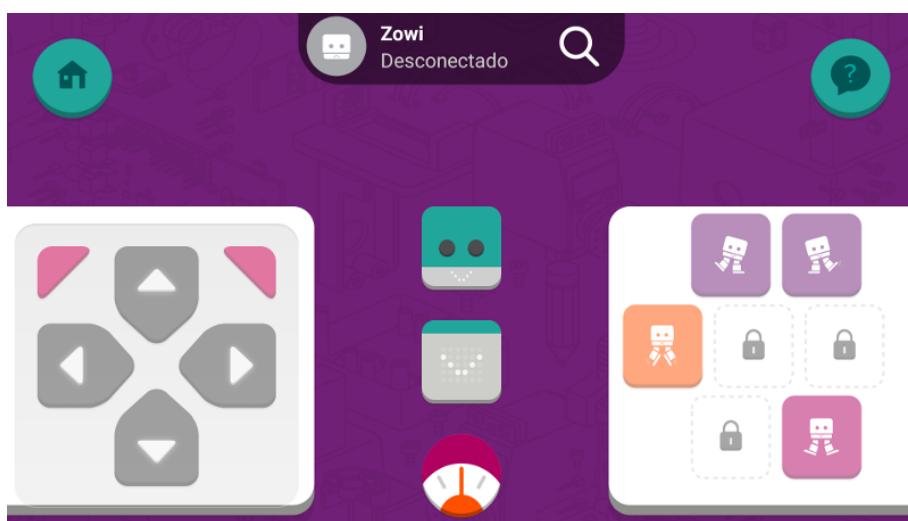


Figura 3.8: Sección Gamepad de ZowiApp.

La idea aplicada en este TFM es análoga, con dos salvedades importantes que deben ser especialmente destacadas: por un lado, **la aplicación será creada de cero**, con una interfaz rediseñada y adaptada al nuevo contenido del que dispondrá. Y por otro, **la finalidad será completamente distinta**; los niños no sólo van a jugar, sino que principalmente van a aprender.

3.2.1. Lenguaje de programación Java

Como en la mayoría de aplicaciones Android, el **lenguaje de programación utilizado será Java**, enormemente conocido a nivel mundial.

Existirán, no obstante, algunas limitaciones debido a la versión de sistema operativo. Al ser necesaria la compatibilidad con Android 4.4, algunas funcionalidades incorporadas en Java 8 no podrán ser aprovechadas.

3.3. Ampliaciones sobre el hardware

Un buen algoritmo de programación puede conseguir resultados muy útiles y eficientes, pero necesita un hardware sobre el que correr y del que poder hacer uso. Por eso, **hay**

infinidad de acciones que no se pueden llevar a cabo con las piezas de las que Zowi consta por defecto, dependiendo del usuario el ampliarlas o limitarse a la funcionalidad ofrecida de fábrica.

La primera alternativa es la considerada más adecuada en este Proyecto. Aunque una forma ingeniosa de aplicar lo que se tiene es considerado más importante, **por poco dinero se pueden encontrar sensores que mejorar enormemente las posibilidades del robot**. Por lo tanto, se han valorado las siguientes opciones:

TEMA DE SENSORES

- **MPU 6050 ó 9250:** estos sensores son IMUs, acrónimo de *Inertial Measurement Units*. Consisten en **dispositivos electrónicos que proporcionan medidas sobre la aceleración y la orientación de los mismos**, siendo muy utilizado en aviación.

El primero cuenta con 6DOF (6 *Degrees Of Freedom*, lo que significa que la recopilación de datos se hace mediante un acelerómetro y un giroscopio de 3 ejes cada uno (3DOF x 2). Se estudiará si esta combinación es viable para conseguir que Zowi mejore su [orientación y su forma de caminar](#), o por el contrario es necesario optar por la segunda alternativa.

Esta constituye una mejora con respecto a la anterior, al contar este IMU con 9DOF: un acelerómetro y un giroscopio con la misma configuración, y un magnetómetro (o brújula) para complementar la información recopilada.

¿Qué se pretende conseguir con esto? Zowi, a medida que da pasos, se va ladeando poco a poco. Con uno estos sensores se podría **determinar una dirección inicial que se tome como referencia, para posteriormente ir compensando el giro** a partir de las medidas extraídas del IMU.

- **Luces LED:** los colores son muy llamativos y expresivos, más que las luces blancas de la boca de LEDs y, como no podría ser de otra forma, más visuales que los sonidos.

De cara a reforzar la validez o incorrección de una respuesta en las actividades, **se pretende dotar al robot de luz en las “orejas”**, iluminándose en función de lo estipulado en el código de programación.

El no disponer de las comentadas orejas por defecto no supone un problema, dado que se puede hacer uso de la impresión 3D para llevar a cabo cualquier tipo de nueva forma u orificio.

- **Sensor emisor-receptor de infrarrojos:** con el objetivo de complementar la forma de caminar, se estudiará la inclusión de uno o dos sensores de infrarrojos. **El concepto ese el mismo que en los robots siguelíneas**, pudiendo detectar la variación de colores en el suelo y actuar consecuentemente.

- **Nuevas funciones sobre la base:** el hecho de añadir sensores y actuadores no implica que con lo ofrecido de fábrica no se pueda experimentar.

Haciendo uso del código por defecto e implementando nuevas líneas, **se dará un giro de tuerca en lo referente a las acciones de Zowi**. Por ejemplo, se implementará un *banner* en la boca, de forma que se puedan mostrar de derecha a izquierda

operaciones matemáticas en una de las actividades, así como la funcionalidad de un *sonar*.

Capítulo 4

Planificación del Proyecto y Resumen de Presupuestos

4.1. Planificación

Se ha llevado a cabo una **planificación inicial del proyecto**, teniendo en cuenta todos los factores que pueden influir en él para realizar una estimación de tareas, hitos y plazos.

Con ese objetivo, las **ideas evaluadas se han plasmado en un diagrama de Gantt**. Esta metodología de gestión de proyectos se ha escogido dada su simple visualización, unida a la posibilidad de mostrar gran cantidad de información en un espacio reducido. La vista de esquema permite tener una noción clara sobre los tiempos manejados, así como sobre la división en tareas y los hitos que tendrán lugar en el desarrollo.

Se han distinguido varias etapas distintas, desde la planificación hasta las pruebas posteriores al desarrollo. Las distintas tareas y su correspondiente temporalización se pueden ver en las Figuras 4.1, 4.2, 4.3 y 4.4.

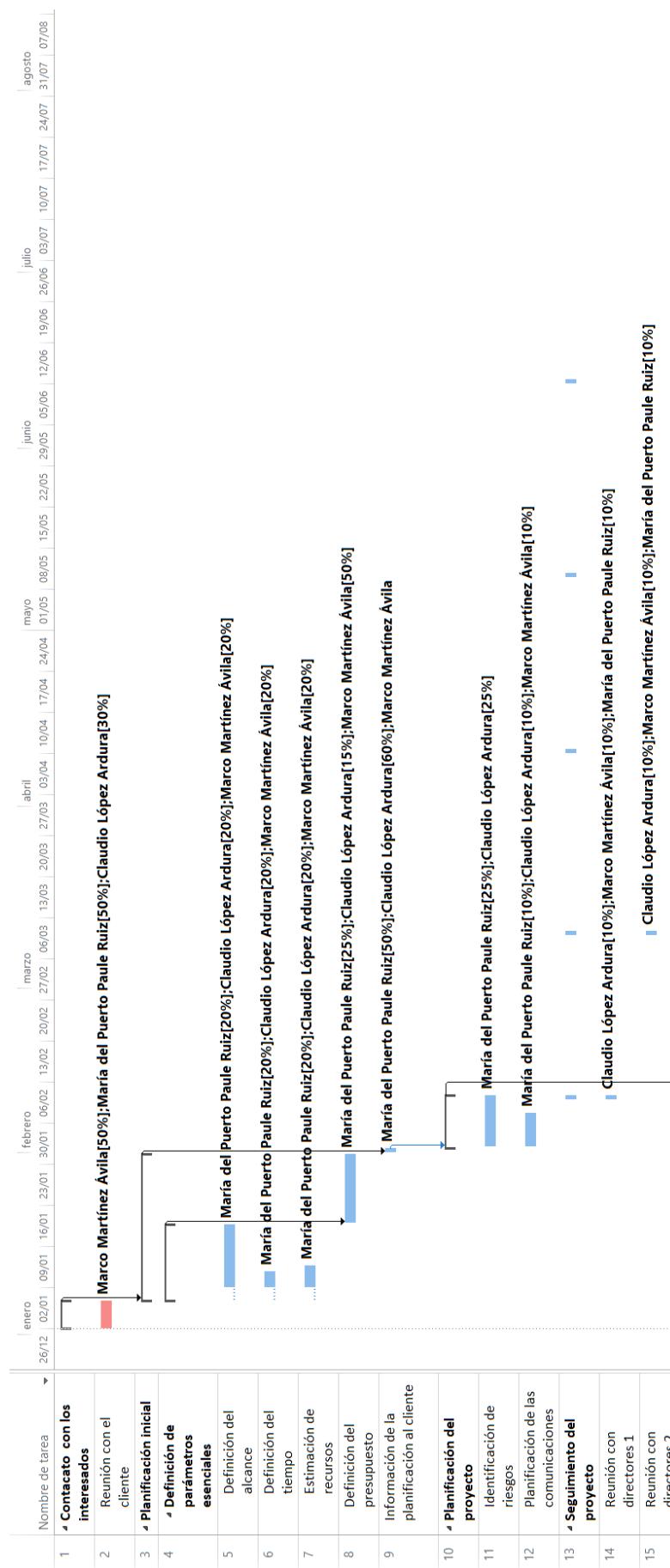


Figura 4.1: Primera parte del diagrama de Gantt

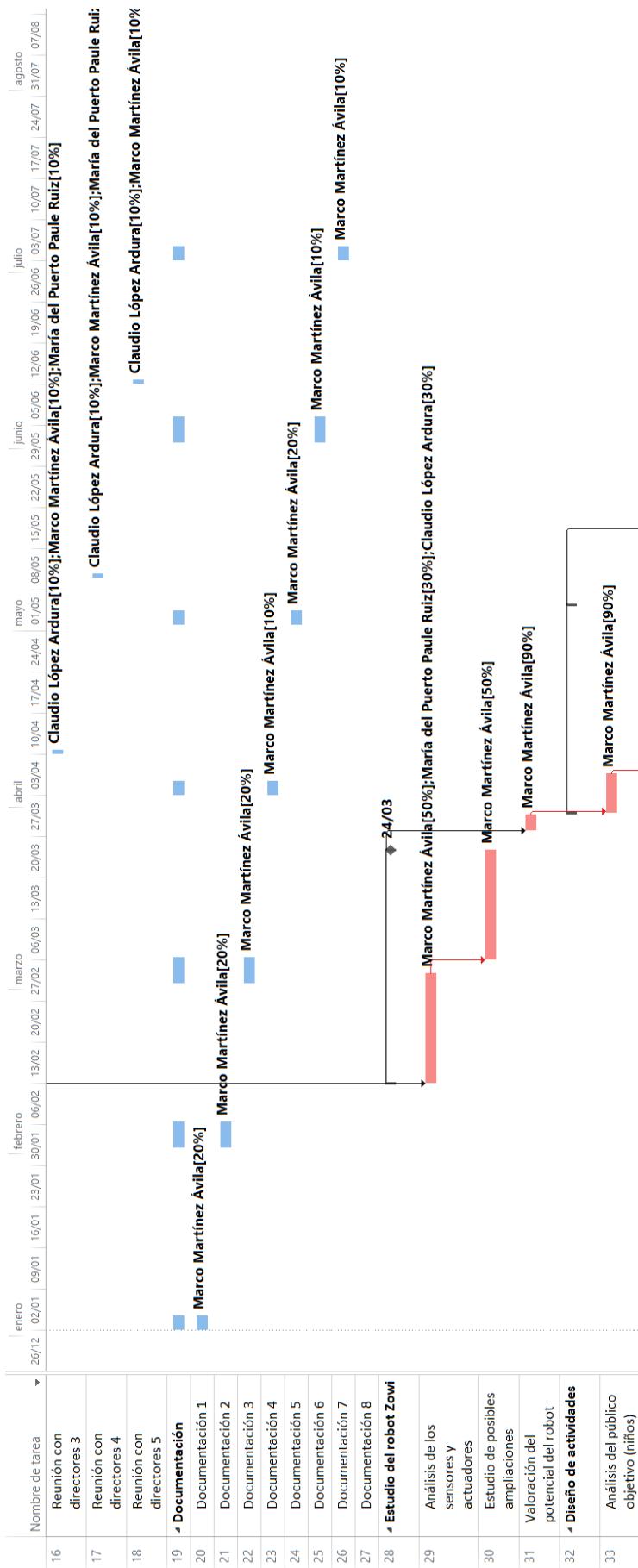


Figura 4.2: Segunda parte del diagrama de Gantt

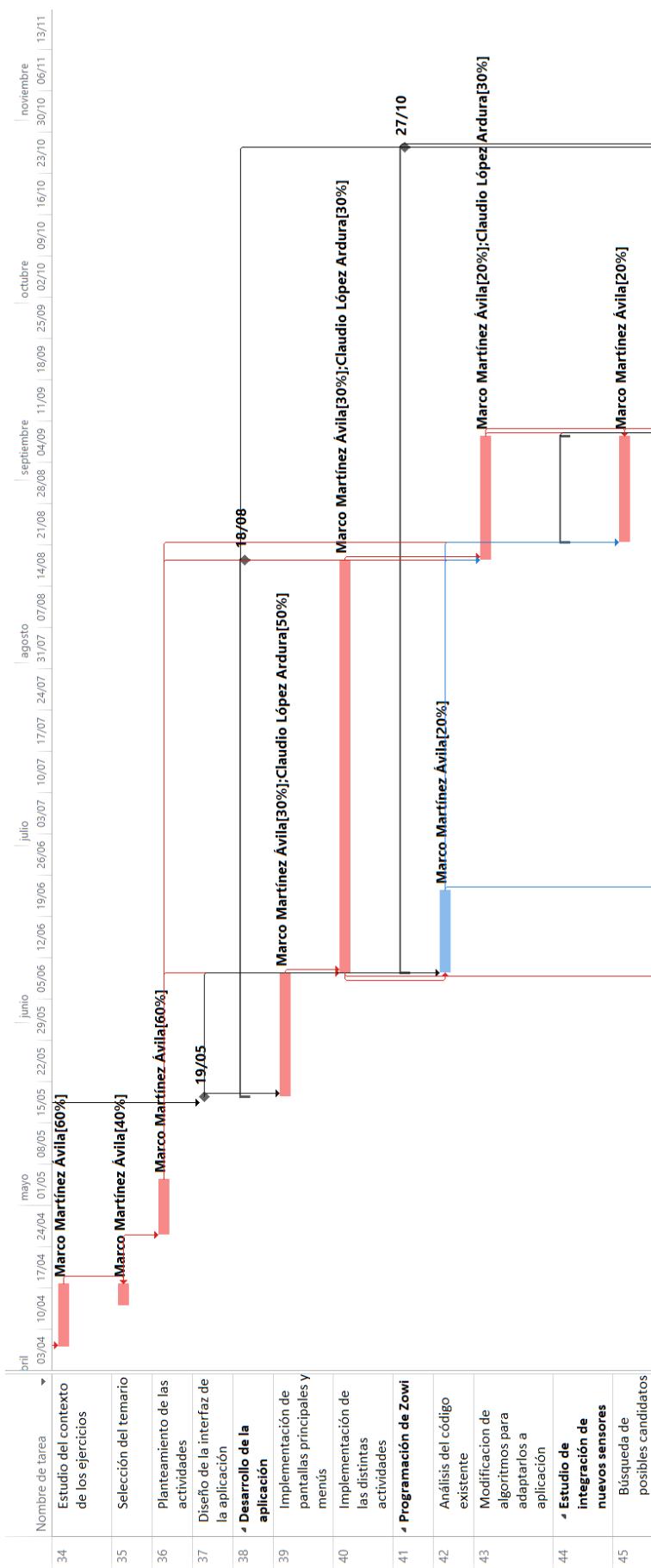


Figura 4.3: Tercera parte del diagrama de Gantt

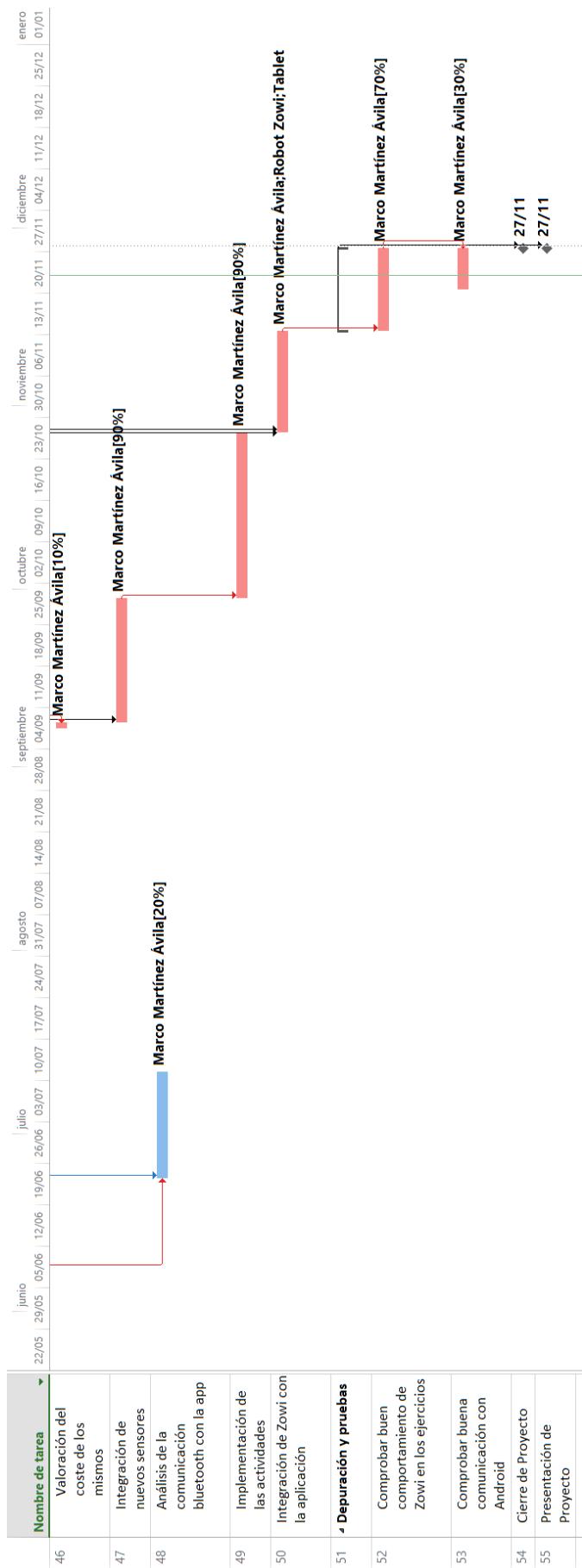


Figura 4.4: Cuarta parte del diagrama de Gantt

4.1.1. Procesos previos al desarrollo

Existen **multitud de procesos que es aconsejable llevar a cabo antes de comenzar con los planteamientos técnicos y el desarrollo**. Para que la propuesta sea realista, y con el fin de que la definición del TFM sea de calidad, se han tenido muy en cuenta los fundamentos para la dirección de proyectos presentes en el PMBOK.

Esta guía, basada en gran medida en la experiencia de especialistas, reúne **una gran cantidad de descripciones y un conjunto de buenas prácticas en la Gestión de Proyectos**. Teniendo en cuenta que, en el caso de este escrito, sólo una persona contará con la responsabilidad de la consecución de todas las tareas, se ha hecho un análisis para **escoger sólo aquellos procesos considerados imprescindibles**.

Aunque la puesta a punto de muchas subtareas en un Proyecto real llevarían semanas o incluso meses, la situación actual dista bastante de este hecho. No obstante, y a pesar de la aparente ausencia de esfuerzo en algunos puntos del diagrama, se ha decidido no borrarlos con el objetivo de **mantener una estructuración correcta y adecuada**.

4.1.2. Estudios previos

Una vez planteada la planificación y la teoría inicial, tendrá lugar un **estudio sobre la viabilidad de las propuestas y requisitos del cliente**.

Por ejemplo, es posible que Zowi no cumpla con las exigencias del Proyecto, o que presente alguna limitación de hardware que haga imposible su uso. Por motivos análogos, se analizarán los recursos necesarios -y potencialmente necesarios- con el fin de determinar si el planteamiento tiene opciones de llegar a buen puerto.

4.1.3. Desarrollo

Este apartado se divide en dos subsistemas bien diferenciados, que requieren el uno del otro para poder avanzar en algunos de los casos.

Por un lado, **Zowi, con su lenguaje de programación y sus componentes**, constituye un elemento indispensable del conjunto. Si bien hay tareas que se pueden llevar a cabo de forma independiente, en numerosas situaciones se tendrá que examinar su comunicación con la **aplicación Android**.

Esta última, por otro lado, constituye **la segunda pieza crucial del puzzle**. Los niños interactuarán directamente con ella -más que con el robot-, por lo que una buena experiencia en su uso es imprescindible.

Ambas ramas avanzarán tanto independientemente como en paralelo, en función de la etapa del Proyecto en la que se encuentre el desarrollo. La colaboración no será un problema, debido al trabajo individualista del TFM.

4.1.4. Pruebas

Se llevará a cabo una **batería de pruebas que permitan analizar el correcto funcionamiento del conjunto**, simulando el uso real de la aplicación y determinando qué aspectos son mejorables o no funcionan como deben.

Para ello, **se contará con la colaboración de una maestra en Educación Infantil**, que aportará su punto de vista de usuario especialista en la materia. De este modo,

además de corroborar la exactitud del código, se evaluarán aspectos relacionados con la temática y la contextualización de ZowiApprende en el entorno de las aulas escolares.

4.2. Resumen del presupuesto

Se ha elaborado un presupuesto teniendo en cuenta **la participación del desarrollador, del director y del subdirector del TFM**.

Esta participación, junto con el capital invertido en la obtención de los recursos necesarios -Zowi, sensores, la carcasa 3D y la tablet en la que se ejecutará la aplicación- resultan en la información mostrada en la Tabla 4.1.

Presupuesto ZowiApprende				
Nombre	Marco Martínez Ávila			
Dirección	Avenida de Noreña, 2 - 3º			
Población	Pola de Siero			
Provincia	Asturias			
NIF	76964701P			
Teléfono	696619477			
Correo electrónico	marcomartinezavila@gmail.com			
Fecha	31/01/2017			
Concepto	Cantidad	Precio (€)	Duración (h)	Total (€)
Ingeniero Web	1	25,00	928,0	23.200,00
Subtotal personal de desarrollo				23.200,00
Subdirector de Proyecto	1	30,00	16,0	480,00
Director de Proyecto	1	40,00	52,8	2.112,00
Subtotal personal de dirección				2.592,00
Robot Zowi	1	89,90	-	89,90
CNY70	5	7,00	-	35,00
MPU6050	2	9,00	-	18,00
MPU9250	1	12,00	-	12,00
IMU Octopus	1	23,00	-	23,00
LEDs	4	4,00	-	20,00
Sensores infrarrojos	2	3,00	-	6,00
Tablet	1	250,00	-	250,00
Caracasa 3D	1	70,00	-	70,00
Subtotal hardware				523,90
Android Studio	1	0,00	-	0,00
PlatformIO IDE	1	0,00	-	0,00
Código Zowi	1	0,00	-	0,00
Subtotal				26.315,90
Beneficio (25 %)				6.578,98
IVA (21 %)				5.505,55
TOTAL				38.400,43

Tabla 4.1: Presupuesto del ZowiApprende.

Capítulo 5

Estudio y análisis del software de fábrica

5.1. Zowi y su comportamiento por defecto, la base de todo su potencial

Para explicar y comprender el desarrollo llevado a cabo, **es necesario partir del código y las funciones por defecto**, pues no sólo se han tomado como referencia sino que también se ha pretendido mejorarlas.

No se va a hacer un repaso de todas las características y líneas de código existentes, pues no se pretende que este escrito sea un tutorial y, en muchos casos, el funcionamiento sigue el mismo patrón: se asigna un sensor o actuador al pin en el que esté conectado, y se lee el valor o se ejecuta la acción por medio de una instrucción específica con algún parámetro.

Este flujo se repite, por ejemplo, con el zumbador, el micrófono o la distancia medida por los ultrasonidos, haciendo tremadamente sencillo su uso. En estos casos, **más importante que la complejidad del código son las ideas en las que se pueda aplicar, su originalidad y su utilidad en las aulas**.

No obstante, el código fuente no está exento de funciones más complejas, dos de las cuales merece la pena explicar en este punto.

5.1.1. Matriz de LEDs

Fundamento teórico

Estos 30 LEDs, organizados en 5 filas y 6 columnas, se consideran la base de una de las ampliaciones más interesantes y explicada más adelante.

Entre las posibilidades de Zowi está el interactuar con el entorno por medio de **diferentes bocas**, como sonrisas, caritas tristes o colmillos de vampiro, **además de números del 1 al 9**. Los ingenieros de BQ también han incorporado las animaciones, que consisten en una lista de caritas que Zowi mostrará una detrás de otra. Se pueden conseguir así efectos bastante interesantes, como el del robot mostrando sorpresa.

Pero como hay que comenzar con el principio, en primer lugar es necesario destacar **el pintado de una sonrisa individual**. El concepto es realmente sencillo, y se reduce a **ubicar un 0 en aquellos puntos que deben estar apagados, siendo 1 en caso**

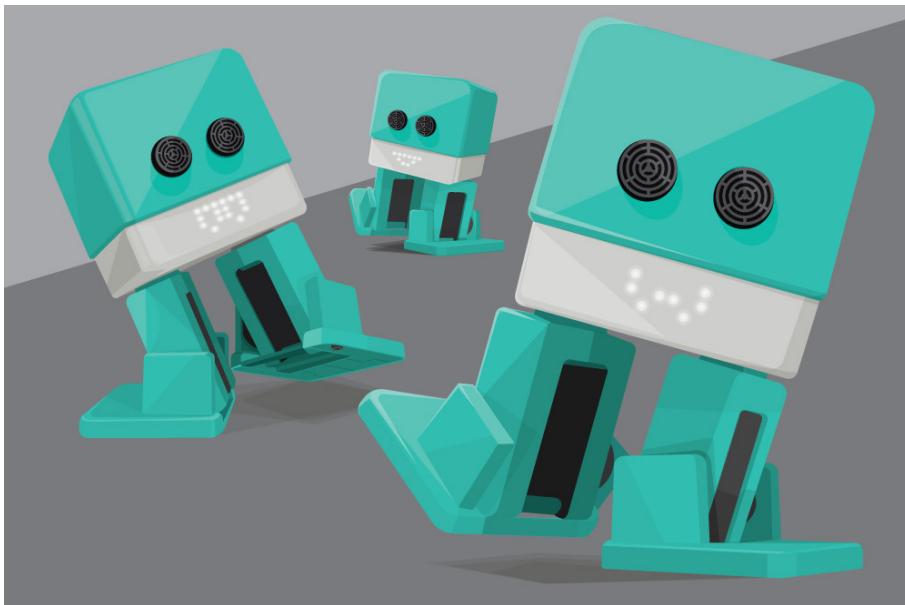


Figura 5.1: Formas mostradas en la matriz de LEDs de Zowi.

contrario. Sin embargo, la explicación a nivel de código, si bien no es compleja, requiere de un procesamiento ligeramente superior.

Para comprenderlo, se debe hacer referencia a los tipos de [variables de Arduino](#). Cada uno de ellos ocupa un determinado espacio en memoria, como se explica en el propio [blog de BQ](#):

- **char:** se utilizan para almacenar caracteres, con un byte de memoria ocupado.
- **byte:** permite almacenar un número entre 0 y 255, y su propio nombre indica cuánto ocupa. Equivale a 8 bits binarios -00000000-.
- **int:** la representación más extendida de números enteros, ocupa 2 bytes en memoria -16 bits-.
- **long:** permite un rango más amplio en la representación de números enteros, al ocupar 4 bytes -32 bits-.
- **float:** números decimales con un espacio ocupado de 4 bytes -32 bits-.
- **double:** tipo con el rango más amplio para la representación de números decimales, y número 1 en el podio con 8 bytes -64 bits-.

La información expuesta puede relacionarse de forma inmediata con lo comentado anteriormente, y es que **la representación de 1 (luz encendida) y 0 (luz apagada) va a tener lugar gracias al tipo long**.

Se puede ver que este ocupa **32 bits, 2 más que el número de LEDs del que dispone la boca de Zowi**. Por lo tanto, una sola variable de este tipo puede albergar toda la información necesaria para pintar una sonrisa, dientes de vampiro o incluso formas geométricas.

Formación de la boca con 1 y 0

De los 32 bits del tipo long que recibe la función que ejecuta esta tarea, sólo 30 son necesarios. Pero, ¿cómo se convierte una serie de bits en una matriz de 1 y 0?

Obviando los 2 primeros bits -cuyo valor es indiferente-, los siguientes se agrupan, de mayor a menor peso, en **conjuntos de 6 -uno por cada columna-**. Así, 5 repeticiones -filas- de 6 bits resultan en el número total deseado.

De esta forma, por ejemplo, para pintar una sonrisa se podría utilizar la siguiente configuración.

```
00000000100001010010001100000000
```

Despreciando los dos bits iniciales:

```
000000
100001
010010
001100
000000
```

Animaciones simples por defecto

La consecución de animaciones simples consiste en la repetición de lo explicado en el apartado anterior; es decir, **pintar bocas más de una vez**. Combinando distintos long binarios y añadiendo un retardo entre ellos, se puede lograr este efecto de forma inmediata.

En el código proporcionado por BQ, estos long se almacenan en arrays, escogiendo uno u otro en función del índice obtenido al recorrer un bucle. Este algoritmo, aunque funcional, presenta un inconveniente que puede ser relevante en algunos casos: el amplio consumo de memoria.

Se tienen 4 animaciones, cuyo número total de elementos es de 28. Esas 28 variables tipo long, teniendo en cuenta la memoria que ocupan, resulta en 112 bytes. Y aunque la cifra puede llamar la atención por ridículamente escasa, no se debe olvidar que **el Arduino de Zowi dispone de 2 KB de almacenamiento**. Por lo tanto, optimizar este tipo de procesos puede repercutir muy positivamente en el desempeño del software.

```
1 unsigned long int littleUuh_code[] = {
2     0b00000000000000001100001100000000,
3     0b00000000000000001100001100000000,
4     0b00000000000000001100001100000000,
5     0b00000000000000001100001100000000,
6     0b00000000000000001100001100000000,
7     0b00000000000000001100001100000000,
8     0b00000000000000001100001100000000,
9     0b00000000000000001100001100000000
10    };
```

5.1.2. Mecanismo para mover las piernas y algoritmo para caminar

Sin duda, este apartado alberga las líneas de código más complejas del cerebro de Zowi, más allá de la gestión de memoria y la -en algunos casos- problemática comunicación con los periféricos. Este hecho se debe a la **necesidad de trabajar de forma sincronizada con los 4 servos disponibles**, moviendo los pies y las caderas de forma que se avance en una dirección, hacia delante o hacia atrás.

Fundamento teórico

Como se explicaba en la sección [Actuadores](#), los servos son motores eléctricos cuyo giro es controlado mediante una instrucción de código. En este caso, el modelo de los 4 componentes es el mismo: Futaba S3003, con un rango máximo de 180º y ubicados como se puede ver en la Figura 5.2.

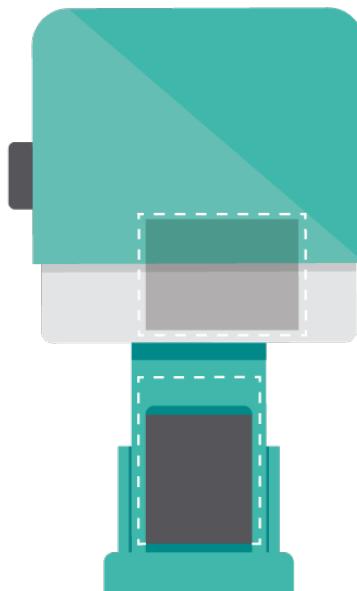


Figura 5.2: Posición de los servos desde el perfil de Zowi.

En la posición de reposo -que se puede ver en la Figura 5.3-, **el valor por defecto de los servos es de 90º**. O dicho de otra forma, cuando mediante código se indique que los servos deben girar hasta los 90º, tras la transición Zowi acabará estando en la posición de reposo.

En la misma imagen también se puede ver **el nombre que reciben los servos a nivel interno**. Su orden es importante de cara a su manipulación, ya que el parámetro principal de la función que los mueve es un array. En él, se debe indicar en ángulo final que se pretende alcanzar, estando cada número en la posición correspondiente.

El orden es el siguiente: YR, YL, RR y RL. Es decir, cadera derecha, cadera izquierda, pie derecho y pie izquierdo. Así, el valor necesario para hacer que Zowi vuelva a la posición de reposo es el siguiente, donde el primer elemento de la lista corresponde a YR y el último a RL.

```
1 int restPosition[4] = {90, 90, 90, 90}
```

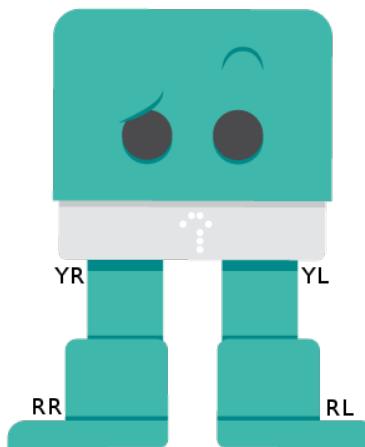


Figura 5.3: Posición de reposo de Zowi.

Partiendo de esta posición y estos ángulos, cabe destacar que los sentidos de giro se adecuan a lo mostrado en las Figuras 5.4 y 5.5.

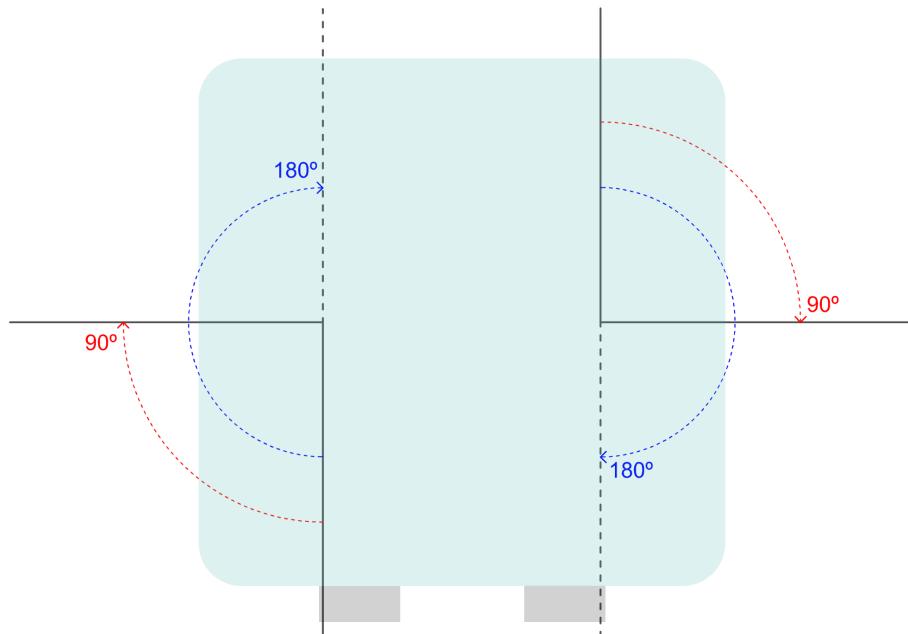


Figura 5.4: Esquema de funcionamiento de los servos de la cadera

De este modo, si se quiere que Zowi levante el pie izquierdo, se deberá proporcionar un valor en el servo RL menor de 90° . Si se quiere replicar este comportamiento con el pie contrario -RR-, el valor necesario deberá estar comprendido entre 90° y 180° .

```
1 int restPosition[4] = {90, 90, 60, 120}
```

El eje longitudinal de YR e YL es perpendicular al plano vertical en lugar de paralelo, por lo que el esquema de la Figura 5.4 se ha adecuado convenientemente. En la misma se puede comprobar que el funcionamiento es análogo al de los pies.

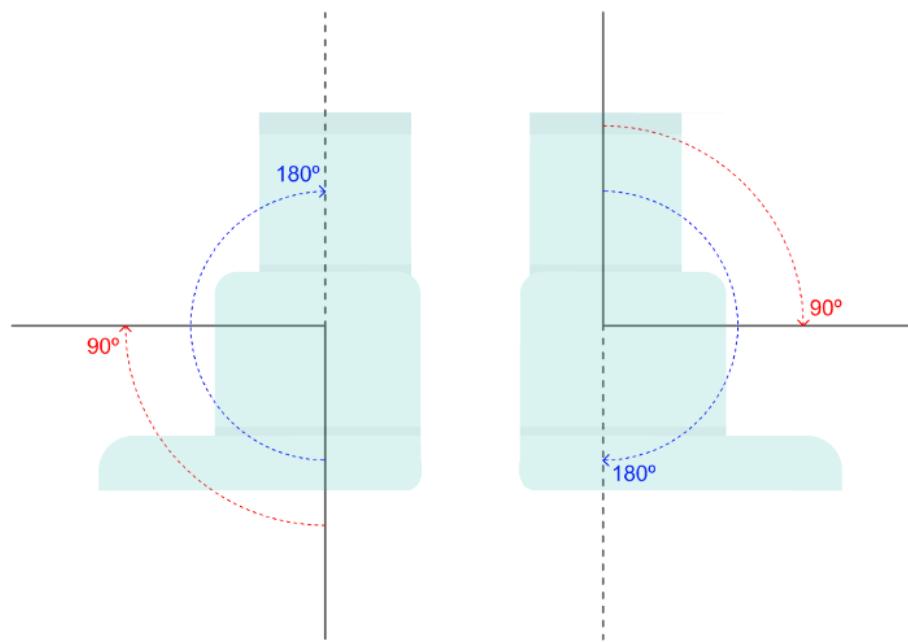


Figura 5.5: Esquema de funcionamiento de los servos de los pies

Bailes y otros movimientos

Combinando las cifras expuestas, se consiguen la mayoría de casos en lo referente al movimiento de Zowi. Normalmente, este consiste en proporcionar el array adecuado al gesto que se pretende conseguir, combinando este con bocas en la matriz de LEDs y la emisión de sonidos.

5.1.3. Algoritmo para caminar

El algoritmo utilizado para dar pasos es la base de la ampliación más importante del TFM, consistiendo esta última en la inclusión de sensores para mejorar su desempeño. Por tanto, es imprescindible comprender el funcionamiento del código por defecto para poder mejorarlo posteriormente. A continuación se muestra la función responsable de llevar a cabo esta tarea.

```
1 void walk(float steps=4, int T=1000, int dir = FORWARD);
```

Los parámetros indican, respectivamente, lo siguiente: **el número de pasos, el período de cada uno de ellos, y el sentido del movimiento** (hacia delante o hacia atrás). No obstante, los pormenores del algoritmo no recaen sobre estos argumentos, sino que lo hacen sobre **variables definidas en el propio método “walk” y en funciones sinusoidales**.

Tanto el seno como el coseno son funciones continuas y periódicas, cuyos valores varían progresivamente entre un máximo y un mínimo. Los límites superior e inferior pueden ser determinados por el programador, así como el período de las ondas y su desfase.

El concepto del algoritmo se basa en esta idea. Mediante la personalización de la amplitud, del desplazamiento sobre el eje Y y del desfase, **se consiguen ondas cuyos**

valores se envían a los servos. De este modo, se obtiene un movimiento perfectamente sincronizado que, si bien cuenta con cierta imprecisión, cumple su cometido con creces en la mayoría de situaciones.

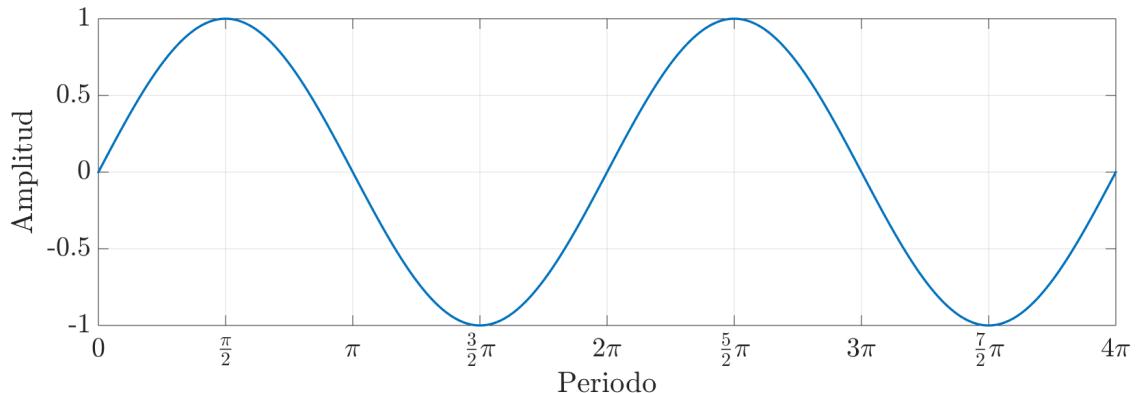


Figura 5.6: Seno con amplitud 1 y período 2π .

Con respecto a las variables cruciales en el correcto desempeño del sistema, se describen a continuación:

- **A:** amplitud máxima de la señal sinusoidal.
- **O:** constante que permite desplazar la función a lo largo del eje Y.
- ϕ : desfase del conjunto, cuya labor es sincronizar el movimiento de las caderas con el de los pies.

En último término, el valor a grabar en cada servo se rige por la siguiente función:

$$\text{round}(A * \sin(x + \phi) + O) \quad (5.1)$$

Siendo posible corroborar que la modificación de las 3 variables mencionadas resulta en un ángulo u otro. Cabe destacar que la función *round* es necesaria para garantizar la escritura de números enteros en los motores, ya que es el único tipo de número que permiten.

En la Figura 5.6 se puede ver la onda base, a partir de la cual se construyen todas las demás, siendo $A = 1$, $\phi = 0$ y $O = 0$.

Concretando la información teórica plasmada un poco inmediatamente más arriba, los valores utilizados a la hora de caminar son siempre los mismos.

- Caderas izquierda y derecha:

$$y = 30 * \sin(x) \quad (5.2)$$

- Pie derecho:

$$y = 20 * \sin(x - \pi/2) + 4 \quad (5.3)$$

- Pie izquierdo:

$$y = 20 * \sin(x - \pi/2) - 4 \quad (5.4)$$

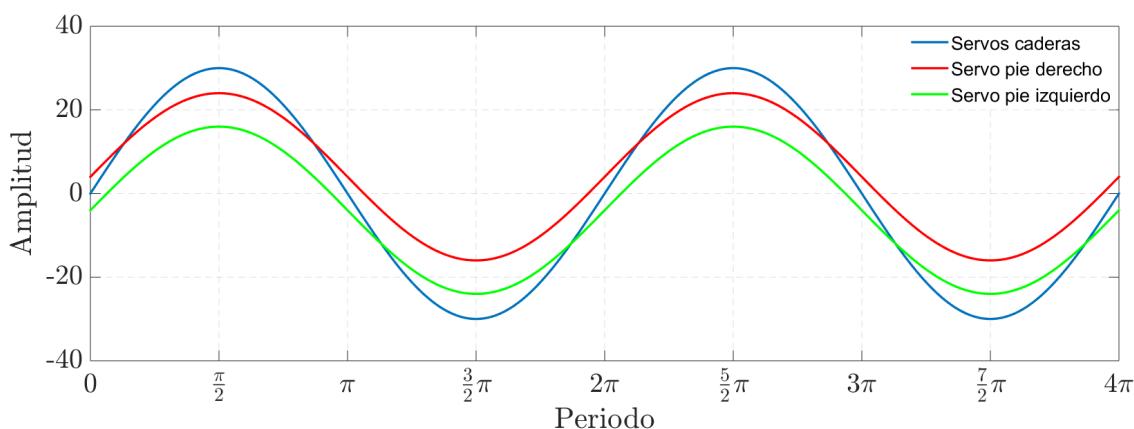


Figura 5.7: Funciones sinusoidales correspondientes a las caderas y a los pies.

La Figura 5.7 muestra las gráficas correspondientes a las ecuaciones anteriores, donde se puede observar cómo afectan los distintos parámetros a las mismas.

En primer lugar, cabe destacar la gráfica azul, correspondiente al giro de los servos de las caderas. Se trata de un movimiento simétrico que permite a Zowi avanzar hacia delante o hacia atrás, simulando los pasos dados por los seres humanos. En cada período de 2π el robot mueve alternativamente una pierna en el sentido adecuado, repitiendo el proceso a medida que lo hace la onda sinusoidal.

El hecho de que la misma cifra permita colocar los motores en posiciones distintas se debe a la orientación y al sentido de giro de los mismos. En la Figura 5.4 se puede ver este comportamiento: 60° en la cadera mostrada a la izquierda no se corresponden con un ángulo simétrico en el lado contrario, sino con el complementario. En la Figura 5.8 se representa este comportamiento sobre la posición que tendría Zowi en ese momento.

Estas modificaciones hacen posible la práctica consecución del objetivo: **la obtención de una cifra en grados comprendida entre 0 y 180°** que pueda ser asignada al servo. No obstante, las mencionadas funciones sólo proporcionan la variación con respecto a la posición base; es decir, el número de grados que se tendría que modificar la posición del servo con respecto a 90° .

Para obtener los valores absolutos, es necesario **sumarle al conjunto un componente constante de 90°** , que centraría la gráfica en esta cifra en lugar de en 0. Esta operación no sería necesaria si la posición de reposo equivaliera a 0° , pero como se ha explicado con anterioridad, no es el caso.

Como se ha podido comprobar, **el algoritmo es muy ingenioso y permite que se pueda disfrutar de una funcionalidad tan básica como el desplazamiento cuando se usa el robot**. No obstante, no se debe olvidar una limitación importante en lo que respecta a la precisión del conjunto: Zowi no camina recto, se ladea al cabo de pocos pasos.

El hecho de que no se pueda controlar el ángulo de giro dinámicamente, sino que lo defina una función preestablecida, limita mucho las opciones para intentar encontrar una solución a este problema. Por lo tanto, una modificación de este algoritmo podría resultar la solución más conveniente. Este tema se discutirá y explicará en la sección [Algoritmo para caminar](#).

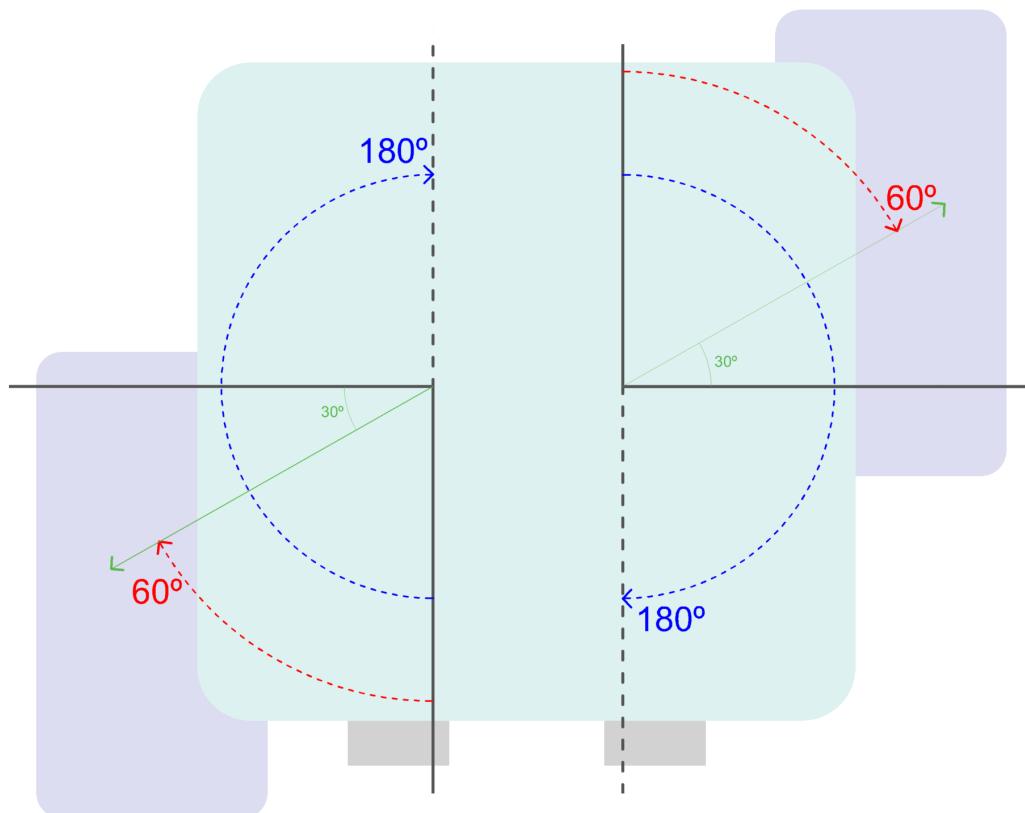


Figura 5.8: Ángulo -valor de la función sinusoidal- que provoca el movimiento de Zowi hacia delante o hacia atrás.

5.2. ZowiApp. ¿Está la aplicación a la altura del robot?

ZowiApp es la aplicación proporcionada por **BQ** para manejar a **Zowi**. Dado que el código fuente no está disponible para su descarga, no se ha podido tomar como referencia para el nuevo software Android, pese a lo cual se pueden evaluar su estética y sus opciones.

Las secciones con las que cuenta son las siguientes:

- **GamePad:** controles táctiles para que Zowi camine, realice gestos combinando movimientos y sonidos, o pinte en la boca distintos bits de la matriz.
- **3, 2, 1 ¡Acción!:** proporciona una línea temporal en la que se pueden encadenar acciones de Zowi. Por ejemplo, primero pintar una boca, después realizar un paso de baile, y acabar pintado otra boca. **Permite que los niños aprendan el concepto de secuenciación.**
- **Repite con Zowi:** juego en el que Zowi realiza una serie de movimientos que deben ser recordados y registrados a continuación en la aplicación.
- **Pinta bocas:** el robot pinta una boca en su matriz de LEDs, y el niño tiene que reproducirla.
- **Editor de bocas:** procedimiento contrario al anterior, donde el usuario define una boca en un editor y esta será representada por Zowi.

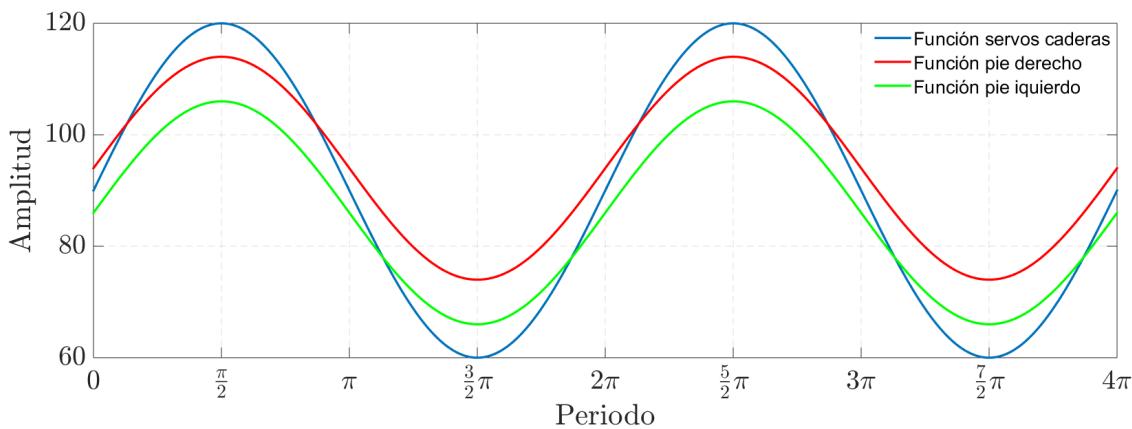


Figura 5.9: Valores absolutos de los servos en el algoritmo de caminar.

También existe un sistema de logros, donde se desbloquea contenido a medida que se usa este desarrollo, y un apartado especial -y realmente interesante- en el que **se explican los fundamentos teóricos del robot**. Este último redirige a la página zowi.bq.com/es/projects/, en la que se detalla qué hace posible que este se mueva, mida distancias con los ojos, etc.

Detallando las funciones de la app, se puede corroborar lo que se afirmó con anterioridad en este escrito: **tiene poco recorrido**. Permite, junto con el software cargado por defecto en la placa de Zowi, evaluar las posibilidades del conjunto y hacer uso de todas sus características, pero **no constituyen una fuente de aprendizaje**; proporcionan un juego o pasatiempo, hecho que si bien es correcto, no suma tanto como podría.

Este inconveniente, no obstante, se resolverá con la nueva ZowiApp desarrollada en este TFM.

5.2.1. Comunicación de Zowi con la aplicación Android

Una vez repasados los dos actores principales -y únicos- del sistema, cabe realizar un repaso a la **comunicación entre ambos**. Esta es crucial para conseguir que las intenciones del niño se vean reflejadas en las reacciones de Zowi, posibilitando una dualidad causa-efecto muy importante de cara a su desarrollo.

La tecnología de la que se hace uso es el Bluetooth, como es muy habitual en este tipo de dispositivos. Una vez establecido el socket, **desde la aplicación se envían comandos con el siguiente formato:**

comando parámetro1 parámetro2 ... parámetroN

Siendo los mostrados a continuación dos ejemplos de uso real:

L 0001011100101001011111001000

T 1000 5000

De este modo, el comando L con el parámetro numérico mostrado ilumina la boca de LEDs de forma acorde a lo **explicado anteriormente**, mientras que el segundo haría que se emitiera un pitido con una frecuencia de 1000 Hz durante 5 segundos.

El diagrama de flujo de este proceso puede verse en la Figura

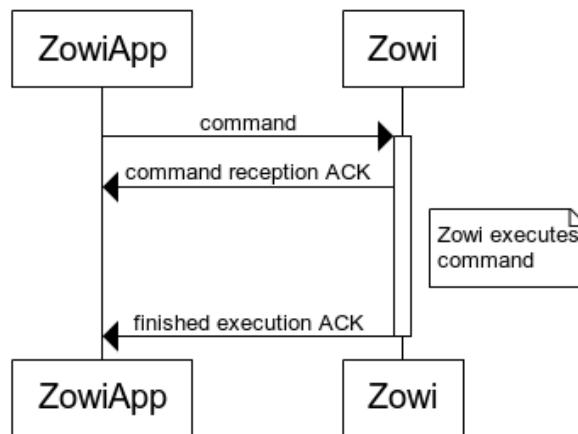


Figura 5.10: Flujo de comandos Bluetooth enviados a Zowi esde ZowiApp.

Puerto serie Arduino

El puerto serie permite la comunicación de la placa con otros dispositivos, como el ordenador o el móvil -mediante Bluetooth-. Como su propio nombre indica, la información se envía mediante una secuencia de bits, contrariamente a lo que ocurre en las comunicaciones en paralelo con múltiples canales.

Al ser bidireccional, **los dos extremos pueden escribir datos** de forma que sean recibidos en la otra parte del canal. Uno de los casos más habituales es que lo haga Arduino, para poder registrar qué flujo está siguiendo un determinado programa.

```

1 int counter = 0;
2 while (counter < 3) {
3     Serial.print("Counter: "); Serial.println(counter);
4     counter++;
5 }
6 delay(1000);

```

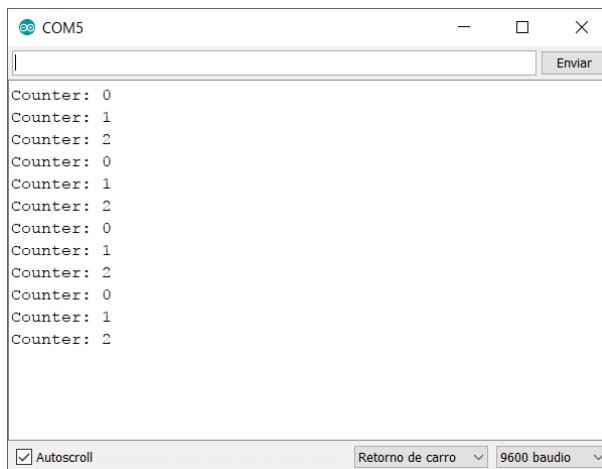


Figura 5.11: Código de ejemplo escrito por Arduino en el puerto serie.

No obstante, **la placa también puede leer aquello que le llegue de forma sencilla**. Por tanto, es perfectamente posible enviar números o caracteres desde el ordenador o, en el contexto de este TFM, desde la aplicación Android.

```

1 if (Serial.available()) {
2     int receivedNumber = Serial.read();
3     Serial.print("ReceivedNumber: ");
4     Serial.println(receivedNumber);
}

```

Envío de comandos desde ZowiApp

Teniendo en cuenta lo expuesto en la sección anterior, resta explicar cómo se lleva a cabo el envío de comandos desde la aplicación Android.

Una vez establecido el socket entre ambos extremos, el único paso restante es escribir en él **una cadena de texto con el formato correcto**. Es decir, “L 0001011100101001011111001000” o “T 1000 5000”.

En el instante en el que se reciben estos Strings en el robot, este los interpreta de la forma expuesta a continuación.

5.2.2. Recepción de comandos en Zowi

Zowi incorpora por defecto el código necesario para **recibir comandos y ejecutar una función en consonancia**.

Para ello, en el *setup*-código que se ejecuta una sola vez al encender el sistema, donde se ubica la configuración y puesta a punto del mismo- se realiza un mapeo de letras a métodos.

```

1 SCmd.addCommand("S", receiveStop);
2 SCmd.addCommand("L", receiveLED);
3 SCmd.addCommand("T", recieveBuzzer);
4 SCmd.addCommand("M", receiveMovement);
5 SCmd.addCommand("H", receiveGesture);
6 SCmd.addCommand("K", receiveSing);
7 SCmd.addCommand("C", receiveTrims);
8 SCmd.addCommand("G", receiveServo);
9 SCmd.addCommand("R", receiveName);
10 SCmd.addCommand("E", requestName);
11 SCmd.addCommand("D", requestDistance);
12 SCmd.addCommand("N", requestNoise);
13 SCmd.addCommand("B", requestBattery);
14 SCmd.addCommand("I", requestProgramId);

```

Donde “SCmd” es un objeto de la clase encargada de gestionar el puerto serie, y “addCommand” es una de sus funciones.

Esta última tiene como labor almacenar en un array estructuras de tipo “struct”. Cada una de ellas, a su vez, tiene asociadas dos variables.

```

1 typedef struct _callback {
2     char command[SERIALCOMMANDBUFFER];
3     void (*function)();
4 } ZowiSerialCommandCallback;

```

- **command**: cadena de texto -con tamaño máximo SERIALCOMMANDBUFFER- que **identifica al comando** -L o T, en los casos mostrados anteriormente-.
- **(*function)()**: puntero a la **función a ejecutar** cuándo se reciba el comando adecuado.

Por lo tanto, el resultado derivado de la ejecución del *setup* es un array de estructuras, mediante el cual a cada cadena de texto le corresponde un método. Así, en función del String que se reciba se ejecuta el método adecuado.

Para ello, cuando la placa base detecta que hay contenido esperado a ser leído, lo almacena en un buffer local y va leyendo tokens -carácter o conjunto de caracteres cuyo delimitador es un espacio-. El primero siempre corresponde al identificador de la acción, mientras que el resto son los parámetros de la misma.

Capítulo 6

Análisis de ZowiApprende

Este capítulo constituye la primera etapa a nivel técnico del Proyecto, donde se analizan aspectos como la definición del sistema, los requisitos -funcionales y no funcionales- o los casos de usos que se han evaluado antes de comenzar el desarrollo.

6.1. Definición del sistema

ZowiApprende consistirá, como se ha repasado en anteriores apartados, en una aplicación Android que permitirá que los niños interactúen con Zowi.

6.1.1. Aplicación Android

La aplicación Android será un **software desarrollado en Java 7 y diseñado para ejecutarse en una tablet** de características concretas: pantalla de 10.1" y resolución HD (1280x720). **El IDE utilizado será el entorno oficial de Google, Android Studio**, haciendo uso del emulador en caso de no disponer del dispositivo correcto.

Antes de describir la estructura, cabe destacar la **importancia de la interfaz** de cara a una experiencia de uso satisfactoria. Dado que el público objetivo son niños de entre 3 y 5 años, la **apariencia deberá ser sencilla, sin alardes o efectos visuales complejos, botones grandes y que destaque por su facilidad de uso**: ausencia de desplegables, de opciones ocultas, de gestos, etc.

De este modo, se basará toda la estética en botones y textos amplios, cuyo impacto visual y significado sean complementados por medio de imágenes aclaratorias. Se intentará que este apartado sea coherente con el robot Zowi; por ejemplo, definiendo una curvatura en las esquinas de alguna pantalla, haciendo que se asemeje a la curvatura de la cabeza del robot.

Al iniciar la aplicación, se lanzará un menú con dos botones, cuya función es escoger una de las modalidades de trabajo disponibles: **juego libre o dirigido**. Escogiendo el primero, se accederá a un menú que mostrará actividades de temática y dinámicas variadas; el segundo, por su parte, dirigirá al usuario a una pantalla donde, en disposición vertical, se mostrarán el título de la unidad y seis actividades para cada uno.

Los ejercicios se diseñarán en colaboración con una maestra de Educación Infantil, contextualizándolos en un trimestre escolar. De esta forma, además de calidad de código, se conseguirá que el **contenido que se aprenda con la aplicación sea adecuado en base a la legislación vigente**.

6.1.2. Programa Arduino

ZowiApprende se comunicará mediante bluetooth con Zowi, que proporcionará feedback sobre las decisiones tomadas en la app y permitirá que la interacción de los alumnos con la aplicación sea mucho más productiva.

Se hará uso de las funciones de fábrica para acciones predefinidas, como ejecución de gestos y animaciones de alegría, tristeza, sorpresa... Así como de parte del código requerido para caminar o girar moviendo los servos. Asimismo, se tomará como **base para todas aquellas modificaciones necesarias**.

Se definirán métodos y eventos que permitirán escoger qué fragmento de código ejecutar en función del comando recibido desde la aplicación, respondiendo así a los requerimientos de la actividad que se esté intentando resolver.

Con respecto a nuevo código, y dado que se ampliará la funcionalidad de fábrica con nuevos sensores y actuadores, **se hará uso de la documentación existente en el ecosistema de Arduino para desarrollar nuevas características**.

Para acabar, **se simplificará el flujo de comunicación mediante bluetooth**, al considerar que se envían multitud de ACKs innecesarios para el correcto desempeño del sistema.

6.2. Requisitos

A continuación, y basándose en los objetivos y el alcance del Proyecto, **se detallan los requisitos evaluados** y que deberán tenerse en cuenta a lo largo del desarrollo.

La información que se muestra se basa en 3 parámetros:

- **Código**: identificador del requisito.
- **Nombre**: identificador del requisito, en formato comprensible.
- **Descripción**: explicación del requisito.

6.2.1. Requisitos ZowiApprende

La Tabla 6.1 muestra los requisitos funcionales de la aplicación Android, ZowiApprende.

Por su parte, la Tabla 6.2 muestra los requisitos no funcionales de la app.

6.2.2. Requisitos Zowi

6.3. Identificación de actores del sistema

Una vez evaluados los requisitos de la nueva ZowiApprende, es importante llevar a cabo la **identificación de los distintos actores involucrados en el sistema**. En este caso, se han diferenciado los siguientes agentes:

- **Maestra**: la maestra que hará uso de ese desarrollo tendrá que -en algunos casos- **supervisar el desempeño de la aplicación y de Zowi**. Además, en varios tipos de actividad será recomendable su intervención en la resolución de las mismas.

Código	Nombre	Descripción
R1	Uso de Zowi	ZowiApprende requerirá el uso de Zowi en la mayoría de aplicaciones
R2	Juego libre y dirigido	Se tendrán dos tipos de juego bien diferenciados
R2.1	Actividades variadas en juego libre	Se diseñarán ejercicios originales sin una temática concreta
R2.2	Actividades coherentes en juego dirigido	Se contextualizarán estos ejercicios según la legislación vigente
R3	Dos niveles de profundidad	Se accederá al contenido con un máximo de 2 clics
R4	Interfaz adaptada a tablet	Se tendrán en cuenta las características de la pantalla de la tablet (10"HD)
R5	Conexión bluetooth	Se enviarán comandos a Zowi mediante bluetooth
R6	Android 4.4	Se usará el software necesario para que la app sea compatible con Android KitKat
R7	Actividad cuadrícula	Se requiere la implementación de una actividad donde Zowi se desplace por una cuadrícula de 3x3 ó 4x4

Tabla 6.1: Requisitos funcionales de ZowiApprende.

- **Alumnos:** los niños harán **uso directo de la aplicación Android y de Zowi**, sirviendo la primera de pasarela para controlar e interactuar con el segundo.
- **Zowi:** el robot constituye la pieza principal del conjunto. Además de **reaccionar a las respuestas de cada ejercicio**, incrementa el interés que los usuarios pueden llegar a tener en el Proyecto.

6.4. Identificación de los subsistemas

La siguiente etapa en el proceso de análisis consiste en la **identificación de los subsistemas principales del conjunto**.

6.4.1. Descripción de subsistemas

Tal y como se ha mencionado en numerosas ocasiones a lo largo de este escrito, este Proyecto cuenta con dos elementos bien diferenciados que se pueden identificar como los subsistemas existentes.

- **Aplicación Android:** subsistema con el que interactuarán los niños directamente. Sus funciones incluyen el **mostrado de las actividades, la ejecución de toda su lógica, su corrección y el envío de comandos a Zowi**.
- **Zowi:** al ser su -prácticamente- única función la de reaccionar de una forma u otra a las acciones llevadas a cabo con la tablet, la lógica será considerablemente más sencilla.

Código	Nombre	Descripción
R8	Diferenciación colores	Se distinguirán las dos secciones con tonalidades en dos colores distintos
R9	Estética basada en Zowi	Se creará una interfaz que recuerde a Zowi de alguna manera
R10	Feedback correcta finalización	Se mostrará una alerta en pantalla indicando que se ha completado con éxito una actividad
R10.1	Mensajes de ánimo	Se mostrarán mensajes motivadores al completar con éxito una actividad
R10.2	Reiniciar actividad o volver a menú al completar	Al terminar un ejercicio, se mostrará un popup donde se podrá reiniciar la actividad o volver al menú
R11	Alta escalabilidad	La estructura de la app se ideará de forma que añadir nuevas actividades sea inmediato
R12	Alta rejugabilidad	La aplicación se diseñará de forma que sea rejutable, evitando en la medida de lo posible que los niños se aburran
R13	Contenido aleatorio	El contenido de las actividades será escogido aleatoriamente entre varias opciones disponibles
R14	Gestionar conexión Zowi	Contemplar la posible no conexión de la aplicación Android con Zowi
R15	Gestionar desplazamiento en pantalla	Muchas actividades consisten en arrastrar elementos en pantalla, por lo que se pretende que este sea fluido y evitar que se salgan de los límites de la misma
R16	Utilizar imágenes libres	La mayoría de imágenes disponibles en Internet tienen derechos de autor, por lo que sería aconsejable usar aquellas que no tengan este inconveniente

Tabla 6.2: Requisitos no funcionales de ZowiApprende.

6.4.2. Descripción de las interfaces entre subsistemas

Llegados a este punto, y habiendo identificado los distintos subsistemas, es el turno de la **comunicación entre ambos**.

Todo **envío de información** entre ambas partes **tendrá lugar en el ámbito local**, sin ser necesaria conexión a Internet para llevar a cabo ninguna acción. Más concretamente, **se usará la tecnología Bluetooth** -disponible en la totalidad de dispositivos Android y en la placa basada en Arduino de la que dispone Zowi- para el envío de datos entre los dos extremos.

El comportamiento es muy similar al definido de fábrica, con alguna diferencia -simplificación- en el envío de paquetes. En ZowiApprende, se requerirá un ACK sólo en caso de ser necesario feedback; en caso contrario, se enviará sin esperar una respuesta.

Código	Nombre	Descripción
R17	Recepción de comandos	Se requerirá la recepción e interpretación de comandos desde la aplicación Android
R18	Mejora del algoritmo de caminar	Se mejorará el algoritmo de caminar para que Zowi no se tuerza tanto al hacerlo
R19	Nuevas interacciones con los alumnos	Se desarrollarán nuevas formas de interactuar con los niños

Tabla 6.3: Requisitos funcionales de Zowi.

Código	Nombre	Descripción
R20	Ampliación de sensores y actuadores	Se añadirán nuevos componentes para mejorar la funcionalidad de fábrica de Zowi
R21	Desarrollo de nuevo código	Se creará el código adecuado para el manejo de los nuevos componentes
R22	Impresión 3D de nueva carcasa 3D	Se diseñará e imprimirá una nueva carcasa para albergar los nuevos componentes

Tabla 6.4: Requisitos no funcionales de Zowi.

6.5. Especificación de casos de uso y escenarios

En esta sección se detallan los casos de uso que se pueden llegar a dar en el uso de ZowiApprende. Dado el flujo de uso y el número de acciones a plasmar en este tipo de diagramas, se ha decidido **agrupar los distintos escenarios en un sólo esquema que se puede ver en la Figura 6.1**.

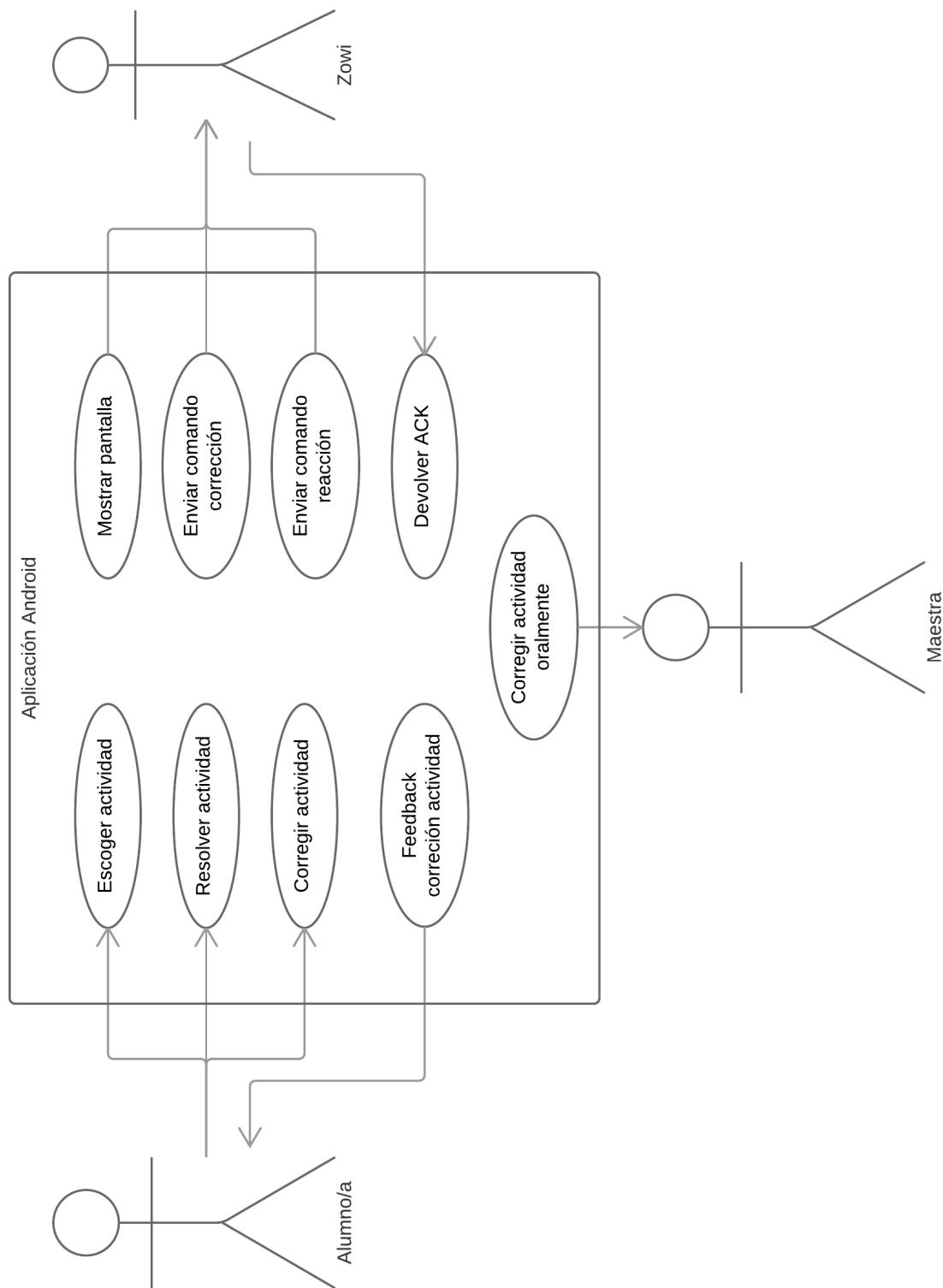


Figura 6.1: Agrupación de los casos de uso de ZowiApprende.

Nombre	Vinculación de Zowi con ZowiApprende
Descripción	Se lleva a cabo la vinculación inicial del robot con la aplicación Android mediante Bluetooth
Actores	Iniciado y terminado por profesor u alumno
Precondiciones	Tablet y Zowi encendidos con ZowiApprende instalada
Escenario principal	Se inicia la aplicación Androide por primera vez. La información sobre la dirección Bluetooth del robot no está almacenada, por lo que se obtiene y se guarda localmente para sucesivas interacciones.
Escenarios secundarios	Tras el rastreo bluetooth, no se encuentra a Zowi. Se habilitará una ventana emergente dónde se permite repetir el proceso

Tabla 6.5: Caso de uso de vinculación Bluetooth de los actores “Zowi” y “Aplicación Android.”

Nombre	Ejecución de ZowiApprende
Descripción	Se inicia la aplicación Android y se conecta con Zowi
Actores	Iniciado y terminado por profesor u alumno
Precondiciones	Tablet encendida con ZowiApp instalada
Escenario principal	Se inicia la aplicación Android y se muestra una ventana emergente indicando que se está estableciendo la comunicación con Zowi. Al cabo de pocos segundos, esta se completa, el robot lo indica con un gesto y el ícono de conexión de la interfaz se colorea de azul.
Escenarios secundarios	Zowi no está encendido o no se encuentra. Se muestra una ventana emergente informando de la situación y desde donde se puede iniciar la búsqueda de nuevo.

Tabla 6.6: Caso de uso de ejecución del actor “Aplicación Android”.

Nombre	Selección de tipo de juego
Descripción	El alumno escoge juego libre o dirigido pulsando uno de los dos botones disponibles
Actores	Iniciado y terminado por profesor u alumno
Precondiciones	Zowi debe estar conectado
Escenario principal	Se abre el menú en el que se ubican las distintas actividades
Escenarios secundarios	Zowi no está conectado y no se permite acceder al contenido

Tabla 6.7: Caso de uso de selección de tipo de juego en el actor “Aplicación Android”.

Nombre	Acceso a una actividad
Descripción	El alumno aprieta el botón correspondiente a alguna de las actividades disponibles
Actores	Iniciado y terminado por el alumno
Precondiciones	Zowi debe estar conectado
Escenario principal	Se carga la interfaz de la actividad para que el niño interactúe con ella
Escenarios secundarios	-

Tabla 6.8: Caso de uso de selección de actividad en el actor “Aplicación Android”.

Nombre	Corrección de actividad por parte del actor “Alumno/a”
Descripción	El alumno aprieta el botón de “Corregir” en alguna de las actividades
Actores	Iniciado y terminado por el alumno
Precondiciones	Zowi debe estar conectado
Escenario principal	Zowi reacciona positiva o negativamente en función de la respuesta del alumno/a
Escenarios secundarios	

Tabla 6.9: Caso de uso de corrección de actividad por parte del actor “Alumno/a”.

6.6. Análisis de interfaces de usuario

Aunque el **diseño de la interfaz se irá adaptando a lo largo del desarrollo** en función de parámetros funcionales o meramente estéticos, **se ha elaborado una serie de mockups con el fin de tener una base de referencia** y de mostrarle las pautas básicas a la maestra que usará la aplicación.

6.6.1. Pantalla de selección de tipo de juego

La siguiente pantalla mostrada en ZowiApprende consistirá en la **selección de juego libre o dirigido**. La interfaz será muy simple, conteniendo únicamente los botones -grandes, de forma que sean fáciles de pulsar- necesarios.



Figura 6.2: Pantalla de selección de tipo de juego.

6.6.2. Juego libre

El menú de actividades del juego libre **muestra en una lista aquellos ejercicios disponibles**.



Figura 6.3: Pantalla de selección de actividad en juego libre.

6.6.3. Juego dirigido

El juego dirigido cuenta con **más complejidad que la modalidad anterior**, al contextualizarse en una temática concreta y subdividirse en diferentes unidades. Por lo tanto, existirán **dos niveles de profundidad**: uno correspondiente a la selección del tema (Figura 6.4), y otro a la del ejercicio (Figura 6.5).

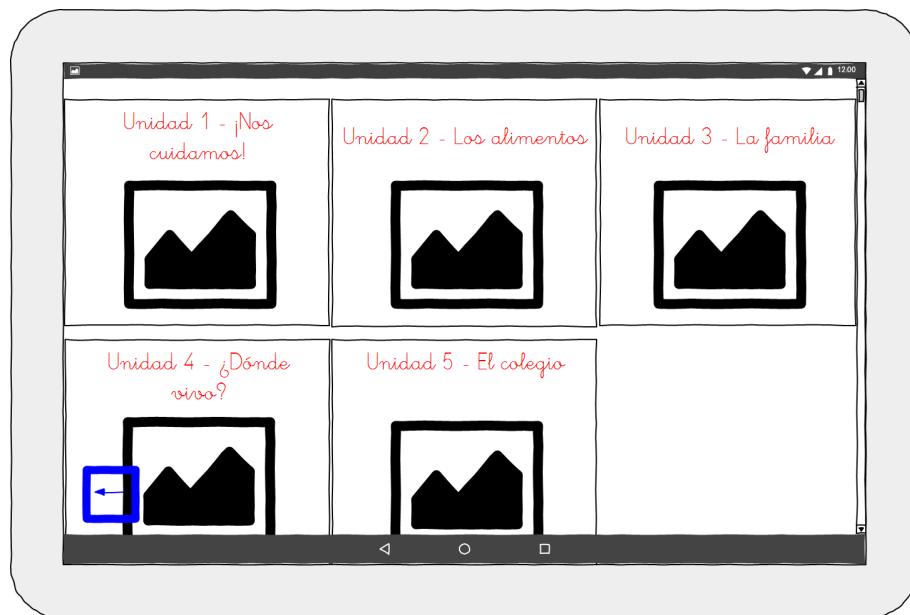


Figura 6.4: Pantalla de selección de tema en juego dirigido.

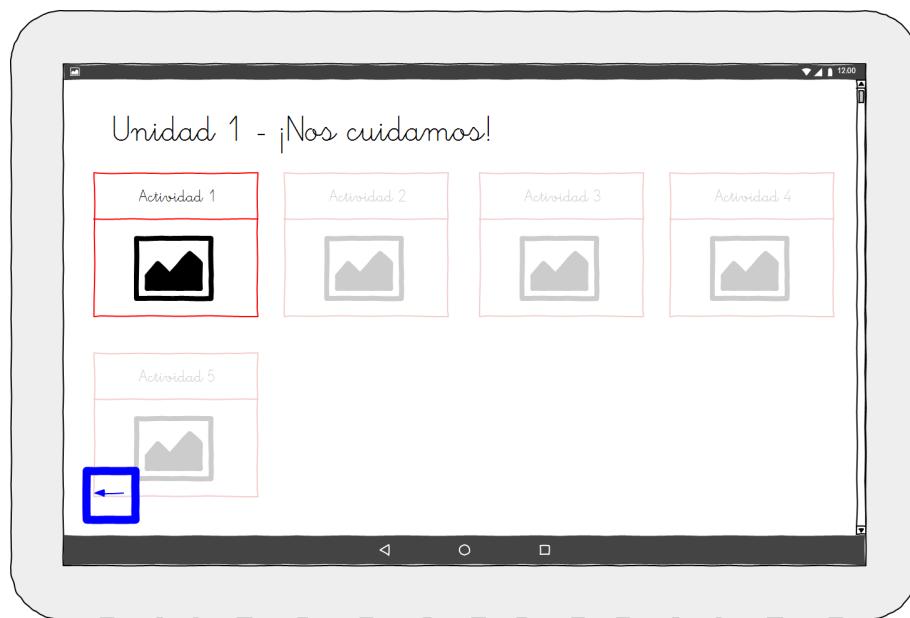


Figura 6.5: Pantalla de selección de actividad en juego libre.

6.6.4. Actividad de cuadrícula

Esta actividad constituye **uno de los requisitos del Proyecto**, habiendo hablado de ella en [Objetivos y requisitos del Proyecto](#).

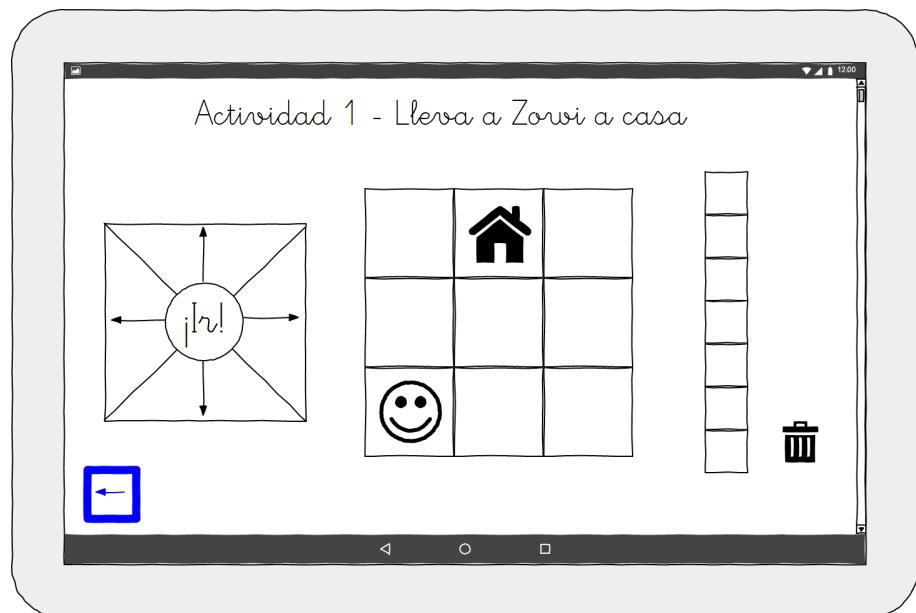


Figura 6.6: Actividad de la cuadrícula (modo sencillo).

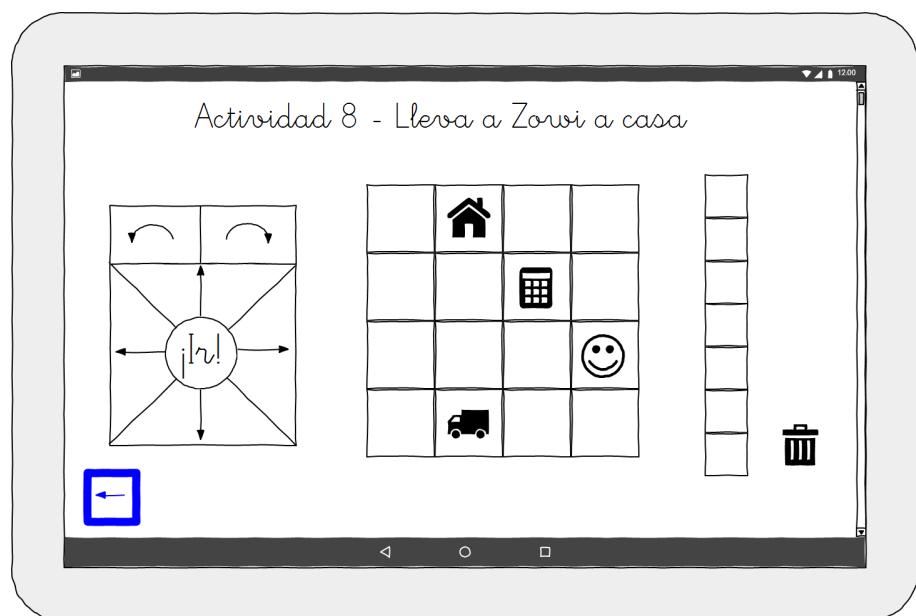


Figura 6.7: Actividad de la cuadrícula (modo difícil).

6.6.5. Actividad de bloques lógicos

Esta pantalla muestra una **representación de lo que será la actividad de bloques lógicos**.

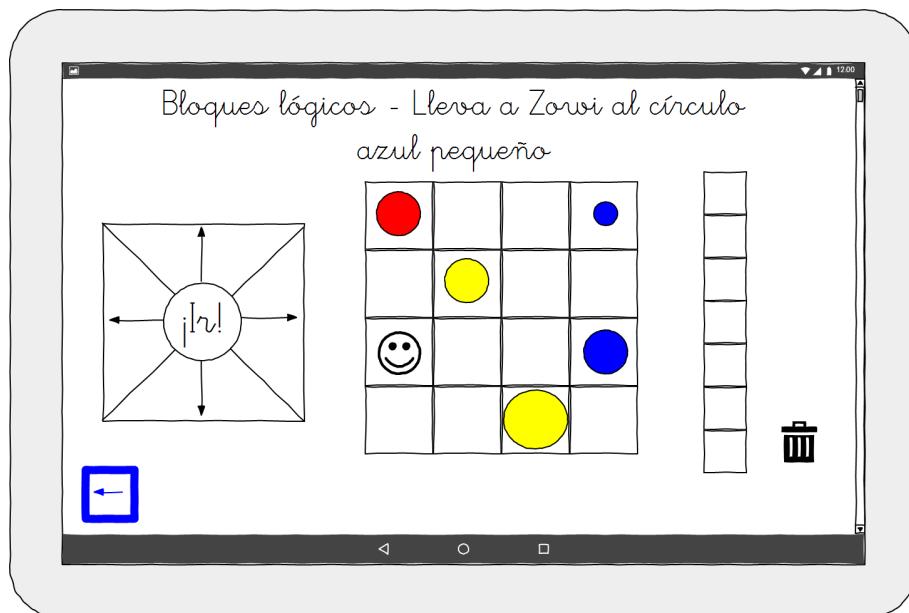


Figura 6.8: Actividad de bloques lógicos.

6.7. Diagramas de navegabilidad

Los diagramas de navegabilidad permiten visualizar el flujo de determinadas acciones de un Proyecto, esclareciendo tanto los pasos necesarios para llevarlas a cabo como las posibilidades del sistema.

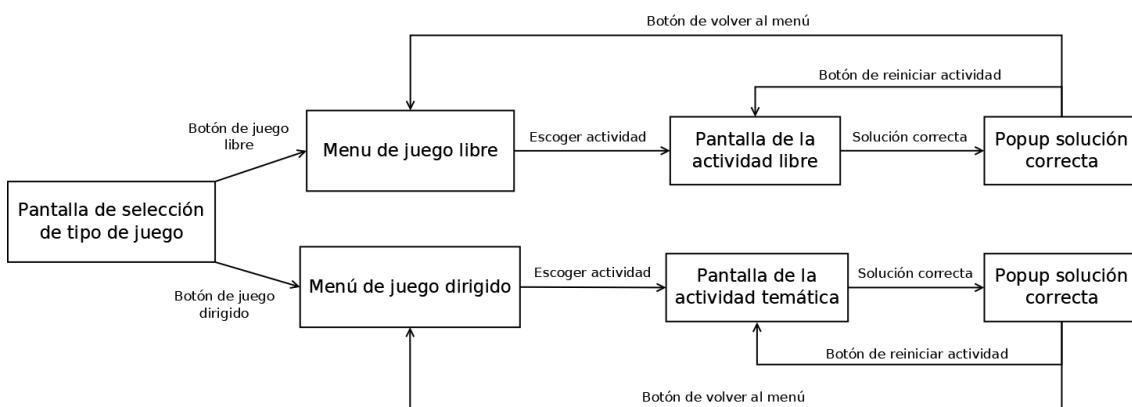


Figura 6.9: Diagrama de navegabilidad de ZowiApprende.

6.8. Definición del plan de pruebas

En esta sección se definen las distintas etapas seguidas en el proceso de **testeo de la aplicación**, corroborando el correcto funcionamiento tanto de bloques de código individuales, como del sistema completo.

6.8.1. Pruebas unitarias

Este tipo de pruebas consisten en la **certificación del correcto funcionamiento de bloques independientes** -y generalmente, de poca extensión- de código. Estos albergan

cierta lógica por sí mismos, y deberían poder ser ejecutados sin depender de factores externos.

Para ello, una opción posible es el *mockeo* de sus relaciones con el exterior; es decir, **simular un determinado comportamiento de las mismas, en función de las cuales cada fragmento testeado debe responder de una determinada manera**.

En el caso de **ZowiApprende**, el núcleo de su funcionalidad reside en la cohesión de las distintas partes del conjunto, basándose en comunicación entre clases y actividades, el envío de información por medio de Bluetooth, etc. De este modo, que cada bloque de código dependa de otros fragmentos es común en el sistema, **considerando las pruebas unitarias de menor utilidad** de la debida para considerarlas en este TFM.

No obstante, se tuvieron en cuenta en el caso de un **algoritmo recursivo creado para la generación de números aleatorios dentro de unos límites**, que posibilita la carga de distintas imágenes cada vez que un usuario acceda a una actividad.

6.8.2. Pruebas de integración, de sistema y de usabilidad

En estas pruebas reside la responsabilidad de asegurar el *correcto comportamiento del sistema o de módulos del mismo*. No se centran en el testeo de pequeños fragmentos de código, sino que su objetivo consiste en evaluar el desempeño de unidades de lógica mayor.

De este modo, es posible **determinar el adecuado funcionamiento de una de las actividades, de algún tema en concreto o de los dos tipos de juego -libre y dirigido-**.

Las pruebas de integración y de sistema se combinarán con las de usabilidad, donde **un usuario externo usará ZowiApprende para plasmar cuáles son tanto sus impresiones, como los fallos o aspectos mejorables encontrados**.

Capítulo 7

La nueva ZowiApprende, aprendiendo con Zowi

Ya se ha comentado en repetidas ocasiones **el principal inconveniente del conjunto del robot y la aplicación**, y no es otro sino **la incapacidad para aprender en profundidad**. Si bien los niños de edad más avanzada pueden experimentar con la programación por bloques y Arduino, este contenido queda muy lejos de los pequeños de Educación Infantil.

Para estos últimos, **la experiencia se limita a lo que pueda proporcionarles la aplicación**. No se puede pretender que realicen investigaciones externas a este pequeño ecosistema o aprendan de forma voluntaria y proactiva, por lo que en este caso **recae sobre el desarrollador y los educadores la labor de acercarles el conocimiento**.

7.1. Nuevo contenido, pensado para enseñar

La aplicación no sólo se ha creado de cero en lo referente al diseño y la interfaz, sino que **el contenido se ha ideado teniendo en cuenta el proceso evolutivo de los niños**.

Como se ha comentado en el [apartado correspondiente a los objetivos](#), la aplicación se divide en **dos grandes secciones**: juego libre y juego dirigido, diferenciadas por colores.

En el primer caso, en la interfaz predomina el color verde, mientras que en la segunda lo hace el rojo. Esta diferenciación se ha tenido en cuenta con el fin de que los niños puedan distinguir de un simple vistazo en qué rama de la aplicación se encuentran, sin tener que recurrir a leer textos o pulsar la tecla “Atrás”.

La pantalla principal, que sirve de puente para acceder a todo el contenido disponible, **dispone de una estética diseñada tomando la cabeza de Zowi como base**. De esta forma, sus ojos constituyen los dos botones que permiten avanzar a los siguientes niveles de la app, como se puede ver en la Figura 7.1.



Figura 7.1: Menú principal de la nueva ZowiApp.

7.2. Aleatoriedad de contenido

Antes de describir las distintas actividades implementadas, cabe destacar un aspecto relevante en el contexto de las mismas: **la aleatoriedad del contenido**.

Diseñar ejercicios con imágenes o textos distintos en cada iteración es prácticamente irrealizable, pero entre este fin ideal y mostrar siempre los mismos recursos existe un punto intermedio. De este modo, y teniendo en cuenta la importancia de las primeras en una aplicación infantil, **se ha creado un código que muestra un determinado número de ellas aleatoriamente**, variando también sus posiciones.

Por ejemplo, en el caso de una actividad con 5 imágenes, estas se escogen al azar de entre un conjunto mayor disponible, variando también su ubicación en pantalla con cada nuevo acceso. **Este detalle alarga enormemente la vida útil de la aplicación**, ya que los alumnos no pueden memorizar las respuestas correctas al depender siempre de un componente aleatorio.

Figura 7.2: Actividad de la cuadrícula de colores.

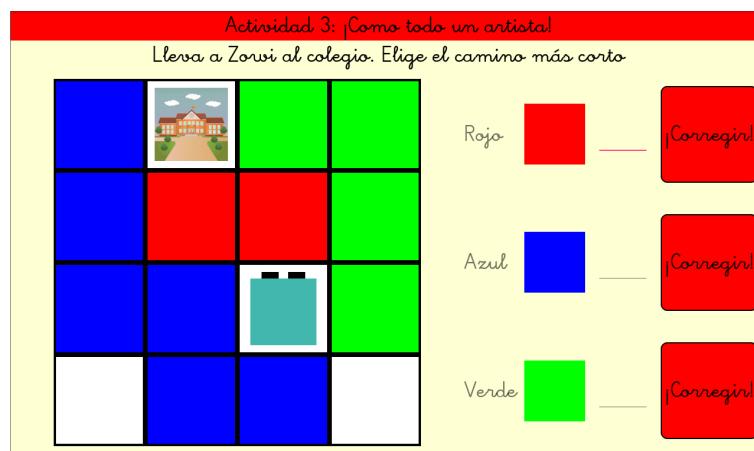


Figura 7.3: Actividad de la cuadrícula de colores con distinto contenido.

7.3. Sección de juego libre

Esta sección consiste en la **agrupación de ejercicios de temática variada**, que permite a los niños trabajar de forma interdisciplinar mientras se divierten. Se desarrollan así habilidades plásticas, lógico-matemáticas y espaciales, junto con otras destrezas relacionadas con el lenguaje verbal.

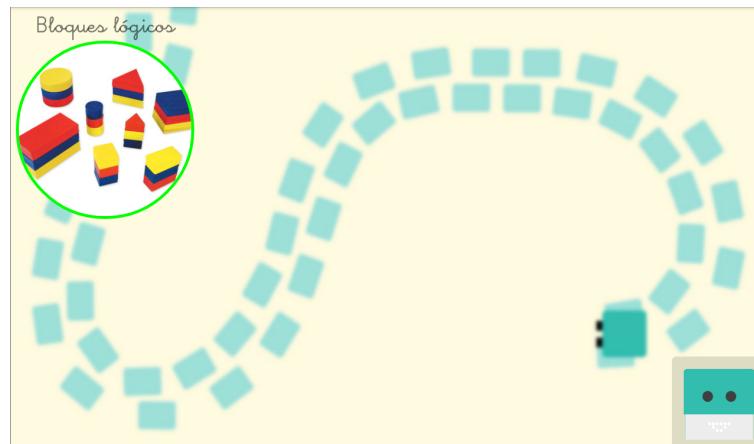


Figura 7.4: Menú de la sección de juego libre.

7.4. Sección de juego dirigido

La sección de juego dirigido constituye la más importante de las dos existentes, donde **se agrupan todas las actividades estructuradas por temas**. Al contrario que en Juego libre, el contenido de los ejercicios es importante, ya que todos ellos tienen que estar contextualizados en un ámbito que se trate en las aulas de Educación Infantil.

Si bien inicialmente se diseñó este apartado con 3 niveles de profundidad -escoger Juego dirigido, escoger tema y escoger actividad-, para reducir el número de pasos se optó por **reunir en una misma pantalla los temas y los ejercicios**. Es decir, el momento en el que el niño pulse sobre el botón correspondiente del menú principal, aparecerán en pantalla todas las actividades precedidas por el título de la unidad.

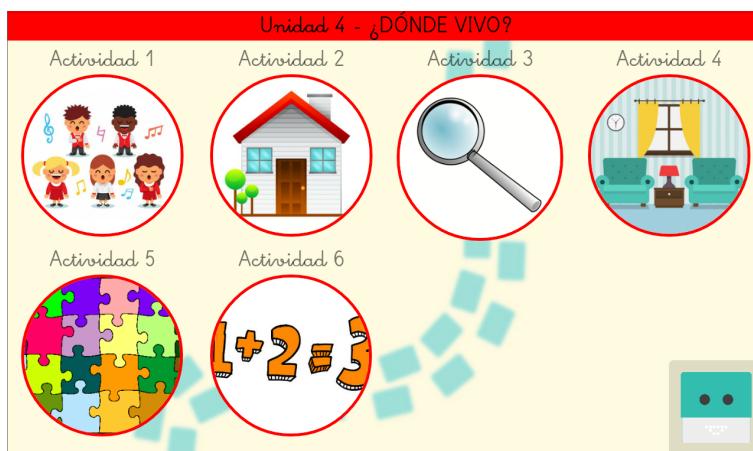


Figura 7.5: Menú de la sección de juego dirigido.

En este punto, es necesario hacer un repaso al desarrollo de los ejercicios. Se diseñaron **distintas plantillas con estructura independiente de los textos o imágenes utilizadas**, de forma que el contenido pueda ser modificado con un esfuerzo mínimo.

Así, algunas de estas plantillas se han reutilizado en distintos temas, pudiendo adaptarse a ellos de forma inmediata. El mecanismo utilizado ha sido **el uso de un recurso XML, donde para cada uno de los 30 ejercicios se especifican algunos parámetros** como el tipo, las imágenes de que contendrá, y otros argumentos necesarios para su correcto funcionamiento. Por ejemplo, a continuación se muestra la línea correspondiente al Memory:

```

1 <item>{"description": "Descripcion 10\\", "type": "MEMORY", "images": [zowi_pointer,
2 green_circular_button, red_circular_button, house, zowi_3_teeth, black]}</item>

```

Cabe destacar que **en este caso, esta información no incluye la corrección del ejercicio**. El motivo es que al generarse las imágenes aleatoriamente en posiciones distintas, resulta imposible conocer de antemano qué pareja va a ser la adecuada. Por lo tanto, **en situaciones similares se calculan las respuestas una vez se ha generado la interfaz y todos sus elementos están colocados en pantalla**.

Consecuentemente a lo expuesto con anterioridad, **la primera acción realizada cuando se escoge una actividad es leer los parámetros** para poder cargar el layout adecuado con aquellos elementos necesarios.

Por su parte, cada una de estas actividades tiene un flujo parecido:

- **Leer los parámetros e inicializar variables:** la etapa ya comentada. Se leen los parámetros asociados a la actividad y se inicializan las variables y objetos utilizados en el ciclo de vida.
- **Generar el layout:** se cargan los distintos Views y ViewGroups del ejercicio, junto con las imágenes especificadas en la etapa anterior.
- **Obtener información sobre los elementos en pantalla:** como por ejemplo, coordenadas que servirán para la corrección.

Una vez comentados los aspectos comunes a la mayoría de casos desarrollados, se hace **un repaso del contenido especialmente creado para este Proyecto**. Los tipos que se pueden consultar a continuación constituyen las plantillas mencionadas, cuyo contenido puede ser personalizado en función de las necesidades existentes.

7.4.1. Columnas

Se dispone **una serie de imágenes en el centro de la pantalla**, mientras que a izquierda y derecha cuenta con dos columnas, cada una con un título. El objetivo es arrastrar las primeras a un lado u otro en función de este último o del criterio especificado en la descripción.



Figura 7.6: Actividad de las columnas.

Por ejemplo, “Antes de comer” y “Después de comer”, o “¿Qué debo hacer?” y “¿Qué no debo hacer?”.

7.4.2. Arrastrar al contenedor

Aparecen imágenes en la parte superior de la pantalla, mientras que debajo se ubican sus contenedores, destino final de las primeras y lugar al que los niños las tendrán que arrastrar.



Figura 7.7: Actividad de arrastrar a los contenedores inferiores.



Figura 7.8: Actividad de arrastrar a los contenedores inferiores.

Este ejercicio se puede aplicar, por ejemplo, para que se ordenen las comidas del día o para ubicar muebles en una estancia determinada.

7.4.3. Semillas

Parecido al anterior, pero con el objetivo de arrastrar semillas a macetas del mismo color.

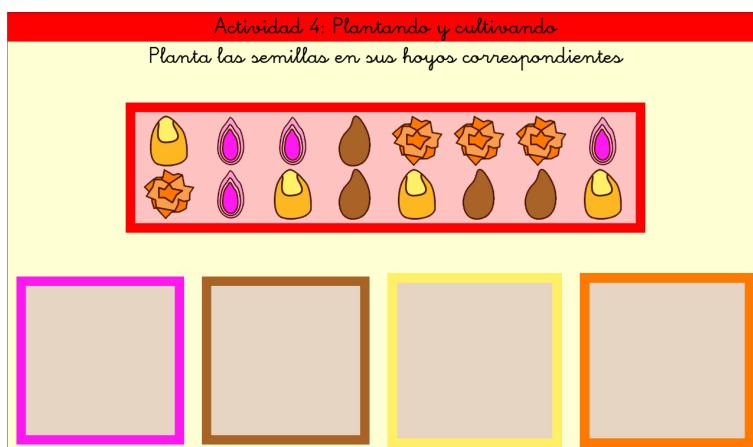


Figura 7.9: Actividad de las semillas.

7.4.4. Pirámide de alimentación

A la izquierda de la pantalla se ubica **una pirámide alimenticia vacía**, mientras que a la derecha lo hacen **distintas imágenes de comida**. El niño debe colocar cada alimento en su lugar correspondiente.

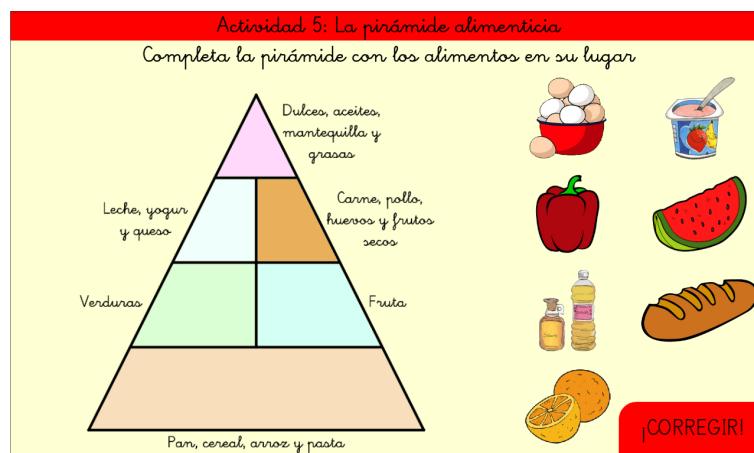


Figura 7.10: Actividad de la pirámide alimenticia.

7.4.5. ¡Los ojos de Zowi!

La habitación se ha quedado a oscuras, pero afortunadamente, Zowi puede “ver en la oscuridad”. El objetivo es que el niño desplace el dedo por la pantalla para encontrar elementos y poder identificar aquellos correspondientes al tema.



Figura 7.11: Actividad de los ojos de Zowi.

7.4.6. Operaciones matemáticas

Esta actividad cuenta con dos plantillas diferentes:

Interfaz simple

La pantalla se divide verticalmente en dos secciones, ubicándose a la izquierda una imagen temática y a la derecha 6 sumas y restas que el niño debe resolver.

Actividad 6: Cálculo

Resuelve las sumas y restas y consigue todos los dientes que faltan

8	-	0	=	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
1	-	0	=	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
5	-	4	=	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
2	+	4	=	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
1	-	0	=	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
7	+	1	=	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]

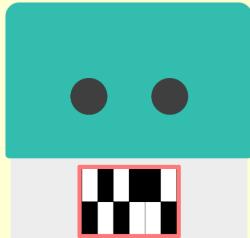


Figura 7.12: Actividad de de las operaciones (modo sencillo).

Interfaz compleja

A diferencia del caso anterior, las operaciones no muestran ni los operandos, ni el signo, ni el resultado. Por medio del botón “Enseñar”, **Zowi mostrará en su boca de LEDs toda la información necesaria para que los alumnos puedan resolver la cuenta.**

Actividad 6: Calcula

Resuelve las sumas y restas con los objetos de la casa

[Enseñar]	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
[Enseñar]	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
[Enseñar]	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
[Enseñar]	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
[Enseñar]	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]
[Enseñar]	<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>		<input style="width: 100%; height: 20px; border: 1px solid black;" type="text"/>	[Corregir]

Figura 7.13: Actividad de de las operaciones (modo difícil).

Además, mediante imágenes situadas inmediatamente después de los operandos, se simula la suma de objetos englobados en el tema. Por ejemplo, “2 sillas + 5 sillas = 7 sillas”.

7.4.7. Memory

Representación del conocido juego de encontrar la pareja, donde durante un intervalo de tiempo se pueden ver las imágenes, para a continuación darse la vuelta y tener que recordar su posición.

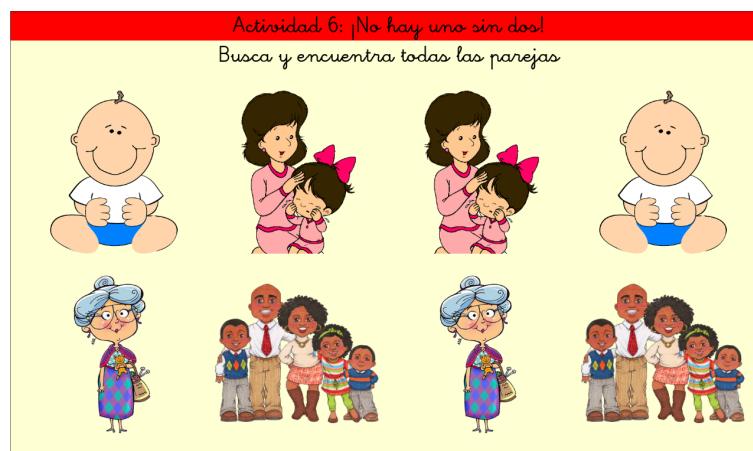


Figura 7.14: Actividad del Memory con las imágenes de cara.

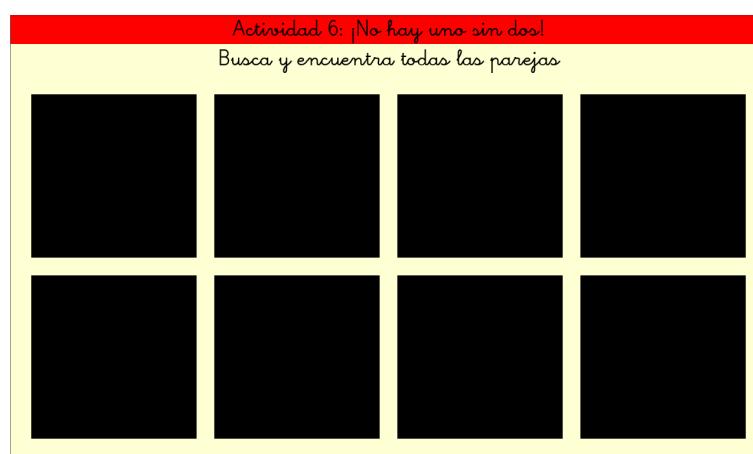


Figura 7.15: Actividad del Memory con las imágenes volteadas.

7.4.8. ¡Música, maestro!

Se muestran en pantalla **tres dictados musicales** y un botón asociado a cada uno de ellos. Al pulsarlo, Zowi comenzará a llevar el ritmo, teniendo en cuenta si los primeros contienen notas musicales blancas -2 tiempos-, negras -1 tiempo-, corcheas -medio tiempo- o silencios.

The image shows three musical dictation tracks in 2/4 time. Each track consists of a staff with vertical bar lines and a 2/4 time signature. The first track has two white notes (two beats). The second track has one black note (one beat) followed by one white note (one beat). The third track has one black note (one beat), one white note (one beat), and one black note (one beat). To the right of each track is a red button labeled "¡Comenzar!". The title at the top reads "Actividad 2: ¡Hora de ir al cole!" and the subtitle says "Toca el instrumento siguiendo el dictado musical".

Figura 7.16: Actividad de los dictados musicales.

7.4.9. Puzzle

En pantalla aparece una imagen dividida en 5 pedazos, siendo el objetivo arrastrarlos por pantalla al hueco central para **formar el puzzle**.

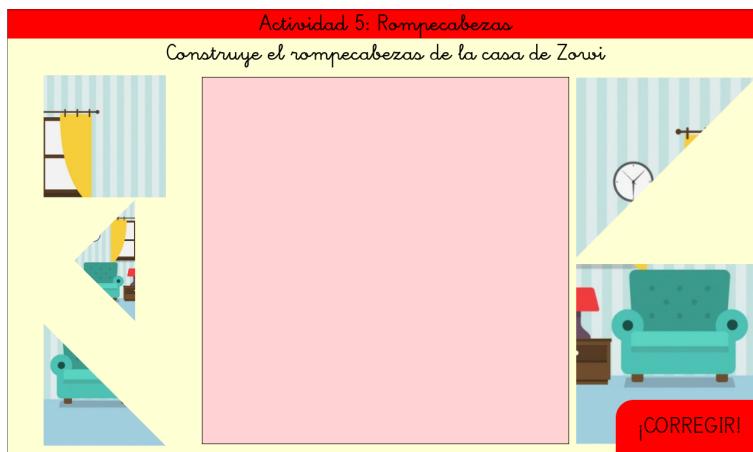


Figura 7.17: Actividad del puzzle.

7.4.10. ¡Guía a Zowi a su casa!

Aparecen varias imágenes de casas -con sus gemelas impresas en papel-, destacándose una por encima del resto. El objetivo es **guiar a Zowi con la mano, el pie o algún objeto**, utilizando el sensor de ultrasonidos; es decir, si se sitúa un objeto delante de los ojos, el robot camina siguiéndolo.



Figura 7.18: Actividad en la que se guía a Zowi a su casa.

7.4.11. Cuadrícula de colores

Zowi se representa en una cuadrícula de 4x4 con caminos de colores, donde su destino se sitúa en otra posición. La labor de los alumnos es calcular los cuadros que los separan e **identificar el camino más corto**.

Las imágenes, utilizadas para exemplificar la aleatoriedad de contenido, se pueden en la Figuras 7.2 y 7.3

7.4.12. Cuadrícula

La actividad para la que es necesaria una ampliación de hardware. Al igual que en la anterior, Zowi y su destino se ubican en dos celdas distintas, pudiendo existir obstáculos entre ellos.

El objetivo de los niños es llevar a Zowi a su destino sorteando aquello que bloquee su paso.

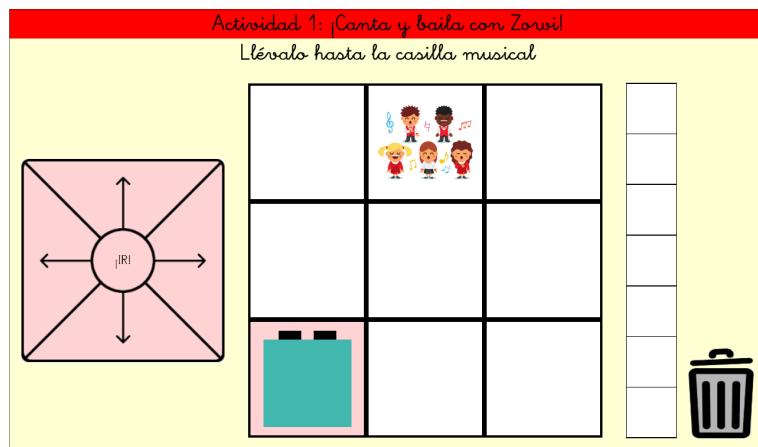


Figura 7.19: Actividad de la cuadrícula.

7.4.13. Ventanas emergentes

En el uso de ZowiApprende, los usuarios pueden encontrar **varias ventanas emergentes cuyo objetivo es resaltar cierta información**: por ejemplo, la búsqueda Bluetooth de Zowi, o el feedback positivo al completar con éxito una actividad.

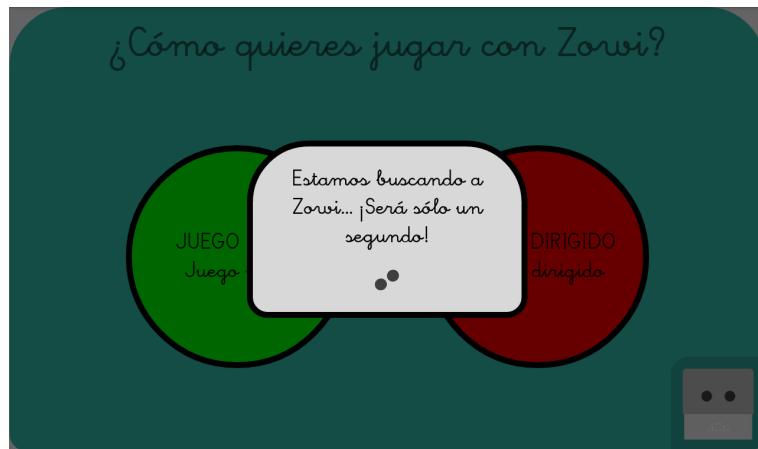


Figura 7.20: *Progress Dialog* que se muestra al buscar a Zowi usando el Bluetooth.



Figura 7.21: *Alert Dialog* que se muestra cuando no se puede conectar con Zowi.



Figura 7.22: Feedback recibido al completar con éxito una actividad.

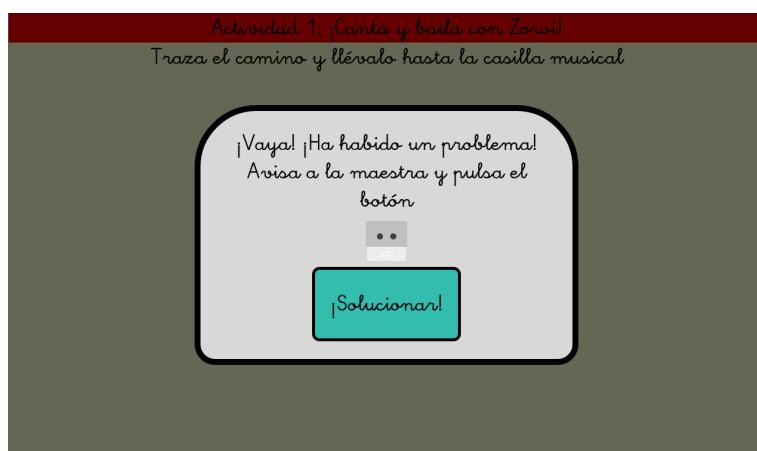


Figura 7.23: Ventana mostrada en caso de excepción por *NullPointerException*.

Capítulo 8

Ampliaciones del software de Zowi

Ya sea basándose en el código existente o creándolo de cero, **se ha dotado a Zowi de nuevas funciones que le permiten adecuarse a las actividades creadas**. Este capítulo trata estas modificaciones, centrándose en aspectos conceptuales y de software.

8.1. *setup* y *loop* de Arduino

El código existente en estas estructuras se ha reducido considerablemente, ya que **gran parte de las líneas no eran necesarias en esta nueva versión**.

Se contemplaban, por ejemplo, las opciones de pulsar los botones A, B, o ambos simultáneamente. Se lanzaban así las secciones del programa que posibilitaban la detección de obstáculos, el baile con pasos aleatorios o la detección de sonidos, respectivamente

Esta funcionalidad ya no es necesaria, ya que los niños van a interactuar directamente con la aplicación -salvo ocasiones puntuales, en las que el uso de botones forma parte de la actividad y no del bucle principal-. Por lo tanto, se ha eliminado todo y **se ha dejado sólo la parte encargada del control de la lectura del puerto serie**.

Esta última, por su parte, también se ha personalizado. Anteriormente, en el *loop* se leía el comando y se ejecutaba la función mapeada para el mismo. Actualmente, por el contrario, **se ha trasladado código de las funciones al bucle**.

El motivo es que en algunos casos, debido a la necesidad de interacción por parte de los niños o a delays, **la ejecución del programa se congela en un punto concreto durante algunos segundos**. A efectos prácticos, este comportamiento interno no es incorrecto, pero no entra dentro de las buenas pautas de la programación con Arduino.

El bucle principal debería ejecutarse continuamente, entrando en determinados bloques si se cumple una determinada condición, en lugar de esperar estáticamente a que esta ocurra. De este modo, aquellas actividades cuyo código bloquea el hilo de ejecución se han trasladado al *loop*, pudiendo permanecer el resto en métodos independientes.

8.2. Bocas, gestos, movimientos y bailes

Muchas características de fábrica se incluían, principalmente, para demostrar las posibilidades y el potencial del robot: numerosos tipos de baile, de caras para la matriz de LEDs, de gestos... En la práctica, no todos ellos eran utilizados, por lo que se optó por eliminar aquellos que no se adecuaban al contexto de ZowiApprende;

ocupaban memoria de forma innecesaria y se prefirió evitar el acumulamiento de código no utilizado.

8.3. Funciones para corregir actividades

Una de las acciones recurrentes en el uso de la nueva aplicación es **corregir ejercicios en función de la respuesta de los niños**.

Para ello, se le ha dotado de tres posibles alternativas, cada una de las cuales puede desembocar en un resultado positivo o negativo. A tal fin, se han creado tres comandos identificados con las letras “C”, “D” y “E”, cuyos parámetros son siempre -salvo en el último caso- “1” ó “0” dependiendo de si la respuesta es válida o no.

C 1 C 2
D 1 D 2
 E

Este tipo de respuestas consiste en una **cadena de acciones, entre las que se encuentran emitir sonidos, mover los servos y pintar bocas**.

De este modo, y empatizando con Zowi, los niños podrán saber cuándo su elección es correcta dependiendo del comportamiento del mismo: se mostrará contento en un caso, y triste y apático en otro.

- **C 1 y C 2:** reúne sonidos, movimientos y bocas.
- **D 1 y D 2:** combina sonidos con un tic o una exis, respectivamente.
- **E:** este comando **simula que es Zowi el que corrige los ejercicios**, en lugar de simplemente reaccionar a ellos. Cuando se envía desde la aplicación, el robot queda a la espera de que el niño acerque la tablet a sus ojos; en el instante en que detecte un objeto a menos distancia que el límite establecido, se envía un ACK de vuelta que el código de Android registra para, ahora sí, corregir el ejercicio y ejecutar alguno de los comandos anteriores.

8.4. Guiar a Zowi a su casa mediante el sensor de ultrasondos

Como se ha comentado anteriormente, una de las actividades consiste en guiar al robot hasta su casa. Para hacerlo, se ha creado **una función que hace que este camine dependiendo de la distancia existente entre un objeto y sus ojos**.

De esta forma, si el niño se pone de pie con la pierna delante de Zowi, este último caminará detrás del primero. La velocidad se define según la siguiente tabla:

Distancia (cm)	Velocidad
$d < 10$	Lenta
$10 < d < 20$	Media
$20 < d < 30$	Alta
$d > 30$	Inmóvil

Tabla 8.1: Velocidades en función de la distancia al guiar a Zowi usando el sensor de ultrasonidos.

Es importante destacar que **este método sólo permite el desplazamiento lineal**, dependiendo del usuario girarlo en caso de ser necesario.

Por otra parte, no necesariamente se debe usar la pierna para conseguir llegar al destino, sino que se podrá utilizar aquello que la maestra considere oportuno.

El bucle que mide la distancia en cada iteración se detendrá cuando se pulse el botón A, por lo que sólo debería ser pulsado al llegar al destino.

8.5. Sonar

Para una de las actividades, **Zowi debe simular un sonar**. Es decir, localización por medio de sonidos.

Concretamente, a medida que el niño desplace el dedo por la pantalla, el robot debe emitir pitidos en una determinada frecuencia u otra en función de la cercanía de una imagen. Para ello, recibe un comando cuyo parámetro indica la agudeza del pitido, siendo mayor cuando menor es la distancia.

S

Distancia	Tono
Escasa	Agudo
Media	Medio
Elevada	Grave

Tabla 8.2: Tonalidad del pitido en función de la distancia del dedo a una imagen.

Además, la frecuencia es otro parámetro cuyo valor se ve modificado, al ser inversamente proporcional a la distancia: será mayor -pitido más rápido- cuanto menor sea la distancia.

8.6. Ritmo de la música

Una de las actividades se centra en los **dictados musicales**, donde los niños se familiarizan con los conceptos del ritmo y de los tiempos.

En este contexto, **Zowi debe ser el director de orquesta, adecuando el movimiento de los pies a las notas del dictado**: negra, blanca, silencio o corcheas. La clase, por su parte, debe acompañarlo e imitar su comportamiento.

Para ello, **parte de un período base de movimiento de 2s**, siendo esta cifra lo suficientemente alta como para que se distingan todos los pasos. Desde Android, se

pueden enviar los siguientes argumentos: 0 -silencio-, 0.5 -corchea-, 1 -negra- ó 2 -blanca-. Multiplicando ambos operadores, **se obtiene el período final**, de modo que dependiendo de lo que reciba Zowi, ejecutará los movimientos más rápido o más despacio.

M 0 M 0.5 M 1 M 2

8.7. Girar y caminar recto

Funcionalidad presente en la **actividad de bloques lógicos**, donde Zowi tiene en cada uno de sus extremos una imagen.

El alumno debe escoger cuál es la correcta y **guiar a Zowi, haciéndolo girar -en caso de ser necesario- y enviándolo a continuación a “coger” la figura**.

En el primer caso, el comando consiste en el carácter “T”, acompañado de un parámetro numérico -1 ó 0- que indica el sentido de giro.

T 1 T 0

El segundo, por su parte, no recibe parámetros y el robot siempre caminará 3 pasos.

W

8.8. Operaciones en la boca

En una de las actividades de operaciones matemáticas, el pequeño protagonista es el encargado de **mostrar tanto los operandos (números del 1 al 9), como el signo (+ o -)**. Tras valorar alternativas como emitir tantos pitidos como fueran necesarios o pintar toda la información, de dígito en dígito, en la boca, se optó por una opción más sofisticada.

Se ha creado una nueva función, mediante la cual **Zowi muestra un determinado texto -en este caso, números y signos- de lado a lado**: el comienzo de la cadena aparece por la derecha, se desplaza por la matriz de LEDs y desaparece por la izquierda.

La consecución de esta característica se basó tanto en la comprensión del funcionamiento por defecto, como en el uso de los operadores lógicos AND y OR, y del bitshift.

El primer requisito indispensable es que la aplicación Android envíe la información en el formato correcto, consistiendo en la siguiente cadena:

O 10101 11111 00110 01010 10001

El comando, como se puede comprobar, es el carácter “O”. Los argumentos, por su parte, son conjuntos de 5 cifras que representan los bits que debe mostrar la columna derecha de la matriz.

En el ejemplo mostrado más arriba, los pasos llevados a cabo serían los siguientes, “entrando” cada una de ellas por la derecha y desplazando aquellos bits que tenga a su izquierda:

000001	000011	000110	001100	011001
000000	000001	000010	000101	001000
000001	000011	000111	001110	011100
000000	000001	000011	000111	001110
000001	000011	000110	001100	011001

Se puede ver cómo la columna va desplazándose, mostrando los bits enviados en el instante actual y los enviados previamente. De este modo, y teniendo en cuenta que el contenido de la lista de caracteres se controla desde la aplicación Android, **es posible personalizar las letras del mensaje.**

A nivel de código, el algoritmo consiste en una serie de operaciones realizadas dentro de un bucle, donde cada elemento constituye una de las columnas coloreadas anteriormente -o uno de los argumentos-. El mayor problema reside en que la cadena de 0 y 1 de la **variable tipo long** los representa de izquierda a derecha, de arriba a abajo; es decir, los bits recibidos de la aplicación no se ubican uno al lado del otro, por lo que una concatenación directa no es plausible.

O 10101 11111 00110 01010 10001

↓

000000-000000-000000-000000-010101

000001 000000 000001 000000 000001

Para colocar cada bit en la posición correspondiente, se optó por combinar la operación lógica OR con el bitshift. Así, **se parte de un long con valor 0** -con nombre matrixCode-, cuyos dos bits altos son irrelevantes en este contexto.

000

Una variable auxiliar, por su parte, almacena la **información correspondiente a cada bit recibido**. En el caso del primero -1- del primer argumento -10101, coloreado de azul-:

La cadena de la izquierda corresponde al contenido de la **variable auxiliar recién leída la información**, mientras que la de la derecha representa esa misma variable, habiendo **aplicado un *left bitshift de 24***. De este modo, se desplaza el “1” 24 posiciones a su izquierda, colocándolo en aquella que corresponde a su ubicación en la columna.

En este punto, el resultado de una operación OR entre la variable auxiliar -con el contenido de interés- y matrixCode resulta en que la segunda también alberga la información necesaria.

De forma análoga, se leen los siguientes bits del argumento, aplicando un bitshift 6 veces menor en cada iteración, para posteriormente realizar un operación OR con matrixCode. Así, una vez leídos todos ellos, esta última contiene los datos correspondientes a la columna completa.

En lo que respecta al desplazamiento lateral, consta de dos etapas:

- **Desplazamiento lateral:** una operación bitshift con un 1 permite conseguir el resultado deseado y **desplazar todas las columnas una unidad hacia la izquierda**, pero en sucesivas iteraciones el nuevo “hueco” puede tener contenido. Por lo tanto, se requiere un paso adicional.
- **Operación AND:** se lleva a cabo una **operación AND con el long**:

0011110111110111110111110111110

Eliminando así todo el contenido de la última columna, para que el resultado de la operación OR en matrixCode sea la que determine su valor.

Para acabar, se muestra el fragmento de código descrito para complementar la explicación.

```

1 unsigned long row;
2 int bitshift;
3
4 char *arg = SCmd.next();
5 while (arg != NULL) {
6     bitshift = 24;
7     for (int i=0; i<5; i++) {
8         row = arg[i] - '0';
9         matrixCode = matrixCode | row << bitshift;
10        bitshift = bitshift - 6;
11    }
12    zowi.putMouth(matrixCode, false);
13    delay(500);
14
15    matrixCode = matrixCode << 1 & REMOVE_RIGHT_BITS;
16    zowi.putMouth(matrixCode, false);
17
18    arg = SCmd.next();
19 }
```

8.9. Tabla resumen de comandos

Comando	Argumentos	Descripción
C	1 ó 0	Zowi corrige la actividad con un gesto
D	1 ó 0	Zowi corrige la actividad con un tic o exis en la boca
E	-	Zowi espera que se le acerque un objeto para corregir
G	-	Zowi camina detrás de un objeto situado delante de los ojos
S	1, 2 ó 3	Zowi emite un sonido como un sonar
M	0, 0.5, 1 ó 2	Zowi lleva el ritmo de la música
T	1 ó 0	Zowi gira 90º a izquierda o derecha
W	-	Zowi camina recto 3 pasos
O	long binario	Zowi pinta números y signos en su boca

Tabla 8.3: Nuevos comandos disponibles.

8.10. Nuevo mecanismo para caminar

Zowi no camina bien. Este es un hecho que desemboca en que, en el momento en que se requiera cierta precisión, **la ubicación final del robot no pueda ser determinada con exactitud**. Por lo tanto, desde un primer momento se buscó una solución que pudiera mejorar este comportamiento.

Para empezar, cabe destacar los dos motivos principales por los que los pasos no son precisos:

- **Zowi resbala en el suelo:** el robot cuenta con unos **pies de plástico** con un área elevada, que le aportan mucha estabilidad. No obstante, y sobre todo en superficies con poco rozamiento, **estos resbalan e impiden que el paso sea preciso**.

A pesar de haber hecho pruebas sobre materiales con adherencia, los resultados no fueron exitosos. En una etapa del paso, Zowi desliza el pie sobre lo que tenga debajo, y superficies con más rozamiento impiden que este movimiento se complete debidamente.

- **Los servos no son precisos:** como se ha explicado en la sección sobre el software base, los servos giran constantemente entre los 60º y los 120º. Sin embargo, motivado por la aproximación a la hora de definir el ángulo -sólo se admiten números enteros- y por la imprecisión existente de fábrica, **el hecho de definir un determinado giro en repetidas ocasiones no implica que la posición del servo acabe siendo exactamente la misma**.

Este suceso es comprensible, sobre todo teniendo en cuenta **el precio del robot en su conjunto, y de los componentes individualmente**; no son lo mejor del mercado, y se nota.

La conclusión es que a los pocos pasos, Zowi empieza a laderase, lo que impide conocer su posición a medio plazo. Se han barajado, por tanto, varias alternativas cuyo concepto es muy similar.

La idea consiste en **encontrar un mecanismo que recopile información a medida que el robot camina**. De este modo, y con el feedback obtenido sobre la trayectoria, se podría **cambiar la dirección en función de las necesidades**. Este inconveniente viene motivado por el uso de la **cuadrícula**, en la que Zowi debe llegar de una celda a otra con un mínimo de precisión.

- **Guía de electroimanes:** el objetivo es situar una **guía que indique cuál debe ser la trayectoria del robot**.
- **IMU:** se pretende **tener una medida de referencia de la dirección inicial**, para posteriormente **medir las variaciones con respecto a ella al caminar y compensarlas** con un nuevo algoritmo.
- **Siguelíneas:** el concepto de un siguelíneas es autoexplicativo, pero se intentará **encontrar una estructura y realización que evite los cables en el exterior de Zowi**.

8.10.1. Guía de electroimanes

La primera opción barajada consiste en **ubicar debajo de la cuadrícula una guía de imanes, junto con otros dos situados en el interior de los pies de Zowi**. De este modo, al dar pasos **se contaría siempre con una referencia en el suelo**, evitando que el protagonista se desplazara por otros espacios de la superficie donde no hubiera piezas metálicas.

El primer impedimento consistió en la atracción constante por parte de los imanes, que impedía que Zowi levantara los pies del suelo una vez se hubiera efectuado la imantación. Se intentó encontrar un punto intermedio en el que esta fuera lo suficientemente fuerte como para atraer sus pies, pero lo suficientemente débil como para permitir que siguieran su recorrido en el siguiente paso. No obstante, *no se tuvo éxito en esta tarea*.

La segunda versión de esta idea pasó por la contemplación del **uso de electroimanes**. Así, se podría generar una corriente eléctrica cuando se tuviera que ejecutar la atracción, extinguiéndola en caso contrario. Se hizo un estudio barajando posibles candidatos teniendo en cuenta la potencia, el peso y el consumo de corriente, pero **no fue posible encontrar un dispositivo que reuniera todas las características necesarias** para este proyecto: pequeño tamaño, potencia considerable, alimentación -a ser posible- por medio de una pila...

El desenlace fue el mismo que en la primera versión, **resultando imposible encontrar un bobina de cable cuya longitud, número de vueltas y tamaño se adecuaran a las necesidades existentes**.

8.10.2. Uso de acelerómetro y giroscopio

Sin olvidar que la placa base de Zowi es un Arduino, **existe la posibilidad de incorporar sensores nuevos**. En este caso, un MPU 6050, un pequeño componente electrónico

que incorpora un acelerómetro y un giroscopio. Dado que la investigación realizada al respecto conllevó un gran número de horas, **se van a explicar detalladamente tanto el fundamento teórico como las pruebas realizadas.**

El acelerómetro

El acelerómetro es un sensor que mide la aceleración -cambios de velocidad- con respecto a la superficie terrestre. De esta forma, es posible conocer en todo momento la orientación del dispositivo en el que esté integrado, tomando como referencia la aceleración de la gravedad -aproximadamente 9.8 m/s^2 .

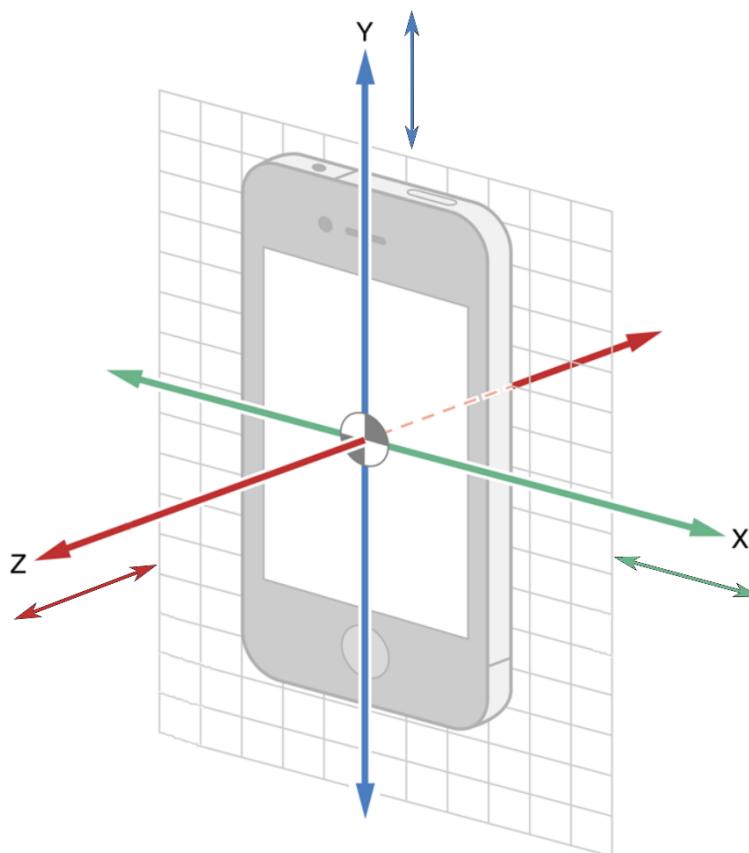


Figura 8.1: Esquema de las mediciones de un acelerómetro.

Como se ejemplifica en la Figura 8.1, **este sensor detecta variaciones lineales**, en contraposición al giroscopio, que hace lo propio con cambios angulares.

La aceleración, según su definición física, **es la variación de la velocidad con respecto al tiempo**. De este modo, es posible calcular la misma a partir de estos dos últimos parámetros, respondiendo la aceleración media a la siguiente ecuación:

$$\vec{a} = \frac{\Delta \vec{v}}{\Delta t} \quad (8.1)$$

Sin embargo, **Newton enunció en el siglo XVII su segunda ley, o Ley de la Fuerza**. En ella, **relaciona la masa de un objeto con la fuerza y la aceleración que actúan sobre él**, según la siguiente ecuación:

$$\vec{F} = m\vec{a} \longrightarrow \vec{a} = \frac{\vec{F}}{m} \quad (8.2)$$

De esta forma, es posible **determinar la aceleración de un cuerpo prescindiendo de las variables velocidad y tiempo**, y es tal y como lo hacen la mayoría de sensores de este tipo disponibles en el mercado.

Para comprender el funcionamiento de este componente, resultan muy esclarecedores los datos devueltos por el mismo cuando este se sitúa encima de una superficie plana.

Según la **Tercera Ley de Newton, o Principio de Acción-Reacción**, si un objeto A ejerce una fuerza sobre un objeto B, B ejerce una fuerza sobre A con igual módulo y dirección, pero de sentido contrario. Por lo tanto, y teniendo en cuenta la ubicación del sensor y que está siendo sometido a la fuerza de la gravedad, **existe una fuerza de igual módulo en sentido contrario en la dirección del eje Z: 9.8m/s², o 1g, la aceleración de la gravedad**.

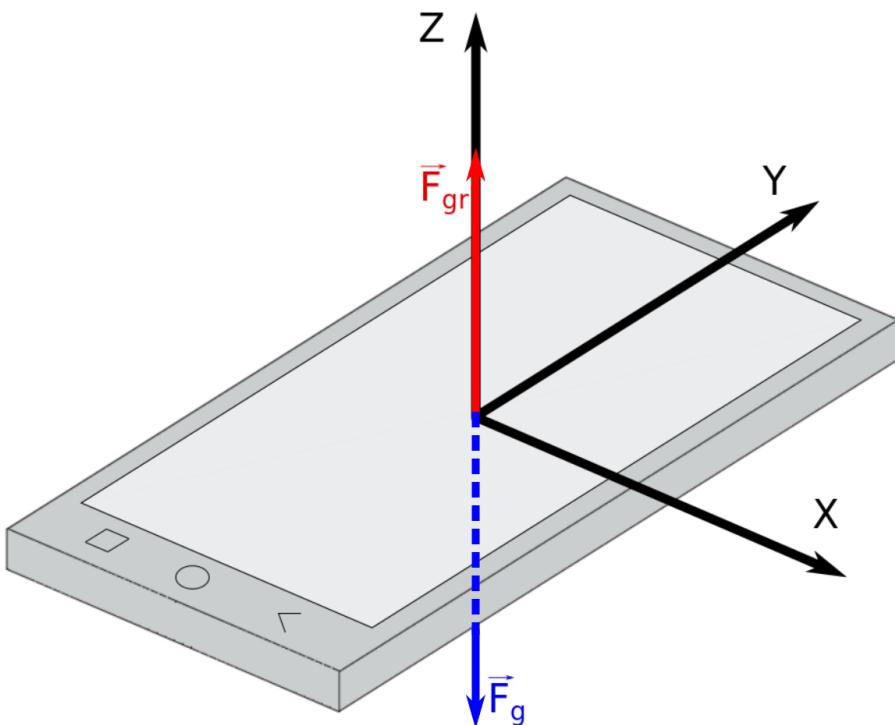


Figura 8.2: Direcciones y sentidos del acelerómetro en un smartphone (los ejes y su sentido podrían cambiar).

	\mathbf{a} (m/s ²)	\mathbf{a} (g)
Eje X	0.153	0.0156
Eje Y	0.0	0.0
Eje Z	9.807	1.00004
Módulo	9,80819	1.0001

Tabla 8.4: Valores devueltos por el acelerómetro en reposo sobre una mesa, atendiendo a los ejes de la Figura 8.2.

Esta fuerza normal coincide en sentido con el semieje positivo de las Z, motivo por el que el valor devuelto por el acelerómetro es mayor que 0; situar el móvil boca abajo implicaría la cifra opuesta.

En este caso, toda la fuerza a la que está sometida el dispositivo recae sobre un solo eje; sin embargo, **inclinar el dispositivo en distintos ángulos tiene repercusión en los valores devueltos por el sensor** (prueba de ello es la gran cantidad de juegos que aprovechan su funcionalidad). Por ejemplo, podría estar inclinado con un determinado ángulo con respecto a la superficie de una mesa, devolviendo los valores de la Tabla 8.5.

	a (m/s ²)	a (g)
Eje X	0.153	0.0156
Eje Y	2.2298	0.22738
Eje Z	9.5	0.96873
Módulo	9.75938	0.99518

Tabla 8.5: Valores devueltos por el acelerómetro inclinado con respecto a una superficie plana.

Si bien el acelerómetro es un dispositivo muy útil en numerosas circunstancias, **por sí sólo no es suficiente para definir cuánto se desvía Zowi mientras camina**. Su funcionamiento se basa siempre en una dirección absoluta; es decir, para hacer los cálculos siempre toma como referencia la dirección de la gravedad, que apunta al centro de la Tierra. Este hecho implica que, cuando rotamos el sensor con respecto al eje Z -atendiendo a la imagen 8.2-, el sensor no es capaz de determinar la orientación del mismo.

No sólo eso, sino que **los acelerómetros adolecen de mucho ruido en sus medidas**, aún estando en estado de reposo. Este hecho implica que es poco fiable hacer medidas precisas basándose sólo en un dispositivo de esta clase.

El giroscopio

Por lo tanto, de cara a conseguir el objetivo de esta sección, es necesario **complementar la información devuelta por el acelerómetro con la que nos proporciona el giroscopio**.

El giroscopio es un sensor que mide variaciones angulares; más concretamente, **cuánto varía el angulo a lo largo del tiempo**. De esta forma, es posible **detectar giros a lo largo de los 3 ejes**.

Igual que el caso anterior, la velocidad angular es un parámetro que se rige por una fórmula:

$$\omega = \frac{\Delta\theta}{\Delta t} \quad (8.3)$$

No obstante, y a pesar de servir para un objetivo común, hay una diferencia crucial entre un acelerómetro y un giroscopio: mientras **el primero toma como referencia la gravedad -una fuerza estática, siempre con la misma dirección, módulo y sentido-, el segundo se basa en medidas relativa**.

No tiene un parámetro global para tomar como referencia, sino que **para determinar el giro se debe multiplicar la velocidad angular medida por el tiempo empleado.**

$$v = \frac{e}{t} \rightarrow e = v * t \quad (8.4)$$

$$\Delta\omega = \frac{\Delta\theta}{\Delta t} \rightarrow \Delta\theta = \Delta\omega * \Delta t \rightarrow \theta = \omega * t \quad (8.5)$$

$$\theta_n = \theta_{n-1} + \Delta\theta = \theta_{n-1} + \omega * t \quad (8.6)$$

A efectos prácticos, esto implica que **si se calcula el tiempo girado por el valor medido, se puede obtener el ángulo.** Y aunque esta deducción es perfectamente correcta, existen dos factores determinantes en la precisión de los datos.

- **El tiempo es una medida continua:** aunque pueda parecer evidente, este hecho físico repercute directamente en la toma de datos. Para calcular un ángulo de giro es necesario tomar una muestra del tiempo, aquel que haya estado girando el dispositivo; sin embargo, esto no es más que **una aproximación, lejos de constituir una medida precisa.**

En esta operación se está aproximando una medida continua a un valor discreto, resultando en una pequeña incorrección que impide que el resultado final sea del todo fiable.

- **El error es acumulativo:** si bien los acelerómetros adolecen de error en la medida, cada una de ellas se compara con la misma referencia absoluta. En este caso, por el contrario, **la medida n se basa en la n-1, que a su vez se basa en la n-2, etc.**

Esto ocurre debido a que a la hora de calcular un ángulo, se necesita conocer la posición anterior para sumarle la multiplicación de la velocidad angular por el tiempo -Fórmula 8.6-. Si θ_{n-1} es una aproximación ligeramente imprecisa, θ_n acumula ese error, y así sucesivamente. Es lo que se conoce como *drift*.

Este efecto puede verse reflejado en la Figura 8.3.

Más aún, puede mostrarse cómo evolucionan los valores calculados a lo largo del tiempo manteniendo el sensor completamente inmóvil. De esta forma, el efecto puede corroborarse sin la influencia de factores externos.

Cabe destacar que la **gráfica inferior representa el valor medido por el giroscopio, mientras que la superior refleja el cálculo final del ángulo girado.**

8.10.3. MPU 6050

Aunque de forma independiente el acelerómetro y el giroscopio no consiguen una precisión excelente, **existen mecanismos que permiten combinar ambas mediciones para obtener cifras de orientación más exactas**, como la calibración.

El primer paso que se debe tener en cuenta -igual que en muchos otros sensores- es **la calibración**. Estos chips no recogen medidas exactas por defecto, dependiendo de muchos factores como el lugar geográfico en el que esté situado o el proceso de fabricación.

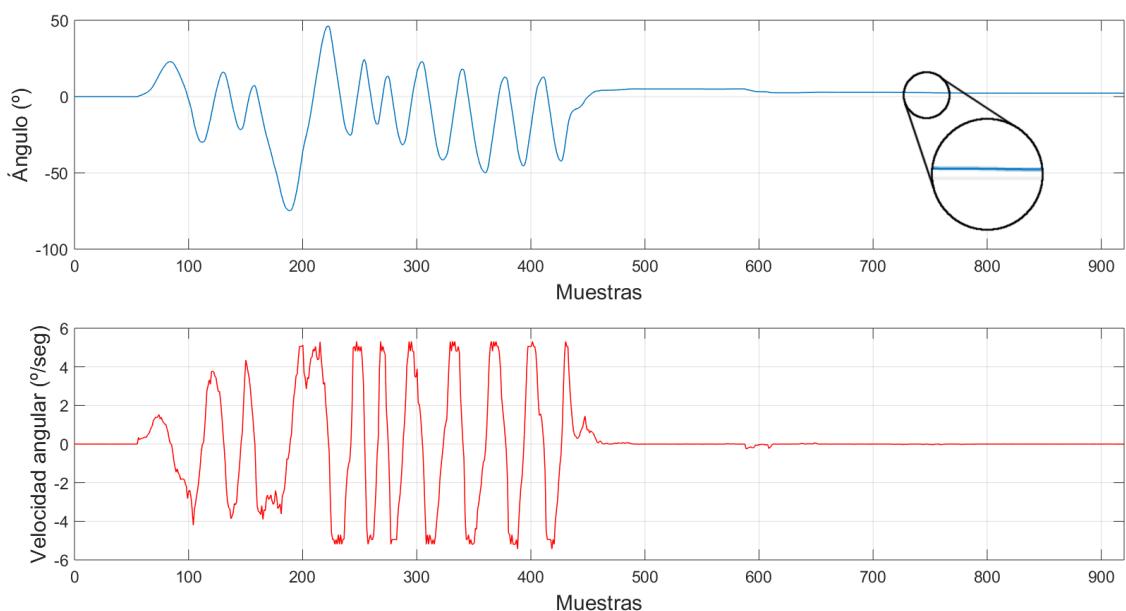


Figura 8.3: *Drift* en las medidas del giroscopio de un MPU6050.

A efectos prácticos, implica que **cuando el sensor debería registrar un determinado valor, lo devuelve con cierta imprecisión**. A continuación se puede ver las gráfica del giroscopio en reposo encima de una superficie plana.

Se puede ver como en la Figura 8.5, los valores correspondientes a la velocidad angular distan ligeramente de 0.

El caso del acelerómetro sería análogo, registrando datos con un pequeño error en los ejes X, Y y Z.

La calibración, mecanismo principal para evitar este inconveniente, **consiste en tomar una muestra representativa de valores, para a continuación hacer la media y obtener una cifra que se usará posteriormente en el cálculo de las medidas**. De esta forma, restando esta constante se consiguen gráficas perfectamente centradas en los ejes.

Se recuerda, además, el efecto del *drift* en el giroscopio mostrado en la Figura 8.3. Este desfase recurrente impide usar sólo este sensor para intervalos medios o largos de tiempo.

A pesar de todo, se puede observar que **la toma final de medidas consiste en una agrupación de muestras entorno al 0**. Este ruido -ligeras variaciones con respecto al eje- es considerado normal en esta clase de dispositivos y es prácticamente imposible de eliminar, pero **trabajando de forma conjunta se pueden combinar los datos de ambos sensores para que el resultado sea inmejorable**.

Para ello, es necesario el **uso de un algoritmo** que desempeñe este trabajo, conocido como filtro. Existen dos muy extendidos y cuyo uso está aconsejado en determinados contextos:

- **Filtro Kalman:** consiste en un algoritmo pesado que consigue eliminar el error de cada iteración, devolviendo el valor correcto. Es **utilizado frecuentemente en aeronáutica**, aunque cuenta con la desventaja de su elevado tiempo de cómputo, siendo así poco apto para este proyecto.
- **Filtro complementario:** consiste en filtro paso bajo y un filtro paso alto, cuyo fin

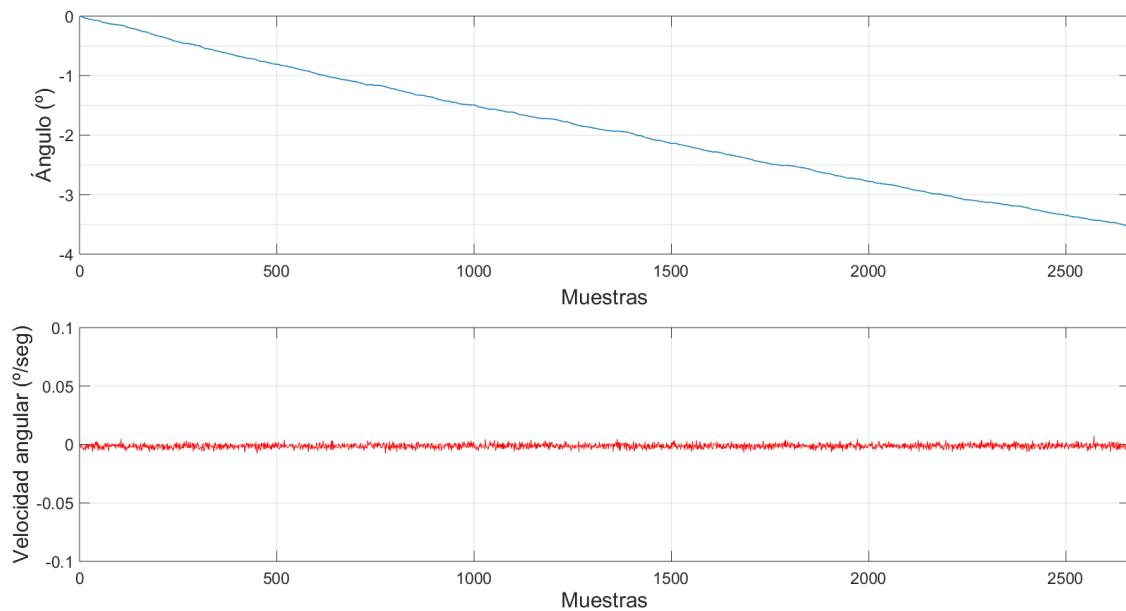


Figura 8.4: *Drift* en las medidas del giroscopio de un MPU6050 inmóvil en una superficie plana.

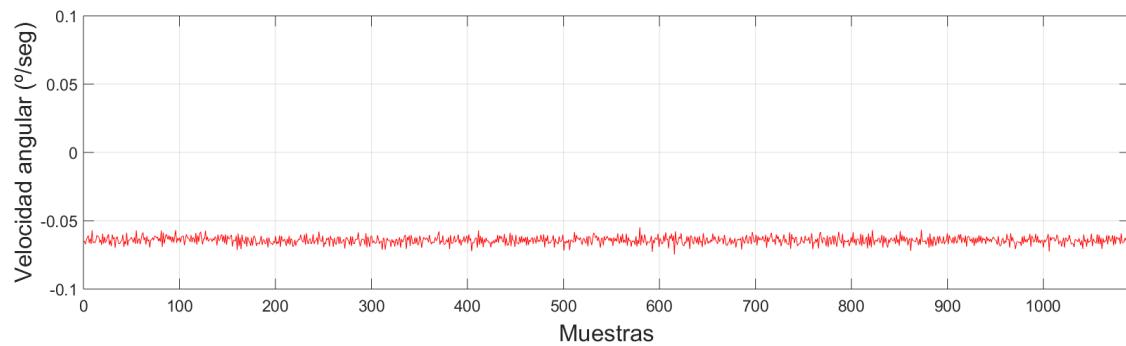


Figura 8.5: Valores devueltos por el giroscopio del MPU6050 sin calibrar.

es -valga la redundancia- **filtrar los valores del acelerómetro y giroscopio**, respectivamente. Se consigue así eliminar aquellas medidas extremas y obtener gráficas más suaves y precisas.

La fórmula es la siguiente:

$$\alpha_{filtrado} = a * \alpha_{giroscopio} + (1 - a) * \alpha_{acelerometro} \quad (8.7)$$

Donde “a” es una constante con valor 0.98, aunque puede ser modificada ligeramente para mejorar el rendimiento del conjunto.

La aplicación de la Fórmula 8.7 requiere, como se puede ver, del cálculo previo de los ángulos obtenidos mediante los dos sensores -aplicando trigonometría sobre los valores del acelerómetro y el fundamento ya explicado en el caso del giroscopio-.

Combinando los valores de los 3 ejes, es posible obtener la orientación espacial del sensor, identificando en todo momento qué es delante, detrás, izquierda o derecha. En el siguiente vídeo de YouTube (www.youtube.com/watch?v=qmd6CVrlHOM) se puede observar tanto esta afirmación, como las diferencias entre la toma de datos con un sensor, otro o ambos a la vez.

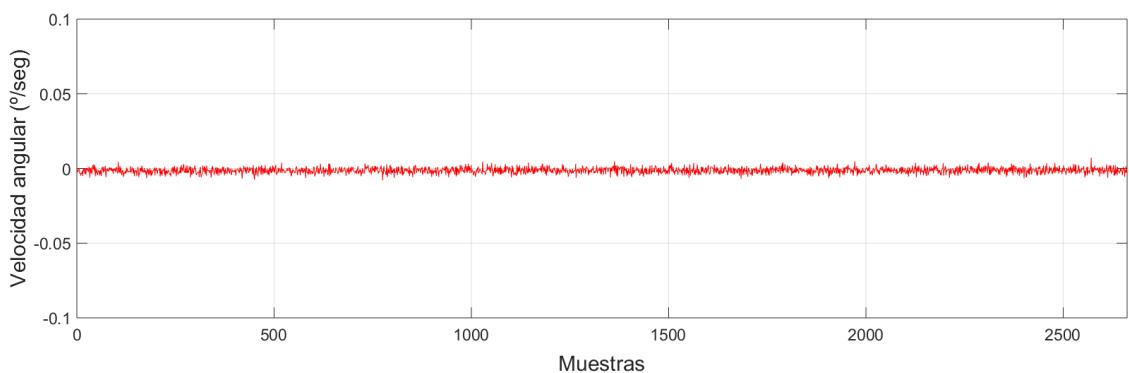


Figura 8.6: Datos devueltos por el giroscopio del MPU6050 calibrado.

Sin embargo, aunque la información recabada hasta el momento pueda parecer útil y más que suficiente, existe **un inconveniente que repercute directamente en el rendimiento del conjunto**. No se debe olvidar el objetivo de esta sección: **tomar una dirección como referencia, para a continuación medir y compensar los giros en torno al eje Z**; es decir, aquel con la misma dirección que la fuerza de la gravedad.

Este hecho es crucial, ya que -como se ha explicado en [El acelerómetro](#)- **el acelerómetro no es capaz de registrar este movimiento**: toma como referencia absoluta la dirección, el módulo y el sentido de esta fuerza, y **no es posible identificar ninguna diferencia cuando el vector que se toma como referencia actúa siempre de la misma forma sobre el objeto**.

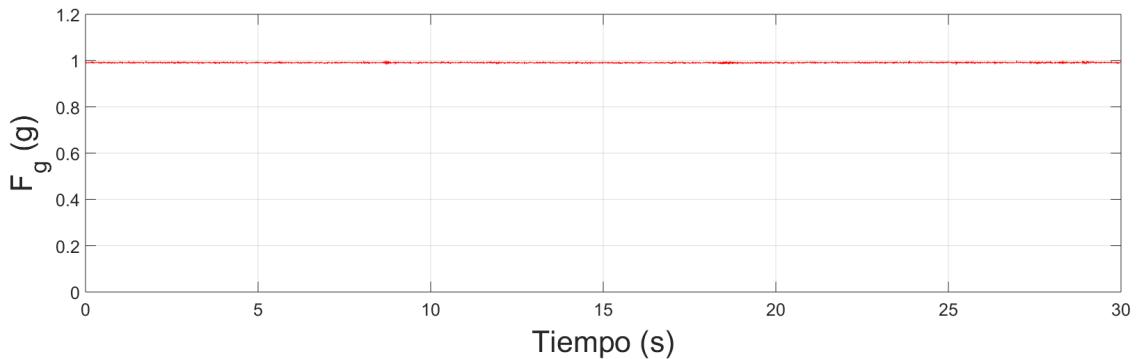


Figura 8.7: Datos devueltos por un acelerómetro al girar lentamente sobre el eje Z.

Por lo tanto, y a pesar de haber realizado pruebas al respecto, **el sistema adolece de los mismos defectos que un giroscopio convencional**.

Esta reflexión sirve como introducción al siguiente apartado de este escrito: la teoría y la práctica del magnetómetro.

8.10.4. MPU 9250: uso de acelerómetro, giroscopio y magnetómetro

El eje con respecto al cual se pretende medir es el más problemático de los 3, por lo que se requiere buscar una solución que permita identificar giros en el mismo. Complementando todo lo expuesto en las secciones anteriores, los datos medidos por el acelerómetro y el giroscopio se pueden combinar con aquellos devueltos por otro sensor: el magnetómetro.

El magnetómetro proporciona valores sobre la densidad de flujo magnético, permitiendo obtener información relevante relativa a la orientación del dispositivo. Es

comúnmente conocido como brújula, capaz de señalar el norte y otras direcciones a partir de los datos recabados.

Más concretamente, se obtiene una cifra por cada uno de los 3 ejes; **la fuerza magnética tiene una dirección, sentido y módulo determinados**, y es a este sensor lo que la fuerza de la gravedad al acelerómetro. Proporciona, por tanto, **una referencia absoluta frente a la que se pueden comparar las cifras obtenidas sin acumular el error**.

Calibración del magnetómetro, de crucial importancia

Si bien en los dos primeros componentes este paso era importante, en este apartado lo es más aún. **Los magnetómetros presentan dos tipos de distorsión**, que sin la corrección pertinente desembocan en valores completamente falseados.

Teniendo en cuenta que el módulo del campo magnético en un punto es constante, y la existencia de un vector con componentes en 3 ejes -una para X, Y y Z-, **la respuesta ideal de este sensor consiste en una esfera centrada en el origen de coordenadas**. Cambiar la orientación de este dispositivo electrónico implica que los valores de los ejes varían, manteniendo su módulo constante; por lo tanto, tomando suficientes muestras, el resultado final debería ser una esfera cuya superficie esté cubierta por los puntos obtenidos. El acelerómetro No obstante, **de fábrica este comportamiento ideal es inalcanzable**, por lo que se requiere de ciertos procedimientos antes de poder sacarle el máximo potencial al conjunto. Las dos distorsiones que ensucian el resultado final son las siguientes:

- **Distorsión de hierro duro:** se produce por la **imantación permanente de algunos componentes ferromagnéticos** del sensor. Al ser intrínsecos al propio material, son constantes y, por tanto, relativamente fáciles de corregir.

Provocan que el muestreo ideal de datos se desplace con respecto al centro, por lo que el procedimiento para corregirlo es parecido a los explicados con anterioridad: se toma un conjunto de datos lo suficientemente grande, se calcula su media y se resta sobre las medidas posteriores.

- **Distorsión de hierro blando:** son más difíciles de compensar ya que no son absolutas, sino que constituyen alteraciones variables en el campo electromagnético por parte de algunos elementos.

En el muestreo final provocan un escalado de la esfera, haciendo que se parezca a una elipse. La solución consiste en reescalar manualmente para que recupere su forma original.

Los efectos comentados se pueden corroborar en las Figuras 8.8 y 8.9. Ambas gráficas constituyen **proyecciones de la esfera previamente mencionada en alguno de los planos**, trasladando los puntos de un espacio tridimensional a una superficie. Aunque este paso no es estrictamente necesario para confirmar la existencia de las dos distorsiones, proporcionan más legibilidad en lo que a lectura de datos se refiere.

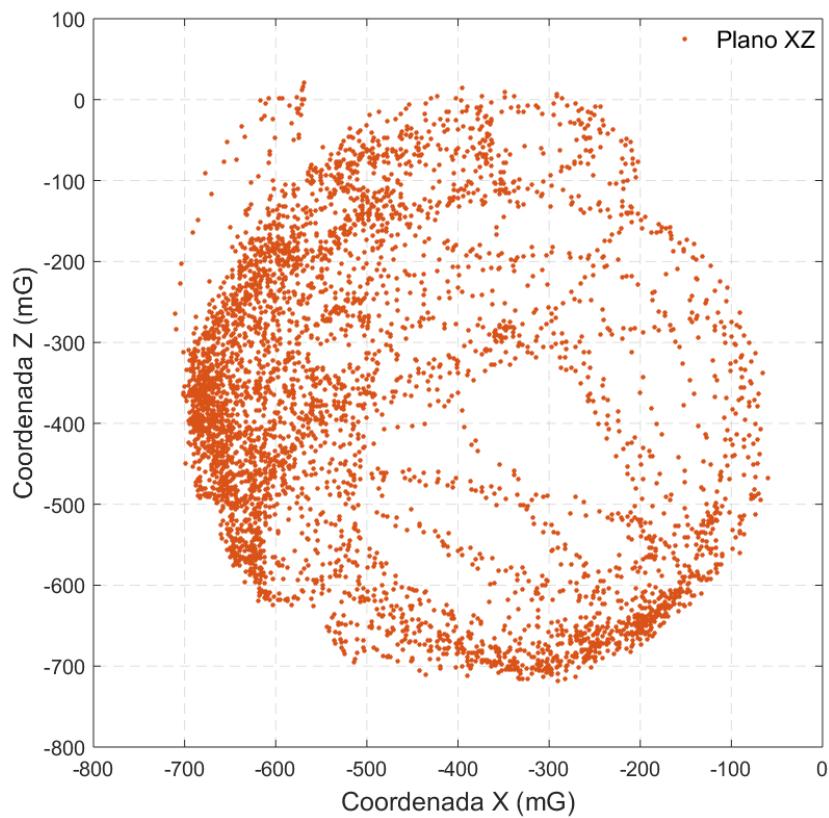


Figura 8.8: Valores obtenidos por el magnetómetro desplazados con respecto al (0,0).

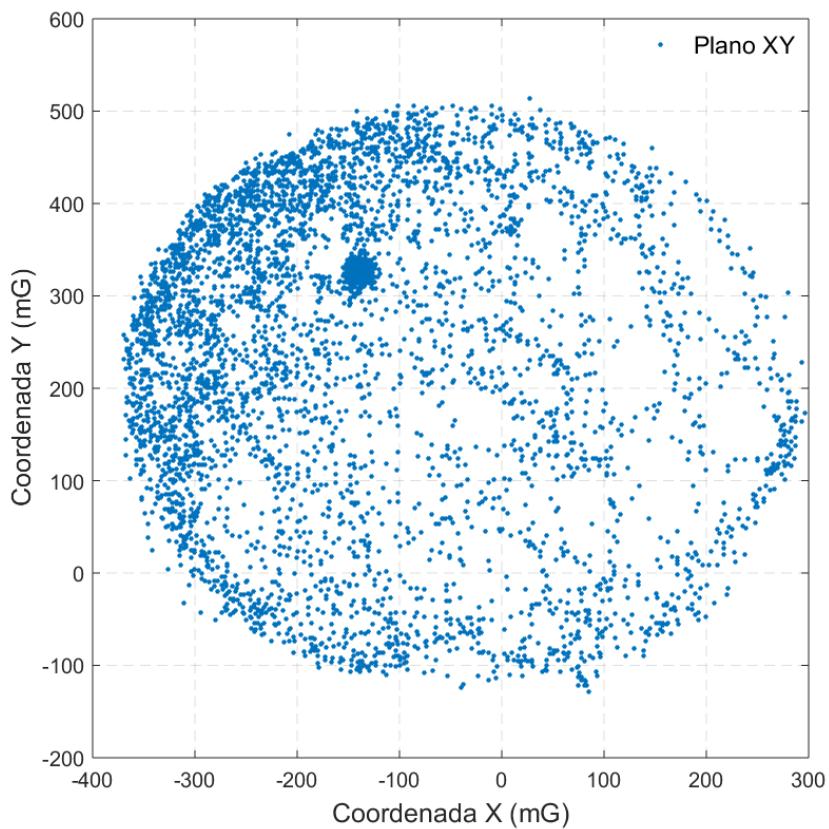


Figura 8.9: Valores obtenidos por el magnetómetro con forma ligeramente elipsoidal.

En la Figura 8.10, por su parte, se puede comprobar cómo **ninguno de los ejes presenta un comportamiento correcto de fábrica**, siendo la Figura 8.12 la corres-

pondiente a la esfera en cuya superficie se distribuyen las cifras del campo magnético.

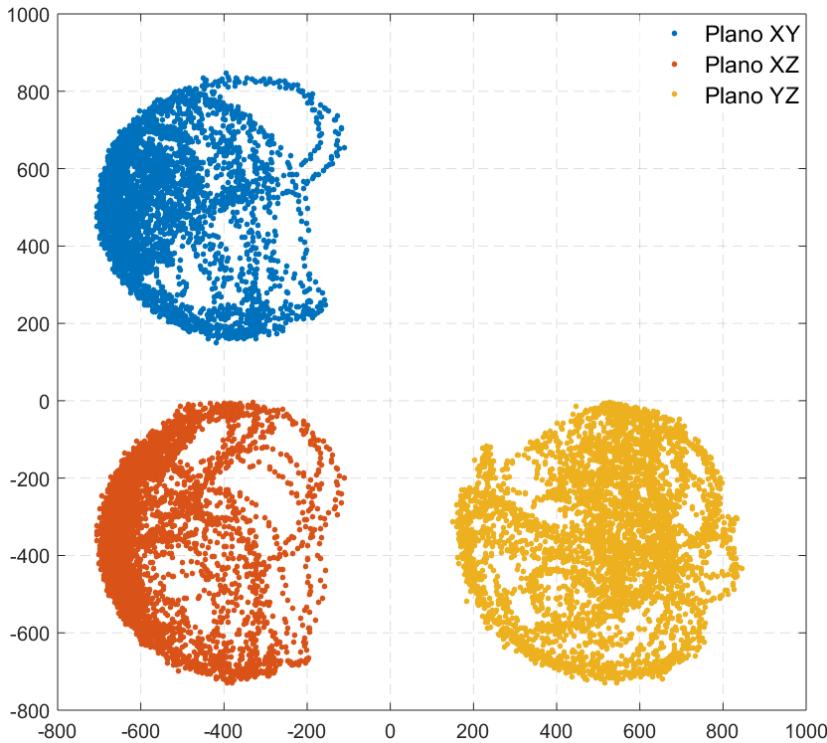


Figura 8.10: Valores obtenidos por el magnetómetro no calibrado proyectados en los 3 planos del sistema de coordenadas cartesianas.

El procedimiento de calibración, aunque considerablemente simple, ha implicado multitud de inconvenientes. En lo referente a la configuración del sensor, **fue problemática la puesta a punto del magnetómetro con la placa base de Zowi**, y la utilización de una unidad defectuosa derivó en mucho tiempo perdido intentando solucionar problemas irresolubles.

Una vez lograda la obtención de datos y la conversión correspondiente a unidades comprensibles (Gauss o Tesla), fue necesaria la propia calibración. Para ello, se requirió mover el robot -con el sensor instalado dentro- en forma de 8, en las 3 dimensiones posibles. Aunque el procedimiento no estuvo exento de fracasos, **se consiguió obtener unas cifras que adecuaban la esfera con bastante exactitud a su modelo ideal**.

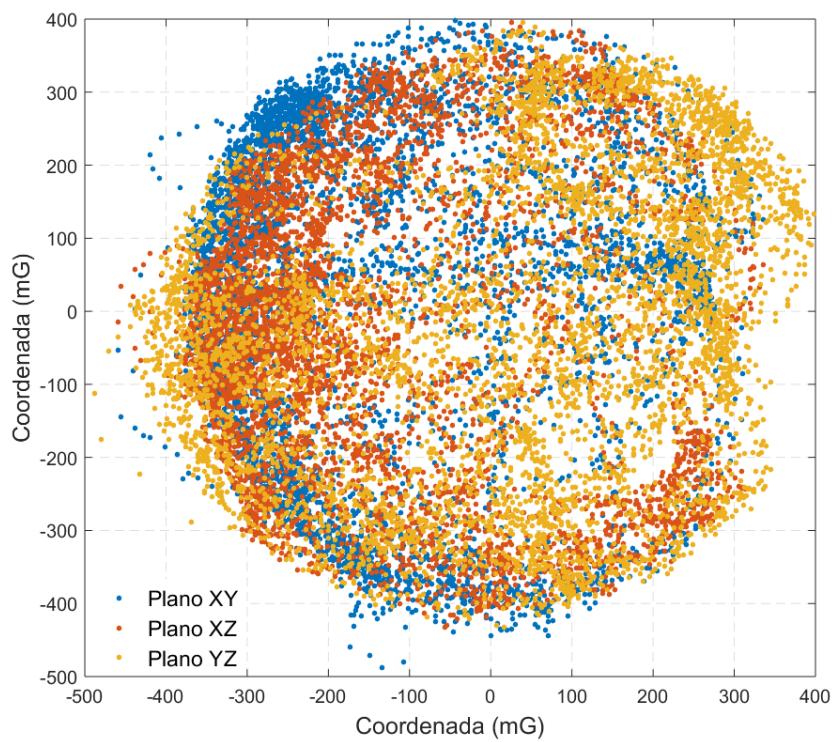


Figura 8.11: Valores obtenidos por el magnetómetro calibrado proyectados en los 3 planos del sistema de coordenadas cartesianas.

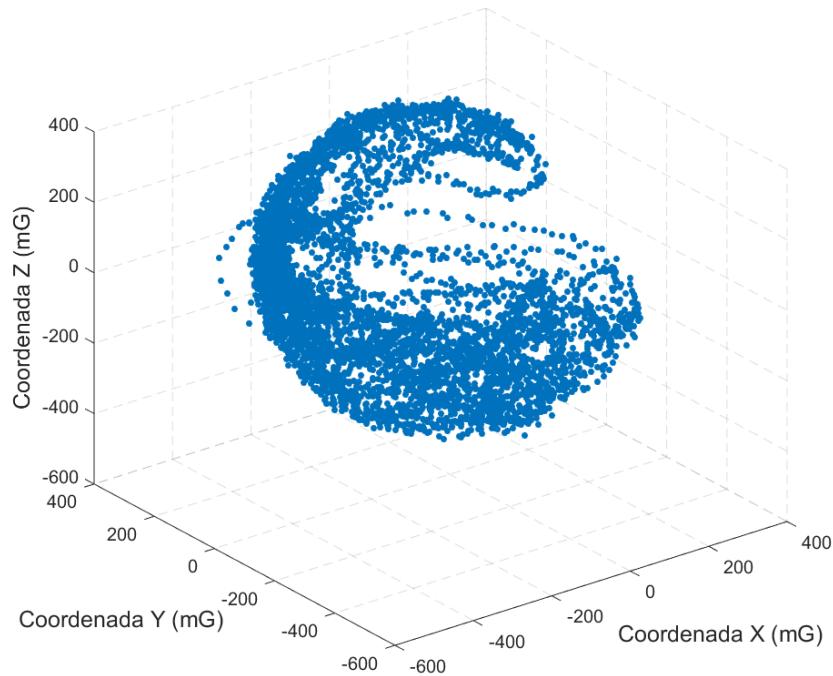


Figura 8.12: Valores obtenidos por el magnetómetro calibrado.

8.10.5. Transformación de los valores a ángulos Euler

En la situación actual, se cuenta con los **valores proporcionados por el acelerómetro, giroscopio y magnetómetro**, que consisten en cifras para los ejes X, Y y Z. Sin embargo, **estos datos en bruto no proporcionan la información necesaria para**

el cálculo de la desviación de Zowi, por lo que es necesario operar con ellos antes de poder obtener un resultado determinante.

A tal fin, existen numerosos algoritmos cuyo objetivo es determinar la orientación espacial del dispositivo, calculando los ángulos correspondientes al yaw, pitch y roll -ver Figura 8.13-.

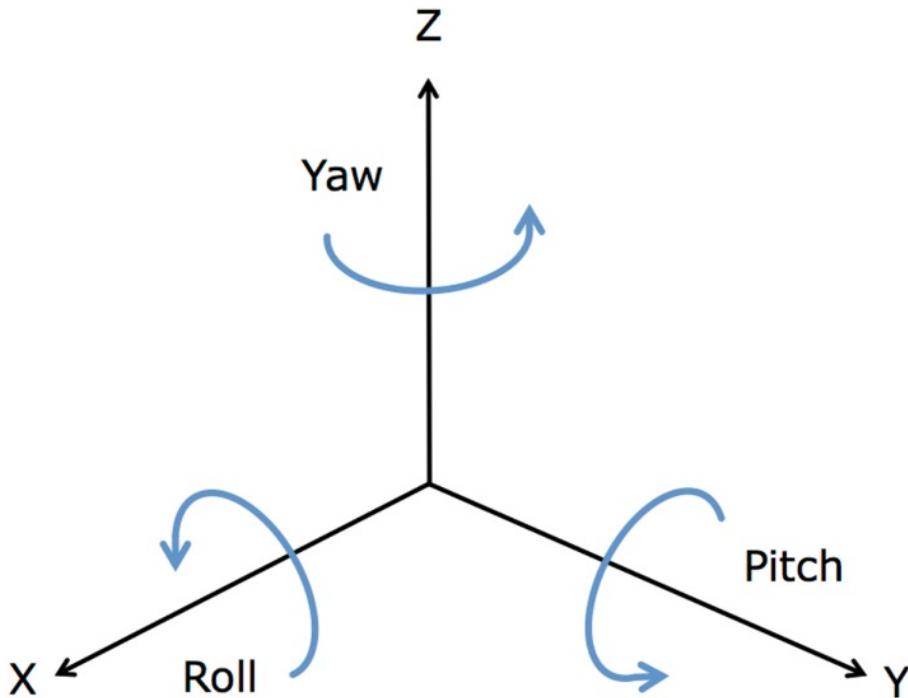


Figura 8.13: Esquema explicativo de los ángulos yaw, pitch y roll.

Atendiendo a la dirección y sentido de los ejes del MPU9250, el equivalente al cambio de dirección en el plano paralelo al suelo es el yaw. Por lo tanto, es únicamente ese el interesante en este Proyecto.

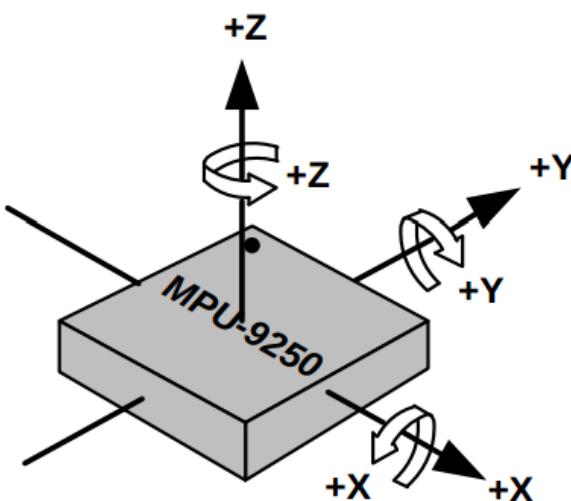


Figura 8.14: Dirección y sentido de los ejes del acelerómetro y el giroscopio en el MPU9250.

Volviendo a los algoritmos, algunos de los que se pueden utilizar en este tipo de casos son el filtro de Kalman y el complementario, siendo el razonamiento análogo al explicado en la sección MPU6050. No obstante, y dado que se manejan también los valores obtenidos

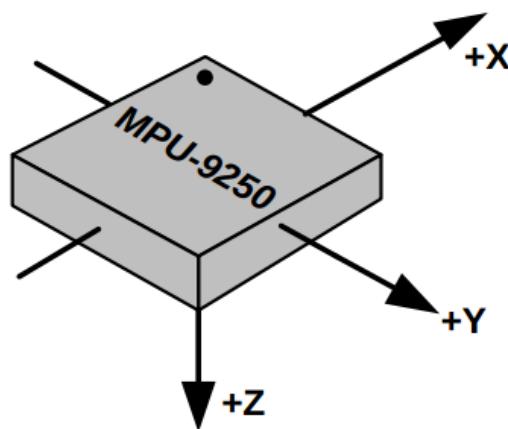


Figura 8.15: Dirección y sentido de los ejes del magnetómetro en el MPU9250.

mediante el magnetómetro, existe otra alternativa a tener en cuenta que destaca por su buen rendimiento y que se ha terminado usando en este TFM: el filtro de Madgwick.

Desarrollado por Sebastian Madgwick en 2009, consiste en la **obtención de cuaterniones por medio del cálculo y mezclado de datos de los 3 sensores**. Los cuaterniones, a su vez, son 4 variables cuyo valor es recalculado a medida que se itera con cada nuevo muestreo, minimizando errores propios de cada uno de los componentes.

Después del proceso, mediante la aplicación de fórmulas se puede **calcular el valor correspondiente a los 3 ángulos** en el sistema de coordenadas cartesianas, si bien en este caso sólo se hará para uno de ellos: el yaw. Para ello, se requiere conocer la arcotangente de las dos siguientes funciones:

$$func1 = 2 * (q_1 * q_2 + q_0 * q_3) \quad (8.8)$$

$$func2 = 2 * (q_0 * q_0 + q_1 * q_1 - q_2 * q_2 - q_3 * q_3) \quad (8.9)$$

De este modo, al finalizar el proceso y con cada nueva iteración, **se obtiene el número de grados que el dispositivo gira con respecto al eje de las Z**, acercándose enormemente al comportamiento deseado en este Proyecto.

8.10.6. Creación del nuevo algoritmo para caminar

El algoritmo para caminar implementado por defecto en Zowi basa su finalización en el tiempo empleado. Es decir, si se especifica un período de 2 segundos y 1 paso a realizar, este último se completará cuando pase ese intervalo.

Teniendo en cuenta que se pretende modificar la dirección del robot dinámicamente, este mecanismo no se adecua a las necesidades existentes; una vez se definen los parámetros iniciales de la función -tal y como se explica en la sección [Algoritmo para caminar](#)-, se ejecuta sin contemplar ningún otro factor. Por lo tanto, de cara a tomar datos de los sensores en tiempo de ejecución y poder corregir los desvíos, **se creó un nuevo algoritmo que basa su funcionamiento en el giro de los servos**, en lugar de en el tiempo o el número de pasos.

De este modo, se evalúa cuál es el ángulo inicial, determinando el final del ciclo en el momento en el que -tras completar el giro- se vuelve a obtener la misma cifra. Este

comportamiento se basa en el uso de ondas sinusoidales -Figuras 5.6 y 5.7-, que debido a su periodicidad devuelven el mismo valor con cada ciclo. En lugar de utilizar las ondas base, se han adaptado los parámetros para mejorar el comportamiento de Zowi, si bien este no es el factor crucial en el nuevo algoritmo. Tras las modificaciones realizadas para este TFM, **el método llamado para ejecutar el código de caminar acepta un nuevo argumento: el ángulo de corrección.**

En primer lugar, y estando en reposo, el protagonista **recopila medidas de los sensores para calibrarlos y obtener la dirección de referencia** -lo que se considera hacia delante-. A continuación, comienza a caminar, calculando tras el primer paso -con los mecanismos explicados en [Transformación de los valores a ángulos Euler](#)- la dirección a la que apunta Zowi y comparándola con la teórica obtenida por trigonometría. En función de la diferencia existente entre ambas, el robot girará un ángulo mayor o menor, **compensando las desviaciones durante la marcha.**

Este mecanismo es teóricamente solvente, pero presenta algunos inconvenientes importantes derivados del uso de la plataforma Arduino.

Tiempo de cómputo y multihilos

El algoritmo original para caminar es perfectamente funcional, desde el punto de vista de los componentes y de la velocidad de cómputo. No existe ningún retraso ni falta de potencia a la hora de procesar las operaciones necesarias.

No es el caso, no obstante, del algoritmo de Madgwick. Cada iteración requiere de 3 ó 4 muestreos de datos, lo que sumado al hecho de que Arduino no es multihilo, puede generar un problema en lo referente al rendimiento del sistema.

La idea ideal sería **desempeñar el proceso de caminado en el hilo principal**, mientras que **uno secundario se encargaría de llevar a cabo todas las tareas relacionadas con el cálculo del ángulo**. Sin embargo, como se ha comentado, esta alternativa no es posible en esta plataforma, por lo que fue necesario adecuar este concepto a aquello viable en la placa base de Zowi.

Así, dentro del *loop* principal se combinan las dos acciones, habiendo modificado ligeramente el algoritmo de caminar: **si bien inicialmente en cada iteración de este bucle se giraba 1º, se aumentó esta cifra para que aproximadamente en el mismo tiempo se consiga el mismo giro.** Esto se debe a que en lugar de emplear el 100 % del tiempo en todo lo referente al cálculo del yaw, **en la versión final se requiere emplear parte del ciclo a caminar, y otra parte a esta tarea.**

En otras palabras, si antes en 100 ms se giraban 10º en 10 iteraciones, ahora en el mismo intervalo se giran los mismos grados, pero en -por ejemplo- 2 iteraciones. Se consigue así un **comportamiento notoriamente más fluido**, que sin llegar a estar al nivel del algoritmo original, proporciona una mejora en lo referente a la corrección de la trayectoria.

Falta de viabilidad del concepto

No obstante, y a pesar de lo prometedor de la idea, **existen algunos factores que impiden que el funcionamiento sea el esperado a priori.**

En primer lugar, y menos importante, cabe destacar el **peor rendimiento comentado más arriba**. No constituye un factor determinante por sí solo, pero el hecho de

que Zowi tenga un comportamiento poco fluido al caminar genera tanto la **sensación de un producto mal terminado, como de poco profesional**. Sería, sin embargo, un handicap admisible si no existiera el siguiente punto en esta lista.

En segundo lugar, considerándose el **motivo por el que se desechó la idea del IMU después de emplear numerosas horas en investigación**, la inclusión de estos sensores **no garantiza al 100 % la correcta ejecución de la actividad** de la cuadrícola. Dos son los motivos:

- **IMU impreciso:** un MPU9250 tiene un coste de unos 12€. Partiendo de esta premisa, no es de extrañar que aunque la precisión sea buena, no se pueda comparar con alternativas utilizadas en sectores como la aeronáutica.

Zowi adolece de una forma de caminar errática, y la **solución propuesta consistía en añadir un componente con el objetivo de compensar este comportamiento**. No obstante, **¿cómo se puede corregir la imprecisión con una pieza que es de por sí imprecisa?** Una buena calibración contribuye a mejorar el sistema en todos los sentidos, pero **cuando se pretende alcanzar una resolución de aproximadamente 1º ó 2º, cualquier desviación es demasiada**.

- **La imprecisión de Zowi sigue siendo limitante:** se ha comentado con anterioridad cuál pretendía ser -e incluso se llegó a probar- la ejecución de esta infraestructura: **obtener una dirección inicial, calcular el ángulo y compararlo con la cifra de giro teórica para aplicar una corrección.**

Y es el aspecto teórico el que obtiene el dudoso título de problemático. Tomando como base las explicaciones del [algoritmo para caminar](#), se puede afirmar y comprobar que los servos de las caderas giran como mucho 60º: 30º en un sentido y otros 30º en el contrario. Así, aplicando trigonometría es posible calcular cuál es el ángulo girado en cada paso.

Si se conoce esta cifra, y se conoce -gracias al IMU- el ángulo girado realmente, la diferencia constituiría el desfase que se debería compensar. Pero, **¿si Zowi es impreciso, cómo se puede asegurar que en el paso ha girado exactamente el valor teórico calculado?** La respuesta, aunque desalentadora, es que no se puede. De este modo, **aunque sí que se note una mejoría con respecto a la funcionalidad de fábrica**, la solución no es completamente viable.

Cabe destacar que se podría **evaluar el giro basándose únicamente en las medidas del IMU**, calculando los grados empleados con respecto a la dirección inicial. Sin embargo, entra en juego el primer punto comentado: **este sensor tampoco es preciso, por lo que en cualquier caso la situación sería análoga a la encontrada de fábrica.**

8.10.7. Sensor de infrarrojos... En la cabeza

Los siguiéneas llevan existiendo desde que la robótica se conoce como tal. **Gracias a los sensores de infrarrojos, se pueden distinguir** obstáculos o -como resulta de interés en este caso- **distintos colores**.

La dificultad existente en este caso no es la complejidad del desarrollo, ya que conceptualmente es razonablemente sencillo. **La dificultad reside en que estos sensores**

deben estar a poca distancia del suelo para poder detectar una línea negra en él. Contextualizando con Zowi, esto **implicaría situar estos componentes en los pies**, con la consecuente disposición de los cables en el exterior de la carcasa. Dado que ZowiApprende será utilizada por niños de Educación Infantil, esta posibilidad no es viable.

No obstante, se pueden idear mecanismos que permiten solventar el inconveniente de la distancia. En este TFM, **la opción propuesta es situar la nueva pieza en la cabeza del robot**, que conceptualmente choca con todo lo visto en el mercado en lo que respecta a siguelíneas.

Este concepto es motivado por tres factores principales:

- **Ausencia de cables a la vista de los usuarios:** dado que **la cabeza es el único lugar en el que se pueden ubicar nuevos componentes** -las patas no disponen de espacio-, es imprescindible que toda la circuitería se aloje en ese hueco. Tal es el caso con las luces LED o con las pruebas realizadas con el MPU9250, si bien en eran sensores cuyo funcionamiento no dependía directamente de la distancia al suelo.
- **Funcionamiento de los sensores de infrarrojos con espejos:** tras llevar a cabo diversas búsquedas por Internet y hacer pruebas caseras, se determinó que **un sensor de infrarrojos funciona reflejando su señal en un espejo**. Esta posibilidad abre un amplio abanico de alternativas en su uso, ya que **se puede experimentar con los ángulos de incidencia y reflexión** para elaborar una estructura que sobrepase con creces la experiencia conseguida con Zowi recién salido de la caja.
- **Impresión 3D:** como se ha comentado anteriormente, **cualquiera con acceso a una impresora 3D puede crear desde 0 todas las piezas de plástico de Zowi**; e incluso inventar otras nuevas.

De este modo, la cabeza del robot no supone un límite real, ya que **con nuevas formas y/o colores es perfectamente posible alterar tanto estética como funcionalidad**. De hecho, en el caso que se trata, el verdadero protagonismo recae en la nueva “gorra”: un componente -que se puede ver en la Figura 11.7- cuyo principal objetivo es la **sujección de dos sensores de infrarrojos y la incorporación de un espejo que refleje su señal y la redirija al suelo**.

A pesar de haber impreso la nueva carcasa y su complemento, **tras montar el conjunto se pudo determinar un problema determinante**, cuya posible aparición se había sospechado: la distancia.

Situando un objeto a poca distancia del espejo, los infrarrojos son capaces de captarlo, incluso detectando con total facilidad una línea negra sobre un fondo de otro color. Sin embargo, **la distancia que separa la cabeza de Zowi del suelo es demasiado amplia como para que este comportamiento ideal se pueda replicar en un caso real**.

8.10.8. Implementación definitiva

A la vista de las dificultades existentes, y aunque ambas propuestas quedan planteadas como concepto con potencial -y con viabilidad práctica, si se le dedica el tiempo adecuado-

, se decidió optar por una solución cuyas principales características son la sencillez, la fácil implementación y la falta de precisión. Este último factor, no obstante, no repercute hasta el punto de ser crucial, ya que en el ámbito en el que se va a utilizar ZowiApprende es prescindible.

Se creó un algoritmo que permite a Zowi girar en la vertical, sin existir un desplazamiento -según el comportamiento de fábrica, el robot giraba a medida que avanzaba en su recorrido-. Aunque carece de la exactitud que se podría haber conseguido con un IMU o un siguelíneas, se considera una solución solvente dada la ausencia de otras propuestas. Este giro se combina con 2 pasos rectos, que equivalen, aproximadamente, a la distancia media entre las celdas.

Concluyendo, **Zowi realizaría el movimiento por la cuadrícula combinando intervalos donde camine recto, con giros de 90º sin desplazamiento**. En caso de apreciarse un error exagerado, tanto los niños como la maestra podrán corregirlo una vez iniciado.

Capítulo 9

Diseño de ZowiApprende

Este capítulo constituye el desarrollo a nivel de ingeniería del sistema, centrándose en información técnica que explique a nivel teórico cómo funciona Zowi detrás de su carcasa.

9.1. Diagramas de paquetes

ZowiApprende está constituida por dos grandes bloques bien diferenciados, tal y como se ha mencionado en repetidas ocasiones: la aplicación Android y el código Arduino.

Los diagramas asociados a ambas partes se pueden ver a continuación. Cabe destacar que si bien no se hará un análisis exhaustivo de las clases, métodos y propiedades -la sección [Diagramas de clases](#) cumple este cometido-, sí se explicará a nivel general cuál es el cometido de cada bloque estructural.

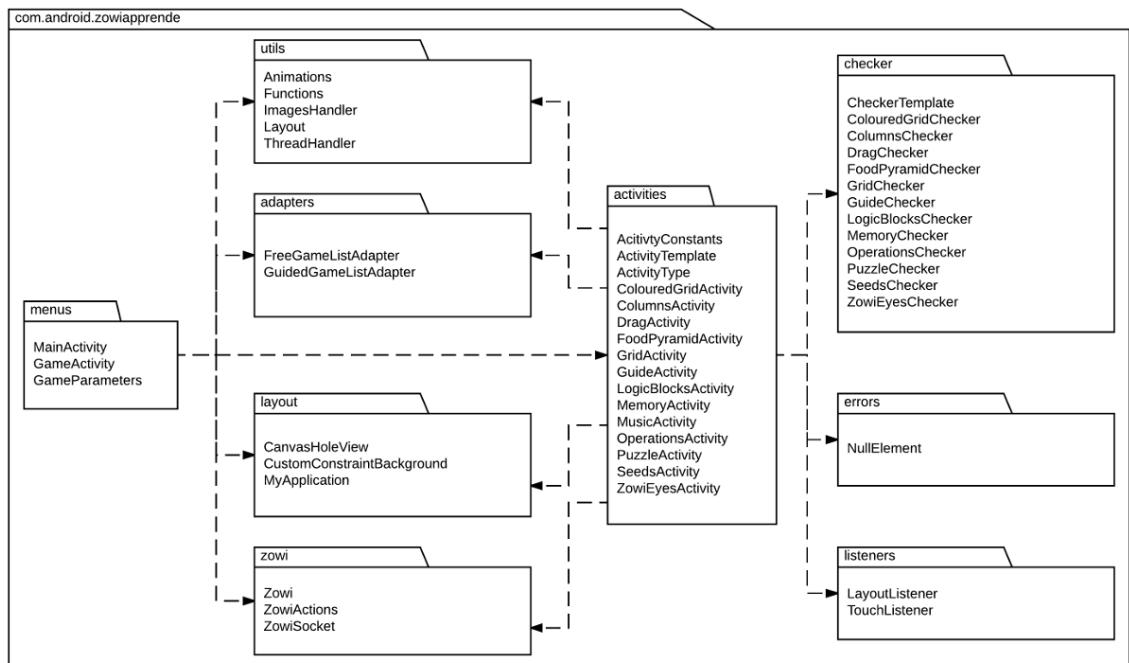


Figura 9.1: Diagrama de paquetes de la aplicación Android.

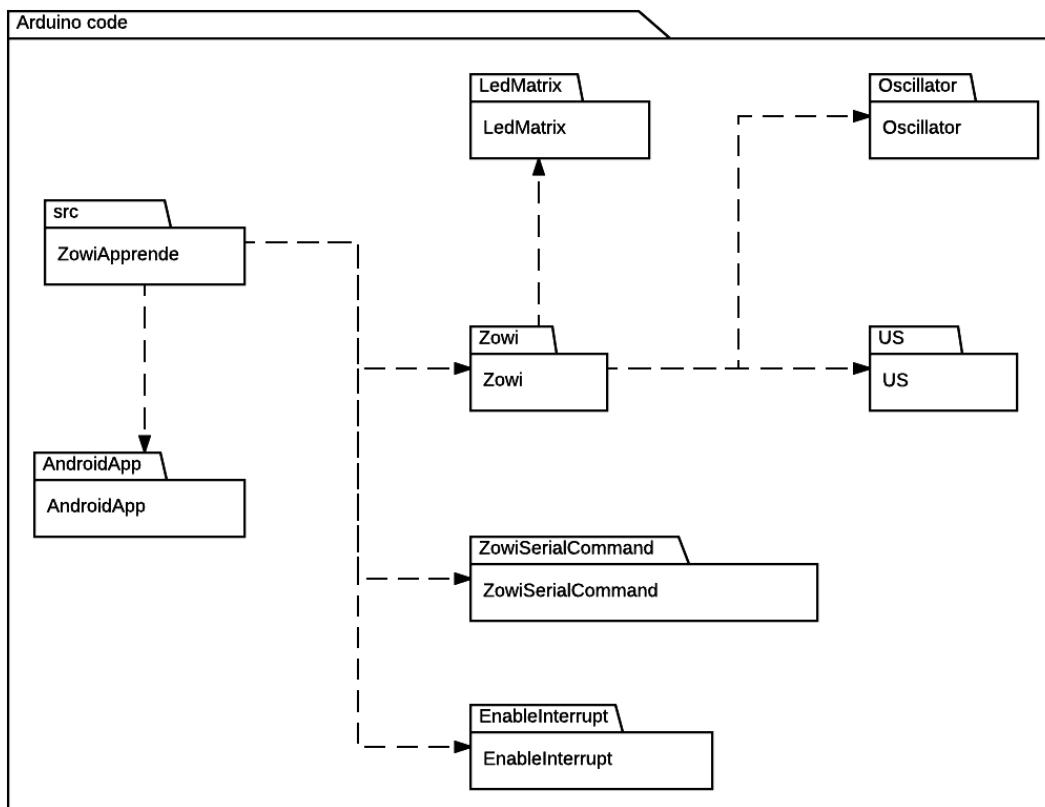


Figura 9.2: Diagrama de paquetes del código Arduino.

9.1.1. Aplicación Android

Esta sección detalla la principal función de cada paquete de la aplicación desarrollada en Java para Android.

- **activities**: directorio más importante del conjunto, ya que en él se ubican las clases encargadas de lanzar y dibujar en pantalla las distintas actividades diseñadas previamente.

Alberga además tres clases auxiliares: `ActivityConstants`, `ActivityTemplate` y `ActivityType`, cuya función es proporcionar una estructuración sólida, claridad de código y una alta escalabilidad.

- **adapters**: contiene los *adapters* personalizados utilizados en la carga de contenido tanto del juego libre como del juego dirigido.
- **checker**: paquete que contiene las **clases y métodos necesarios para corregir las actividades**. Se decidió separar esta funcionalidad de los archivos principales de pintado de ejercicios para mejorar la estructuración y hacer el código más manejable. Al igual que en el paquete “activities”, se cuenta con una clase auxiliar: `CheckerTemplate`, clase abstracta que aumenta la rigidez en la implantación de nuevos correctores.
- **errors**: alberga **clases utilizadas al lanzar excepciones**.

- **layout:** contiene información utilizada para **gestionar aspectos directamente relacionados con la interfaz del dispositivo**, como la carga de un fondo de pantalla en un ConstraintLayout o la modificación de la fuente de la aplicación.
- **listeners:** contenedor de los *listeners* utilizados tanto en la carga de la interfaz de las actividades como en su resolución.
- **utils:** **funciones comunes de utilidad a todo el proyecto** se han clasificado en este paquete, como una clase encargada de gestionar la carga de imágenes u otra responsable de llevar a cabo las animaciones existentes.
- **zowi:** uno de los directorios más importantes, ya que arrastra la responsabilidad de la comunicación con Zowi. Aquí se han organizado las **clases cuyo objetivo es establecer comunicación con el robot, mantenerla y enviar comandos**.

9.1.2. Código Arduino

En lo referente a la estructura del código Arduino, cada paquete ejemplifica una librería utilizada por el archivo de código principal, residiendo -mayoritariamente- en este último la ejecución de los comandos recibidos desde la tablet.

- **AndroidApp:** librería encargada de **hacer de intermediaria entre la información recibida por bluetooth y la clase que maneja a Zowi**.
- **EnableInterrupt:** librería utilizada para **gestionar interrupciones externas**, en este caso de los botones traseros del robot.
- **LedMatrix:** código cuya función es la **interacción con la matriz de LEDs**.
- **Oscillator:** librería en cuyas clases reside toda la **funcionalidad de los servos**, encargándose de controlar el ángulo y la velocidad de giro en función de los parámetros especificados por el programador.
- **US:** clases encargadas de la **gestión del sensor de ultrasonidos**.
- **Zowi:** dependencia más importante del proyecto. Su cometido es la **ejecución de la mayoría de la funcionalidad del robot**, haciendo de puente entre el archivo .ino principal y el resto de librerías.
- **ZowiSerialCommand:** código encargado de la **gestión del puerto serie**, que permite que ZowiApprende se comunique con el robot -y viceversa-.
- **src:** en este directorio se ubica el archivo con el código de ejecución principal, el fichero .ino. De esta forma, es aquí donde se configuran las actividades como tal, interactuando a posteriori con el resto de software disponible en las distintas dependencias.

9.2. Diagramas de clases

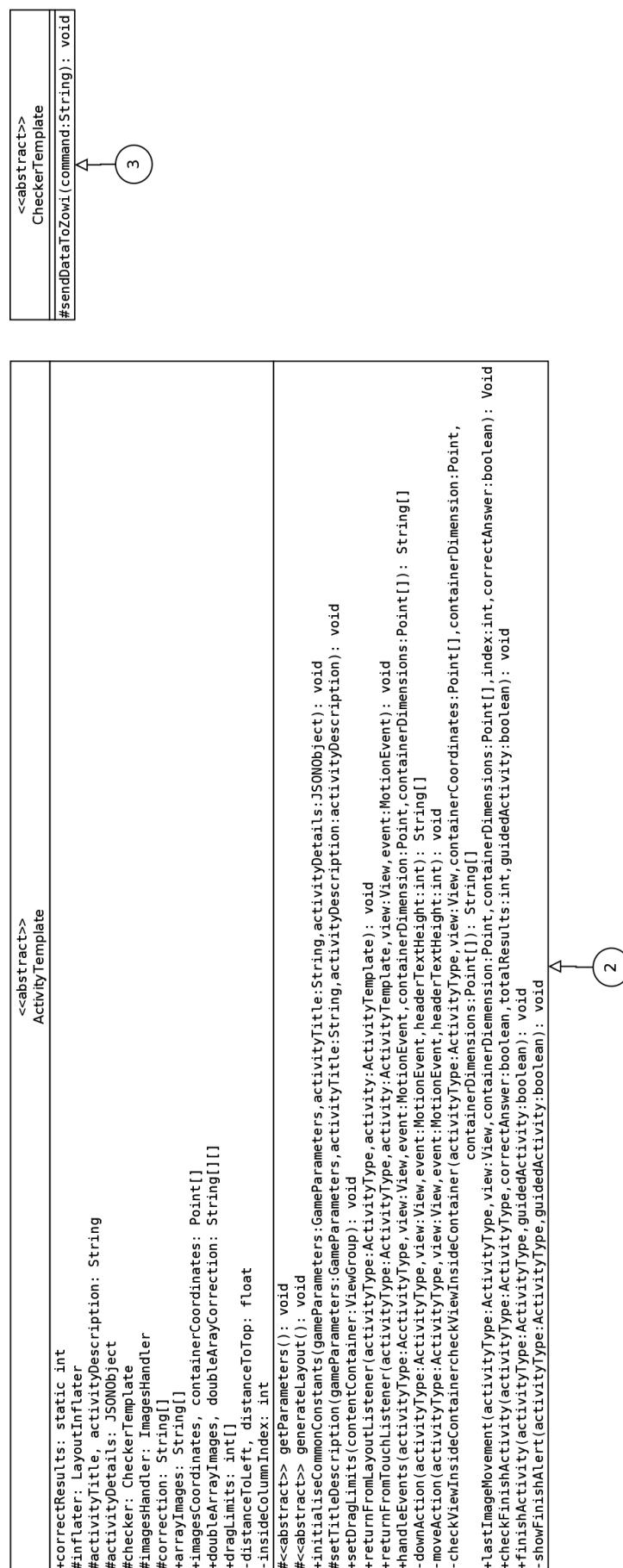


Figura 9.3: Esquema de las clases abstractas.

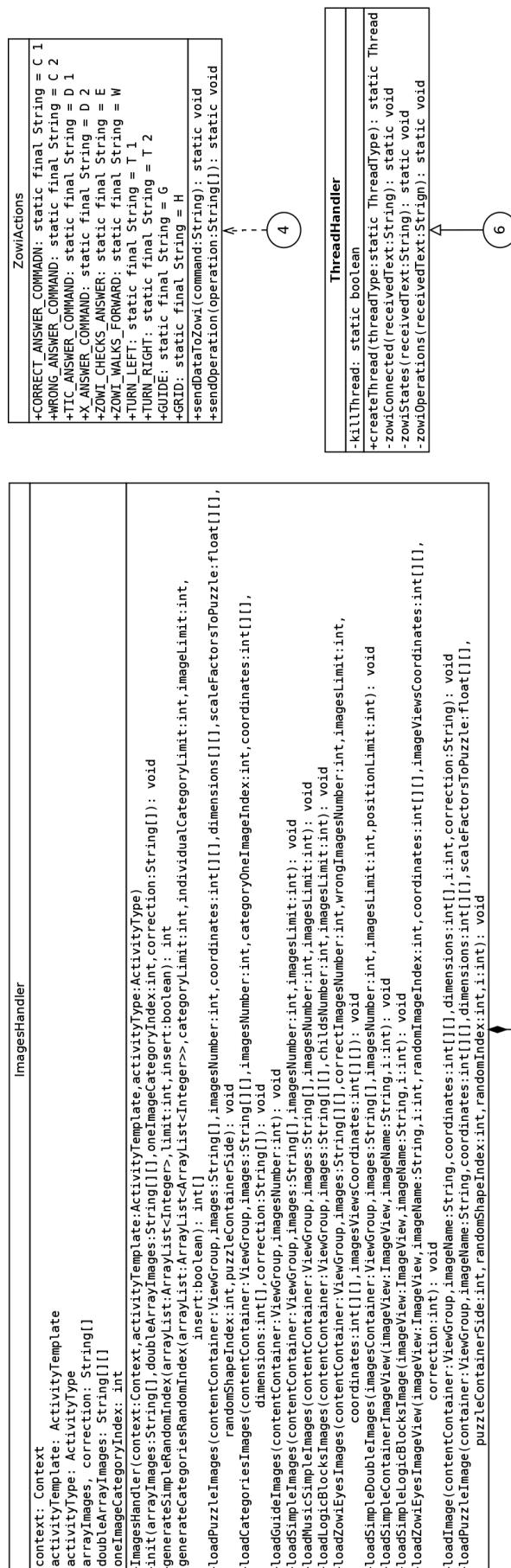


Figura 9.4: Esquema de las clases utilizadas en todo el Proyecto.

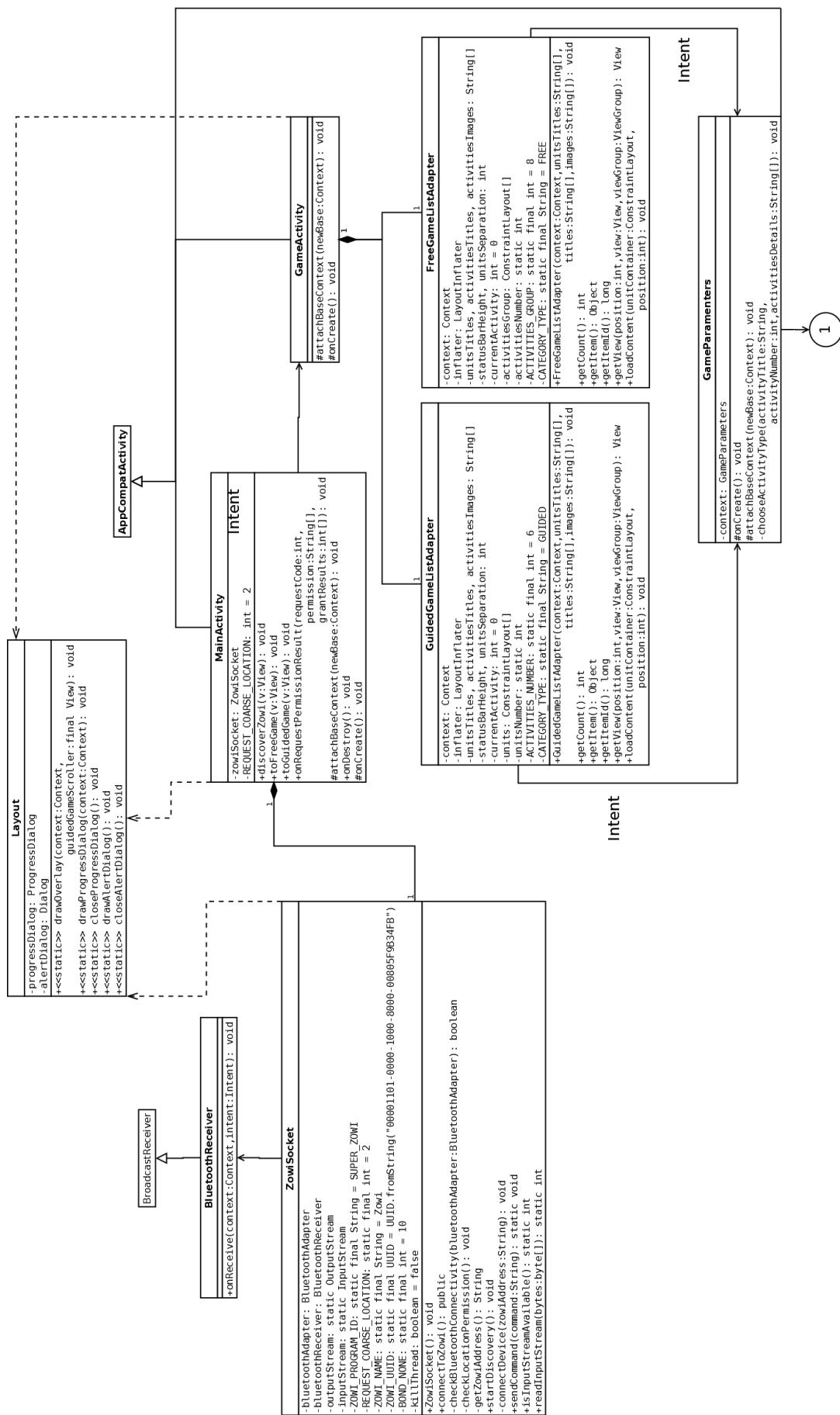


Figura 9.5: Comienzo de la aplicación y selección de actividad.

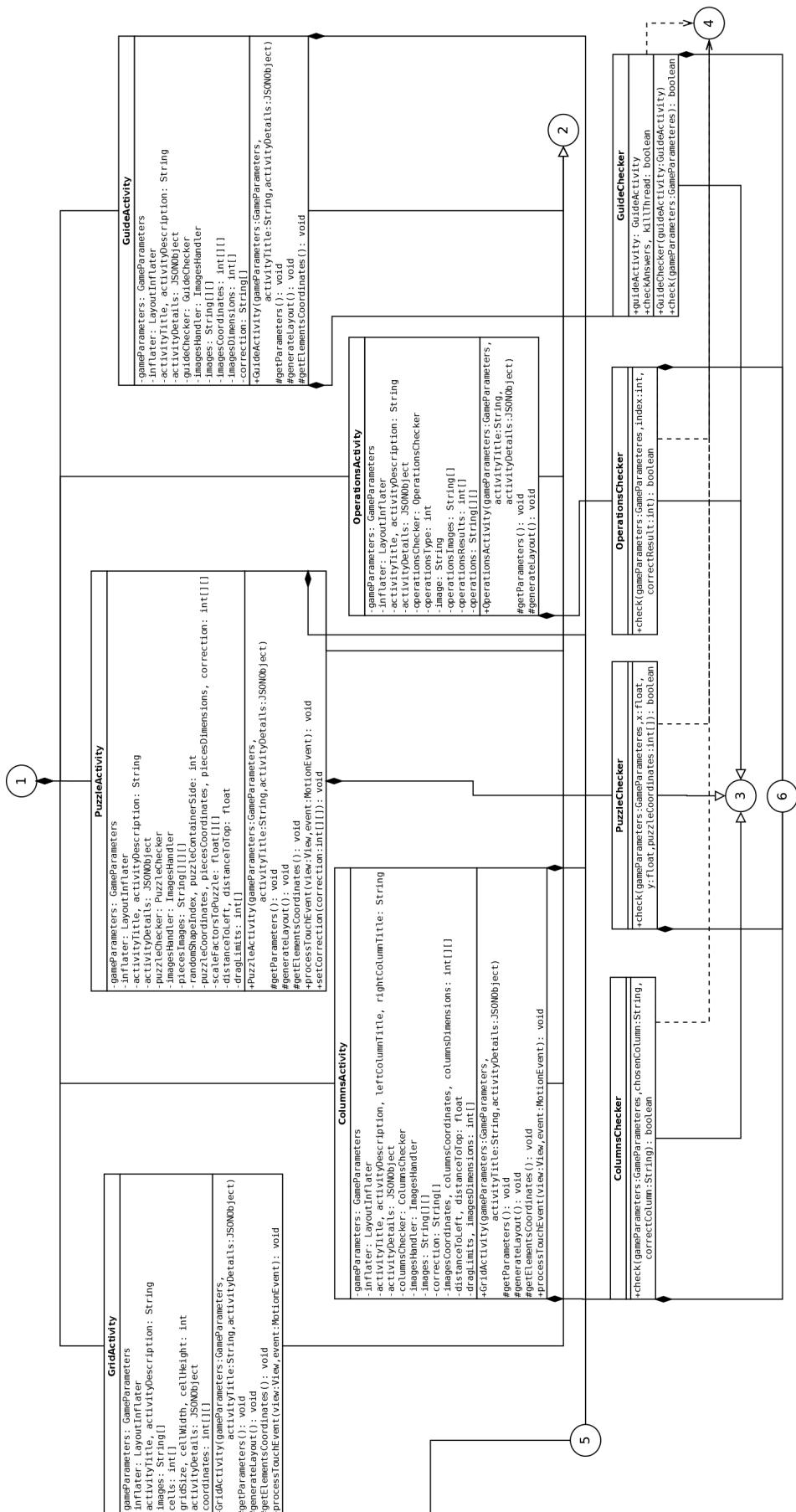


Figura 9.6: Cinco de los doce tipos de actividades.

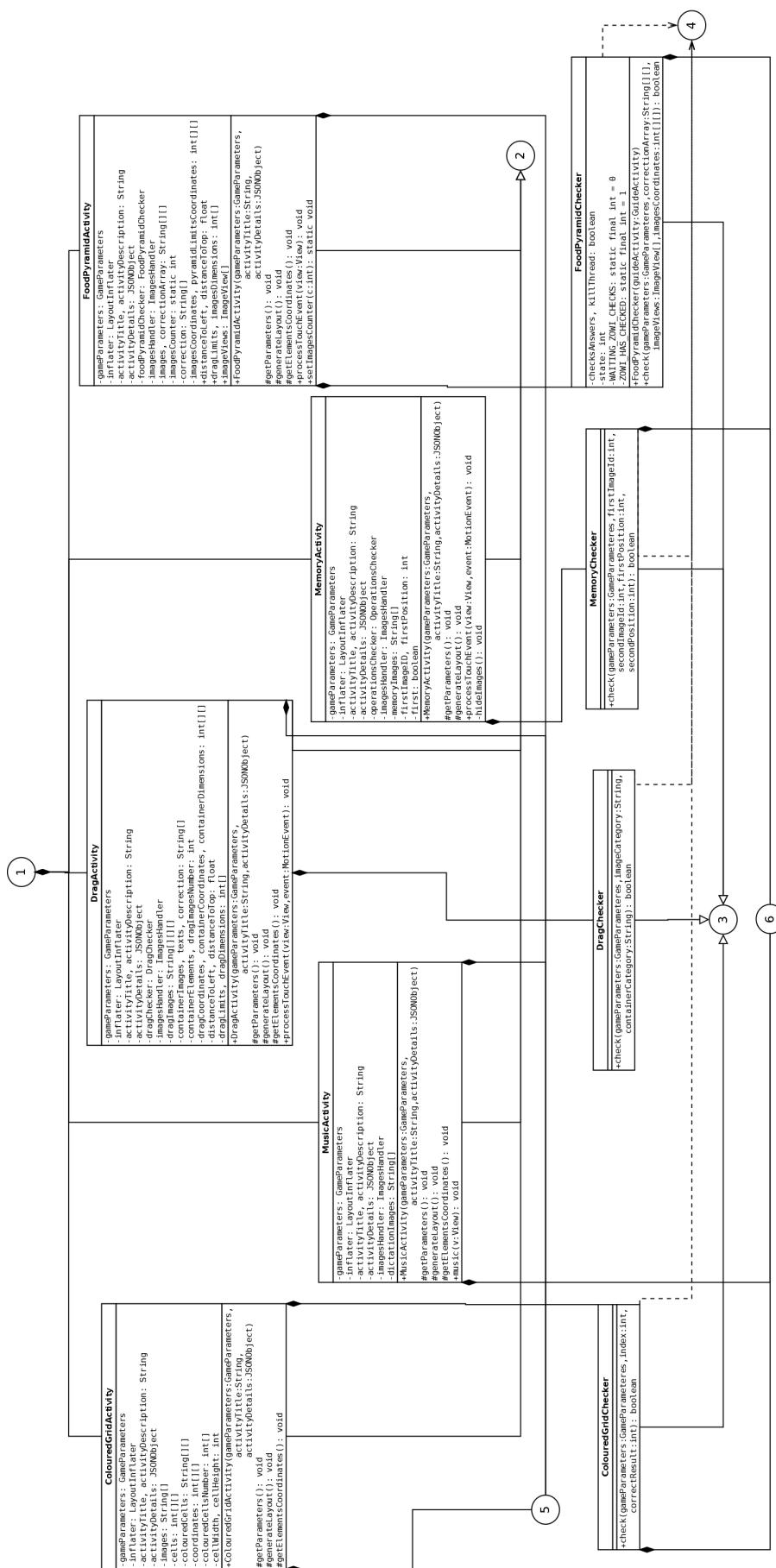


Figura 9.7: Cinco de los doce tipos de actividades.

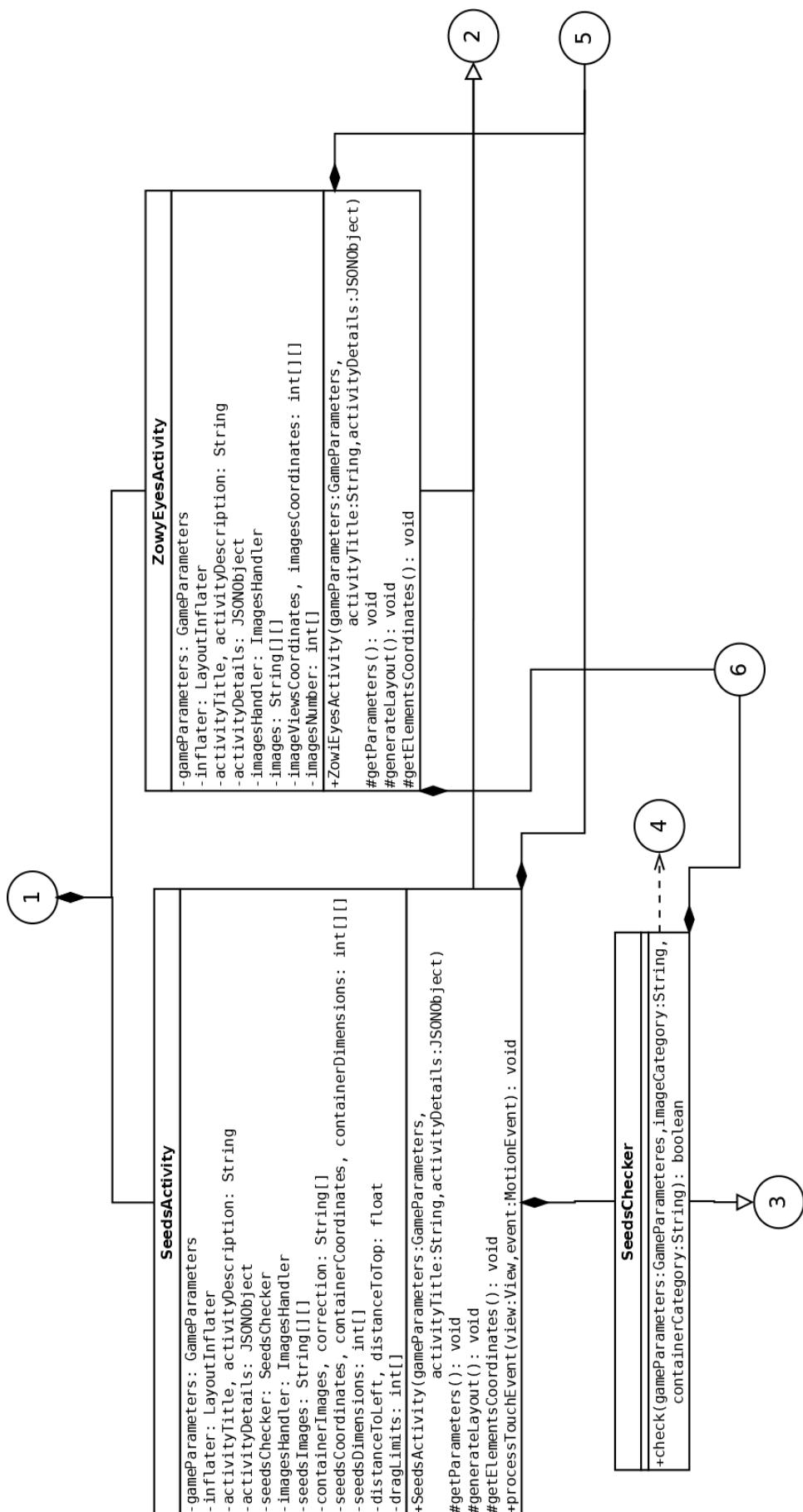


Figura 9.8: Dos de los doce tipos de actividades.

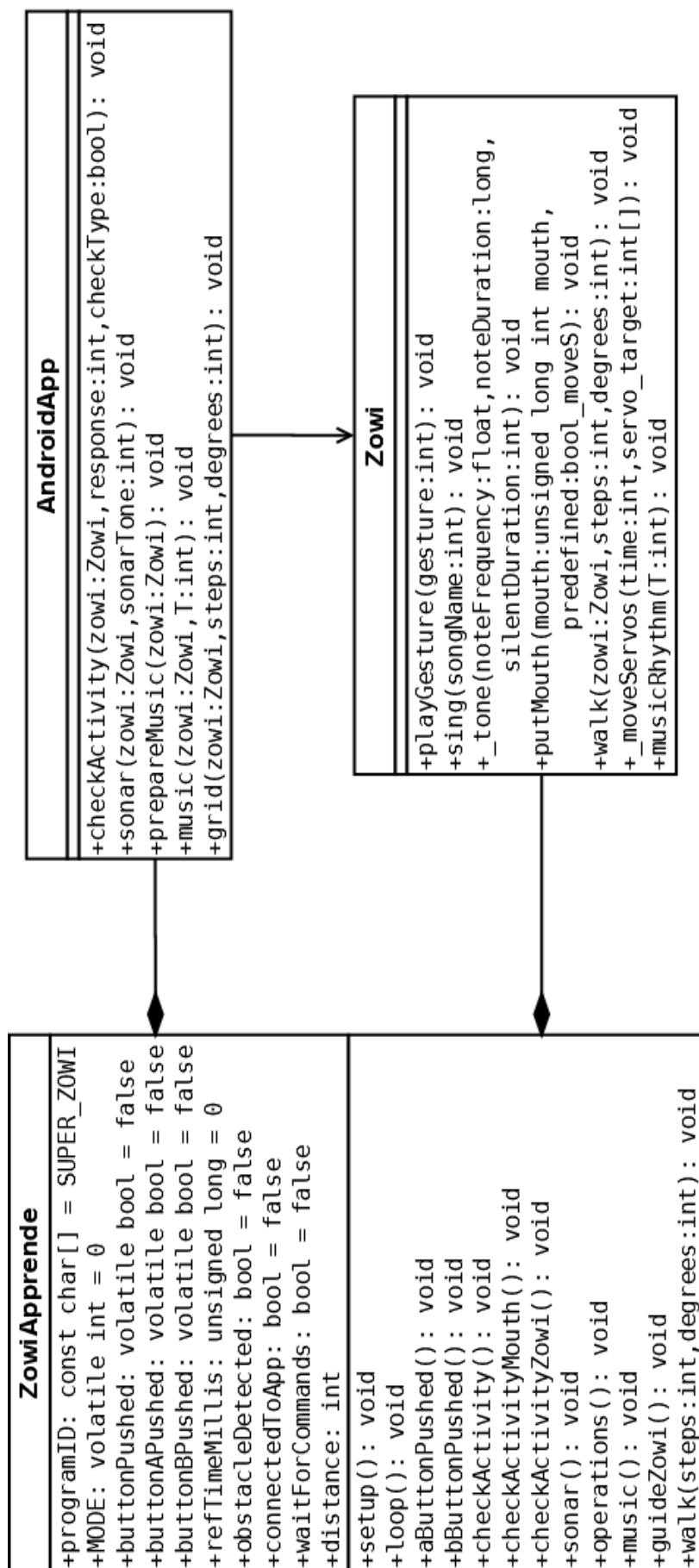


Figura 9.9: Simplificación de las clases Arduino implementadas en Zowi.

9.3. Diagramas de componentes

El sistema de ZowiApprende lo forman **dos subsistemas bien diferenciados: la aplicación Android y el código Arduino**. Teniendo esta premisa en cuenta, a continuación se desarrollan los esquemas correspondientes a cada uno de ellos.

Con respecto a la comunicación entre las partes, se especifican las interfaces que la describen, contando con los siguientes tipos:

- **Requerida:** indica que **un componente debe conectarse con otro** para llevar a cabo una determinada operación.
- **Proveída:** complementaria a la anterior, indica que **un componente debe recibir una conexión**.
- **Puerto in:** petición recibida desde otro componente.
- **Puerto out:** petición realizada a otro componente.

9.3.1. Componentes del sistema

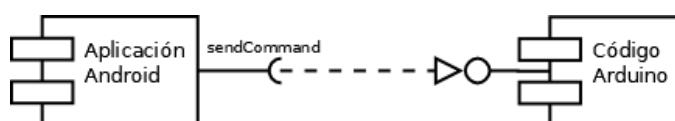


Figura 9.10: Diagrama de componentes del sistema.

Componentes

- **Aplicación Android:** subsistema utilizado tanto por los alumnos como por el profesor para resolver actividades y aprender. Constituye todo el software instalado en la tablet, y permite la interacción con el robot a través de los distintos tipos de ejercicio.
- **Código Arduino:** subsistema encargado de hacer funcionar a Zowi, gestionando y controlando todos sus componentes y recibiendo las peticiones de la aplicación.

Relaciones

La comunicación entre ambas partes se realiza de forma directa utilizando la tecnología Bluetooth. La información intercambiada se ha definido con anterioridad, siguiendo el patrón utilizado por el software de fábrica proporcionado por BQ.

Interfaz	Tipo	Tecnología	Descripción
sendCommand	Interfaz requerida	Bluetooth	Envía un comando con el formato adecuado a Zowi

Tabla 9.1: Interfaces de la aplicación Android

Interfaz	Tipo	Tecnología	Descripción
sendCommand	Interfaz proveída	Bluetooth	Devuelve, en caso de que sea necesario, un ACK de confirmación

Tabla 9.2: Interfaces del código Arduino

9.3.2. Componentes de la aplicación Android

La arquitectura presente en **ZowiApprende comparte gran parte de la funcionalidad del patrón MVC** (Modelo Vista Controlador). La principal particularidad consiste en que no se requiere el guardado persistente de datos, ya que la información mostrada y la lógica ejecutada en las aplicaciones puede ser volátil sin ningún inconveniente.

Se cuenta, por tanto, con **dos secciones bien diferenciadas**: aquella encargada de la lógica de negocio -escoger las imágenes, generar las correcciones de los ejercicios, definir las posiciones de los elementos en pantalla...-, y aquella cuya labor es representar la información.

Dado que este Proyecto trata sobre una actividad para niños, **se considera de más importancia de lo habitual la labor desempeñada por la vista**. Si bien un alumno de Educación Infantil puede no valorar qué hace un juego o una aplicación, puede sentirse tremadamente atraído por una estética afín a sus gustos, lo que constituye un reclamo para que estos hagan uso de ZowiApprende.

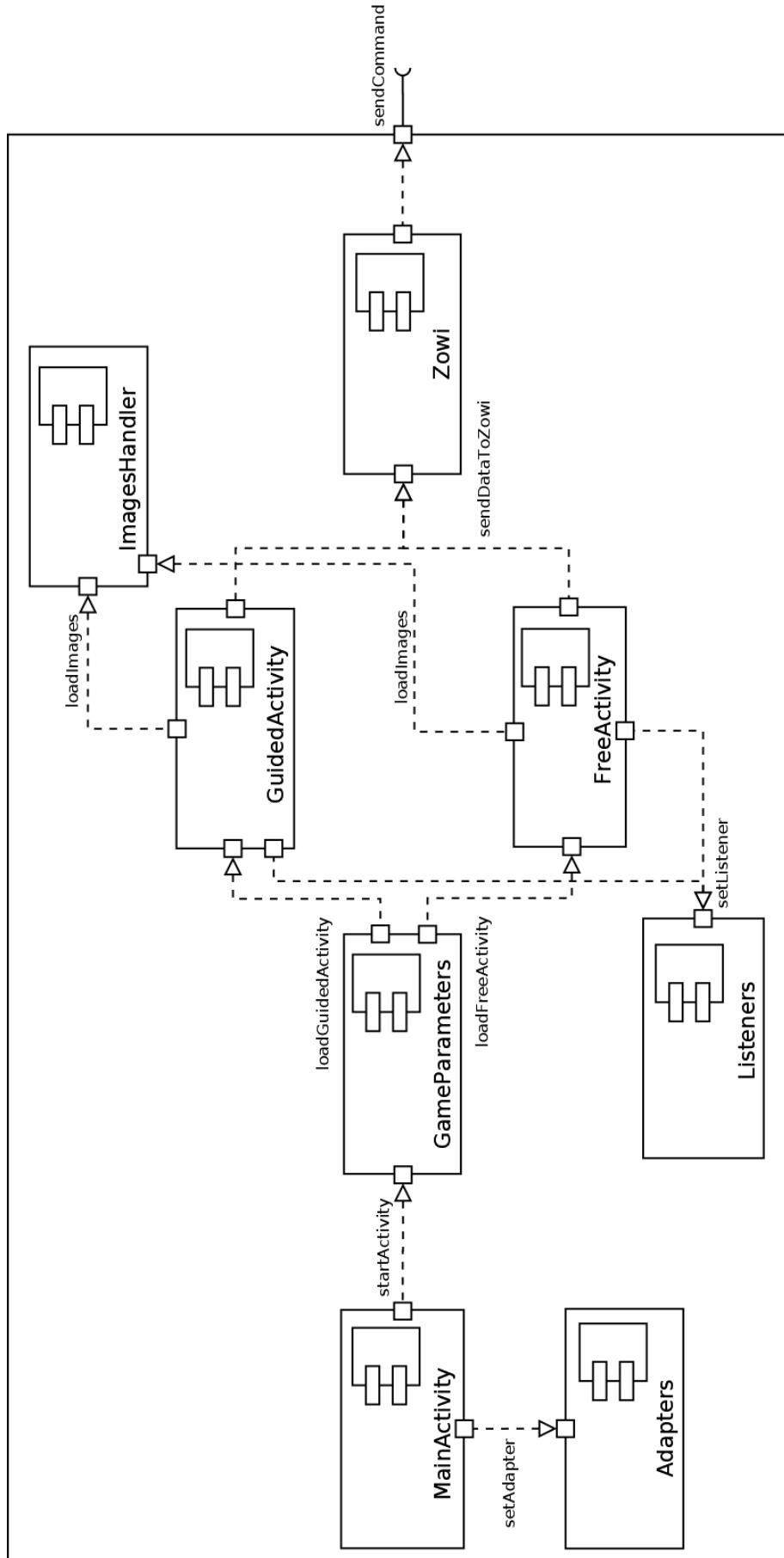


Figura 9.11: Diagrama de componentes de la aplicación Android.

Componentes

- **MainActivity**: actividad que se ejecuta en el arranque de la aplicación, constituye el punto de partida de ZowiApprende. Desde este punto se puede seguir el flujo que se puede ver en la Figura 6.9.

Interfaz	Tipo	Tecnología	Descripción
setAdapter	Puerto <i>out</i>	Method call	Se establece un adapter sobre el contenedor definido para permitir la carga de elementos en pantalla
startActivity	Puerto <i>out</i>	Method call	Se inicia una nueva actividad a partir de un Intent pasado como parámetro

Tabla 9.3: Puertos de MainActivity

- **Adapters**: clases que permiten la carga de los elementos del menú de las dos modalidades de juego. Extendiendo de *BaseAdapter*, proporcionan los métodos necesarios para crear una lista eficiente de objetos basados en layouts creados previamente.

Interfaz	Tipo	Tecnología	Descripción
setAdapter	Puerto <i>in</i>	Method call	Se reciben los parámetros del menú y genera su interfaz

Tabla 9.4: Puertos de Adapters

- **GameParameters**: clase que inicia la actividad escogida por el alumno en la interfaz, pasándole información necesaria para la correcta carga de todos los recursos.

Interfaz	Tipo	Tecnología	Descripción
startActivity	Puerto <i>in</i>	Method call	Se recibe un Intent e inicia una determinada actividad
loadGuidedActivity	Puerto <i>out</i>	Method call	Se ejecuta un ejercicio del juego dirigido con la información leída del archivo XML
loadFreeActivity	Puerto <i>out</i>	Method call	Se ejecuta un ejercicio del juego libre con la información leída del archivo XML

Tabla 9.5: Puertos de GameParameters

- **GuidedActivity**: corresponde a uno de los ejercicios pertenecientes al modo de juego guiado. Habiendo recibido los parámetros de GameParameters, se encarga de la creación de la interfaz a partir de los layouts creados previamente a tal fin, así como de la carga de imágenes, la obtención de las correcciones, etc.

Interfaz	Tipo	Tecnología	Descripción
loadGuidedActivity	Puerto <i>in</i>	Method call	Se recibe la información necesaria para cargar el layout de la actividad
setListener	Puerto <i>out</i>	Method call	Se definen los <i>listeners</i> necesarios en la ejecución del ejercicio
loadImages	Puerto <i>out</i>	Method call	Se ejecuta el método de carga de imágenes con los parámetros adecuados a cada actividad
sendDataToZowi	Puerto <i>out</i>	Method call	Se llama al método que envía los comandos a Zowi a través del <i>socket</i>

Tabla 9.6: Puertos de GuidedActivity

- **FreeActivity:** el razonamiento es análogo con respecto a GuidedActivity.

Interfaz	Tipo	Tecnología	Descripción
loadFreeActivity	Puerto <i>in</i>	Method call	Recibe la información necesaria para cargar el layout de la actividad
setListener	Puerto <i>out</i>	Method call	Se define los <i>listeners</i> necesarios en la ejecución del ejercicio
loadImages	Puerto <i>out</i>	Method call	Se ejecuta el método de carga de imágenes con los parámetros adecuados a cada actividad
sendDataToZowi	Puerto <i>out</i>	Method call	Se llama al método que envía los comandos a Zowi a través del <i>socket</i>

Tabla 9.7: Puertos de FreeActivity

- **Listeners:** clases encargadas de la recepción de algunos tipos de eventos, como de carga completa de la interfaz o de toque.

Interfaz	Tipo	Tecnología	Descripción
setListener	Puerto <i>in</i>	Method call	Se define el <i>listener</i> adecuado en función de la llamada recibida desde una actividad

Tabla 9.8: Puerto de Listeners

- **ImagesHandler:** componente cuya función es la gestión de las imágenes de las distintas actividades.

Interfaz	Tipo	Tecnología	Descripción
loadImages	Puerto <i>in</i>	Method call	Se recibe la información necesaria para cargar las imágenes en GuidedActivity
loadImages	Puerto <i>in</i>	Method call	Se recibe la información necesaria para cargar las imágenes en FreeActivity

Tabla 9.9: Puertos de ImagesHandler

- **Zowi:** componente que reúne todo lo relacionado con Zowi, desde su instancia hasta la gestión del envío de comandos.

Interfaz	Tipo	Tecnología	Descripción
sendDataToZowi	Puerto <i>in</i>	Method call	Se recibe el comando a enviar a Zowi desde una actividad
sendCommand	Puerto <i>out</i>	Method call	Se envía el comando a Zowi a través de un socket bluetooth

Tabla 9.10: Puertos de Zowi

9.3.3. Componentes del código Arduino

Con respecto a la arquitectura del código Arduino, **se ha diseñado de forma que siga los estándares de esta plataforma:** un archivo principal en el que destacan un *setup* y un *loop*, desde donde de forma continua se hacen llamadas y peticiones a otros fragmentos de código.

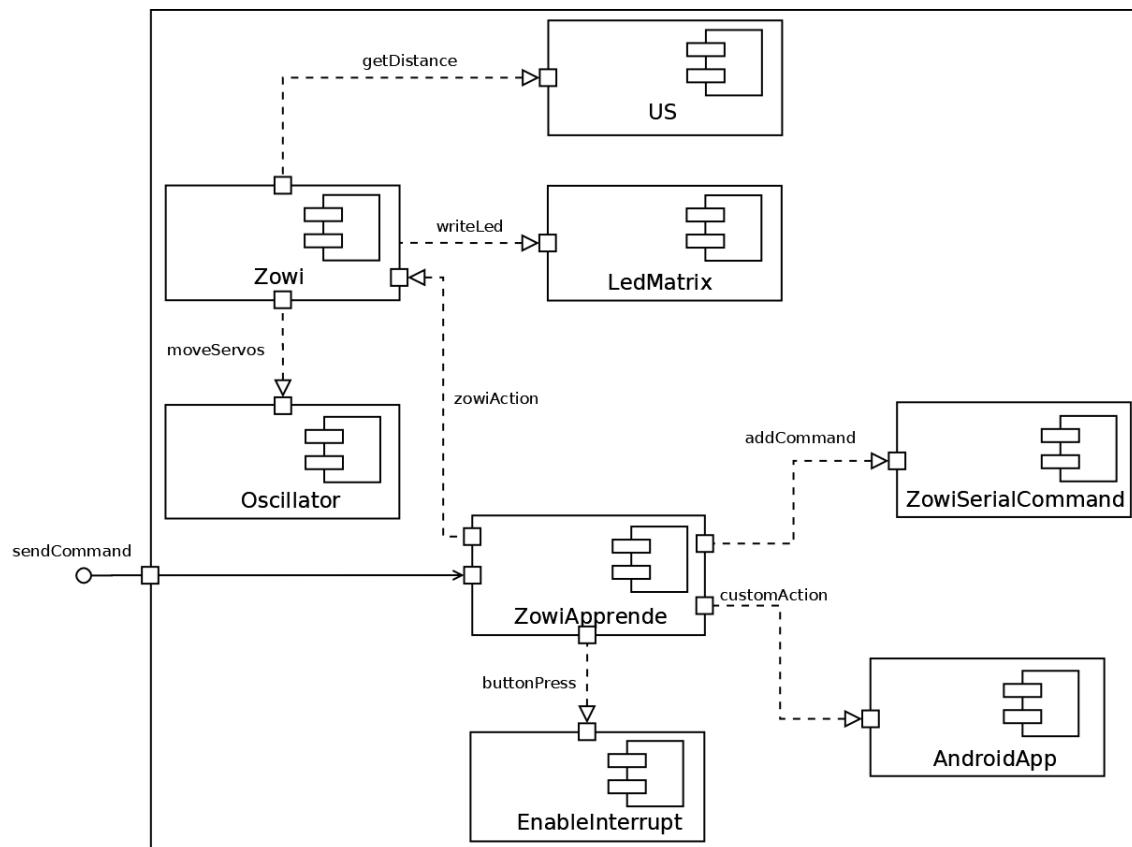


Figura 9.12: Diagrama de componentes del código Arduino.

Componentes

- **ZowiApprende:** archivo principal de Arduino, donde **se ubica el bucle que espera interacciones por parte de la app**.

Interfaz	Tipo	Tecnología	Descripción
sendCommand	Puerto in	Method call	Se recibe un comando a ejecutar de la aplicación Android
zowiAction	Puerto out	Method call	Se llama a alguno de los métodos disponibles en la clase de Zowi
addCommand	Puerto out	Method call	Se hace la llamada para registrar un mapeo nuevo para cada método recibido
buttonPress	Puerto out	Hardware button press	Se pulsa el botón A, el B, o los dos a la vez generando una interrupción
customAction	Puerto out	Method call	Se ejecuta una acción nueva para responder a las nuevas funciones de Zowi

Tabla 9.11: Puertos de ZowiApprende

- **ZowiSerialCommand:** clase encargada de la **gestión del puerto serie**, por donde se reciben los comandos enviados desde Android y se envían los ACK de respuesta.

Interfaz	Tipo	Tecnología	Descripción
addCommand	Puerto <i>in</i>	Method call	Se recibe un comando a mapear y registra una determinada acción para ejecutar cuando se reciba

Tabla 9.12: Puertos de ZowiSerialCommand

- **AndroidApp:** componente destinado a **ejecutar acciones comunes directamente relacionadas con las actividades**, como la corrección de ejercicios.

Interfaz	Tipo	Tecnología	Descripción
customAction	Puerto <i>in</i>	Method call	Se ejecuta alguna de las nuevas acciones implementadas para responder a la aplicación Android

Tabla 9.13: Puertos de AndroidApp

- **EnableInterrupt:** clase cuyo objetivo es **habilitar las interrupciones de los botones físicos** disponibles en Zowi.

Interfaz	Tipo	Tecnología	Descripción
buttonPress	Puerto <i>in</i>	Hardware button press	Se interrumpe el flujo del programa para ejecutar una determinada acción asociada a la pulsación

Tabla 9.14: Puerto de EnableInterrupt

- **Zowi:** clase principal donde **se ejecuta todo el código relacionado con el hardware del robot**, en muchos casos haciendo uso de las librerías disponibles.

Interfaz	Tipo	Tecnología	Descripción
zowiAction	Puerto <i>in</i>	Method call	Se ejecuta la función llamada previamente
writeLed	Puerto <i>out</i>	Method call	Se envía la información sobre los LEDs a encender
getDistance	Puerto <i>out</i>	Method call	Se llama al método para calcular la distancia de un objeto
moveServos	Puerto <i>out</i>	Method call	Se definen los parámetros iniciales y se ejecuta la función del movimiento de servos

Tabla 9.15: Puertos de Zowi

- **Oscillator:** componente que interactúa directamente con los servos de las caderas y los pies del robot.

Interfaz	Tipo	Tecnología	Descripción
moveServos	Puerto <i>in</i>	Method call	Se reciben algunos parámetros, se definen otros y se graban los valores de giro en los servos

Tabla 9.16: Puerto de Oscillator

- **LedMatrix:** librería que accede al hardware de la matriz de LEDs.

Interfaz	Tipo	Tecnología	Descripción
writeLed	Puerto <i>in</i>	Method call	Se escribe el valor especificado en la llamada en la matriz de LEDs

Tabla 9.17: Puerto de LedMatrix

- **US:** clase con la finalidad de recabar información obtenido con el sensor de ultrasonidos.

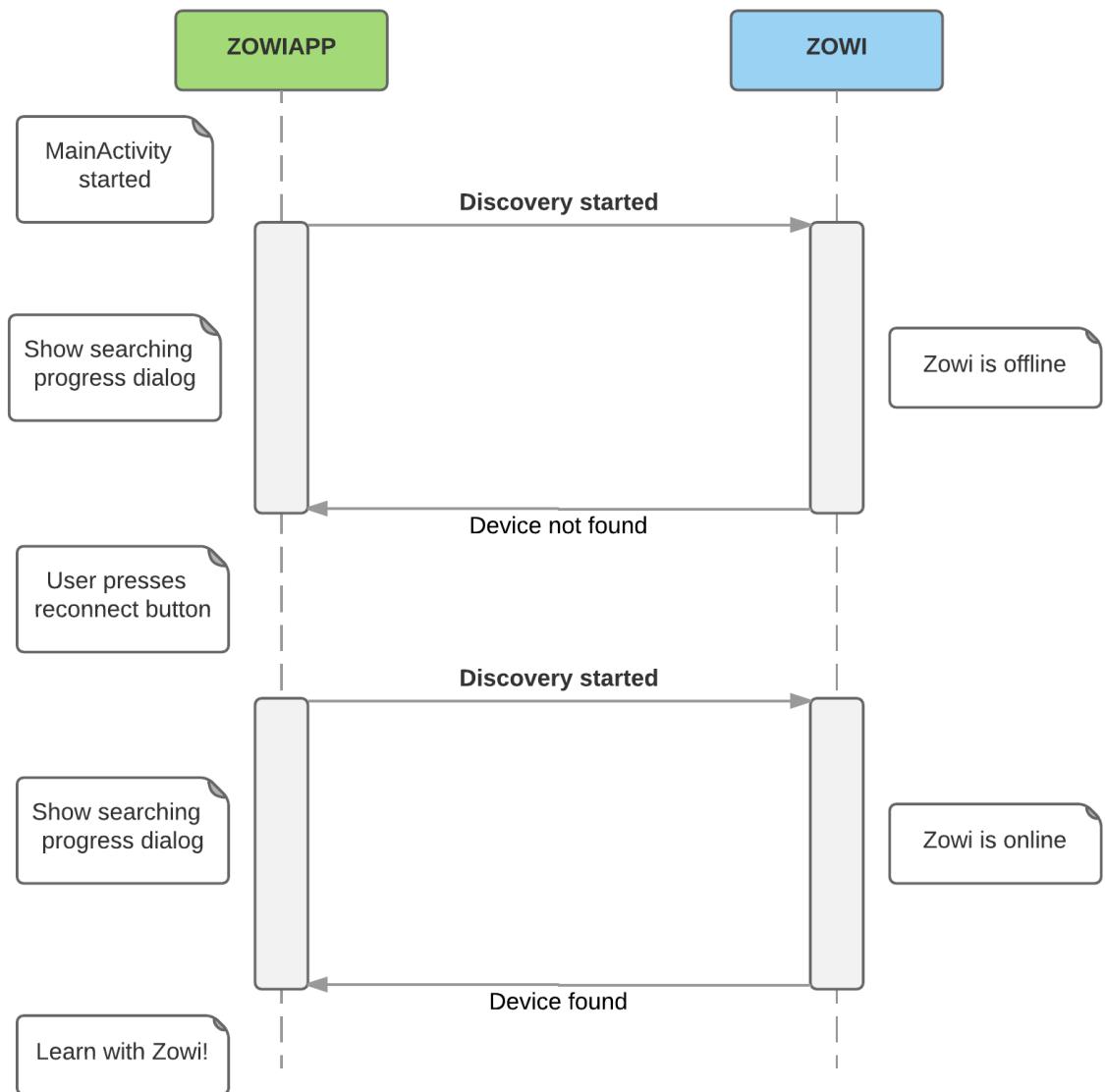
Interfaz	Tipo	Tecnología	Descripción
getDistance	Puerto <i>in</i>	Method call	Se obtiene la distancia medida por el sensor de ultrasonidos

Tabla 9.18: Puerto de US

9.4. Diagramas de secuencia

Multitud de operaciones llevadas a cabo en este proyecto implican el **envío y recepción de datos por medio de Bluetooth**. Es así como se consigue tanto el **establecimiento de la conexión con Zowi**, como el **uso de comandos** que posibilitan la interacción de los niños con él.

De este modo, en este apartado se detallan los pasos que se siguen en ZowiApprende para que la comunicación bidireccional tenga lugar con éxito.

Figura 9.13: Conexión con Zowi por medio de un *receiver*.

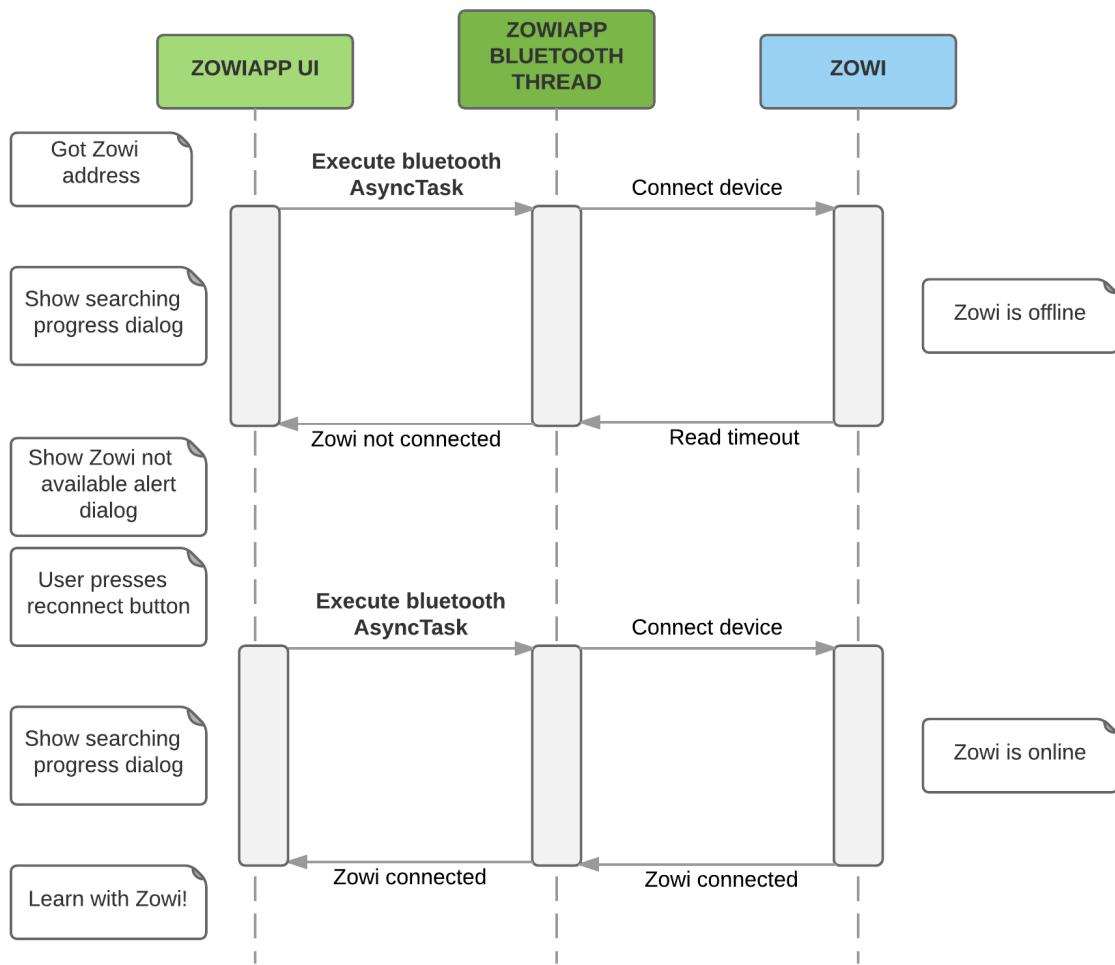


Figura 9.14: Conexión con Zowi una vez almacenada su dirección.

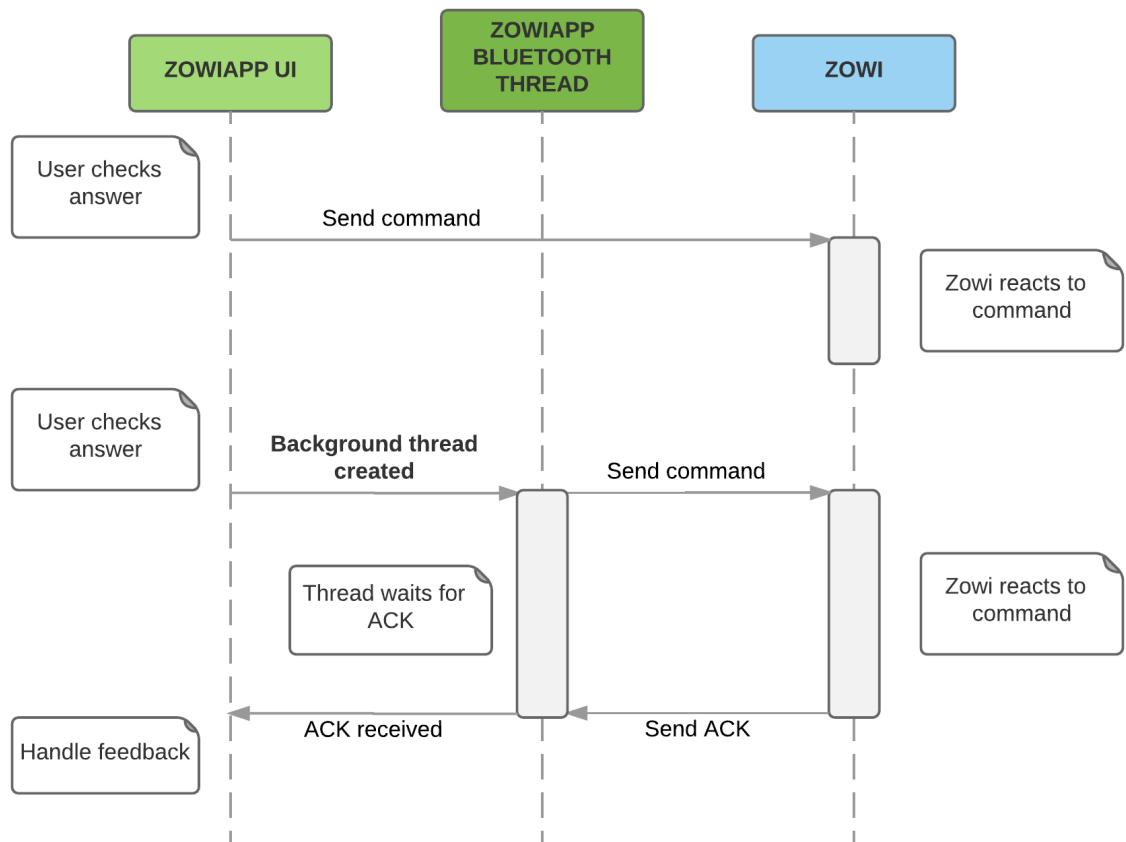


Figura 9.15: Envío de comandos desde ZowiApprende.

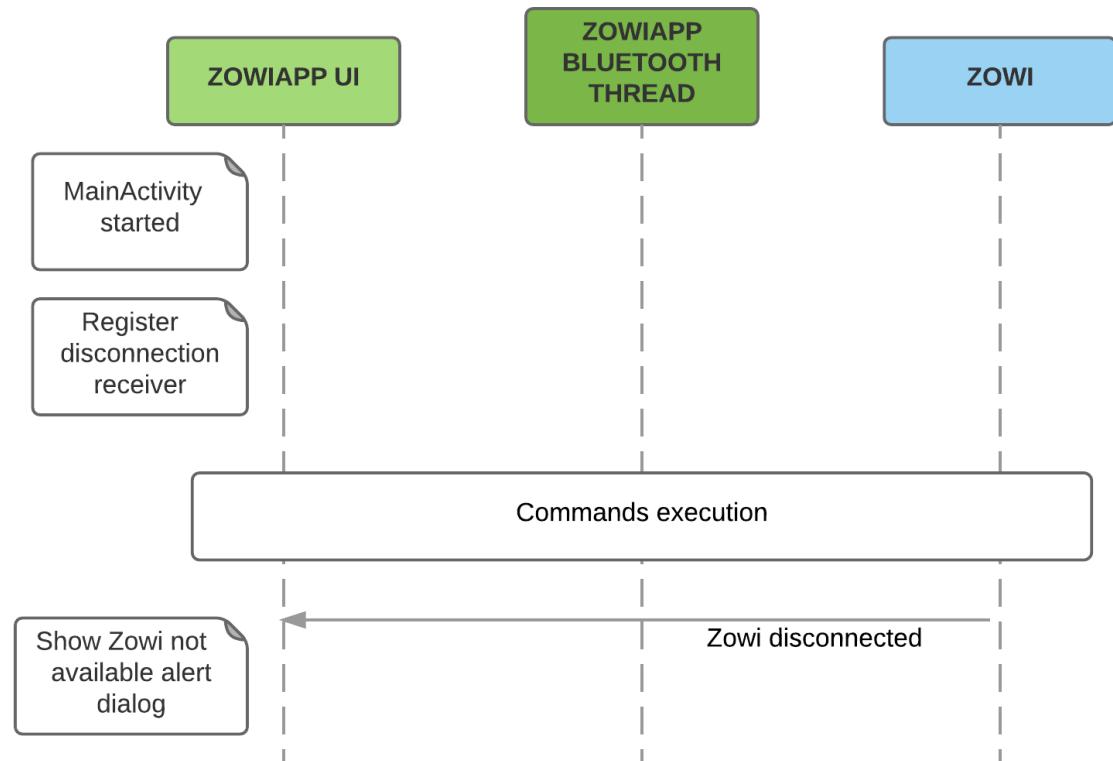


Figura 9.16: Desconexión de Zowi.

9.5. Diagrama de despliegue

En esta sección se detallan los despliegues necesarios para que ZowiApprende funcione correctamente en las aulas. Dada la naturaleza y el alcance del Proyecto, **no se requiere de conexión a Internet ni de la existencia de servidores**, por lo que toda comunicación se queda en el ámbito local. No obstante, es necesaria la **sintonía de los dos sistemas para que el conjunto desempeñe de acuerdo a lo esperado**.

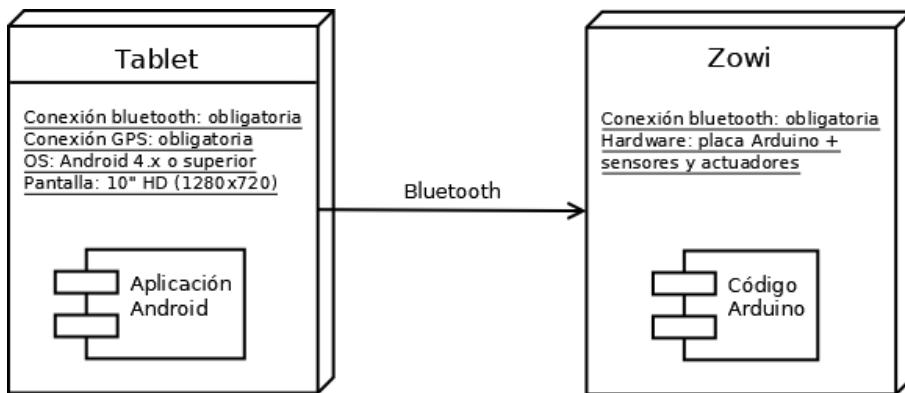


Figura 9.17: Diagrama de despliegue de ZowiApprende.

Como se puede observar en la Figura 9.17, el sistema consta de -como se ha comentado en numerosas ocasiones- dos partes bien diferenciadas: la tablet que usarán los niños, y Zowi.

9.5.1. Tablet

Dispositivo en el que se instala la aplicación Android ZowiApprende, constituyendo uno de los requisitos de este TFM: la maestra que usará el resultado final dispone de una tablet BQ M10 HD, por lo que **el desarrollo se ha enfocado teniendo en cuenta las características de la misma**.

Para empezar, y conformando la base del buen funcionamiento del conjunto, se ha contemplado el **sistema operativo que corre esta tablet: Android 4.4**. Por lo tanto, toda la funcionalidad debe poder ser ejecutada en esta versión obsoleta del software de Google. Aunque se podría hacer uso de Lollipop o Marshmallow, la propia BQ ha confirmado con anterioridad que a este equipo no llegarán más actualizaciones.

Valorando el punto de vista estético, **se ha creado una interfaz adaptada a la densidad de píxeles de la pantalla**, determinante en lo que respecta a la correcta visualización de elementos. Este parámetro se calcula a partir de su tamaño y de su resolución, y varía entre la gran cantidad de dispositivos Android existentes en el mercado.

Dado que ZowiApprende se instala en este hardware, debe disponer de bluetooth para poder establecer comunicación con el robot.

9.5.2. Zowi

El robot de BQ que constituye el eje central de este TFM. **Dado que todo el desarrollo se ha basado en él, no tiene ningún requisito como tal**, sino que se ha aprovechado al máximo su funcionalidad para que los niños puedan hacer uso de la totalidad de sus características.

Cabe destacar la **conexión bluetooth para su manejo, así como los distintos sensores y actuadores** que posibilitan la diversión y el aprendizaje en las aulas de Educación Infantil.

Capítulo 10

Desarrollo de las pruebas

Como se detalló en la sección [Definición del plan de pruebas](#), en este Proyecto se han combinado las pruebas de integración, de sistema y de usabilidad. Se plasmarán, por tanto, distintas tablas donde se recoge la información obtenida, los resultados, las medidas tomadas y comentarios de la maestra que se encargó de usar ZowiApprende.

Cabe destacar, en primer lugar, que **la maestra a quien va destinada esta aplicación no ha podido tomar parte activa en esta etapa del desarrollo**. Su involucración con otros proyectos ha impedido su participación, por lo que se ha optado por **requerir la colaboración de otra maestra de Educación Infantil con conocimientos en robótica**.

En segundo lugar, es necesario especificar los dispositivos en los que se ha basado este apartado: un BQ Aquaris X con Android 7.1, un Samsung Galaxy A9 con Android 6.0.1, y un emulador simulando la Aquaris M10 HD con 4.4 como sistema operativo.

10.1. Pruebas de ZowiApprende

10.1.1. Pruebas del menú principal

Prueba	Acceso a ZowiApprende desde el menú de aplicaciones
Entrada	Se selecciona ZowiApprende en el menú de aplicaciones del dispositivo
Salida esperada	Se abre el menú principal de ZowiApprende
Salida obtenida	Se abre el menú principal de ZowiApprende
Comentarios	Los colores rojo y verde suponen una dificultad en el caso de niños daltónicos
Medidas adoptadas	ColorADD es un sistema de representación que les permite a las personas daltónicas identificar los colores gracias a símbolos. En el menú principal se ha añadido un ícono para cada tipo de juego, tal y como se puede ver en la Figura 10.1

Tabla 10.1: Prueba de acceso al menú principal.



Figura 10.1: Menú principal de ZowiApprende con el código ColorADD.

Prueba	Conexión con Zowi encendido por medio de Bluetooth.
Entrada	Abrir ZowiApprende desde el menú de aplicaciones.
Salida esperada	La aplicación se conecta con el robot automáticamente, mostrando un <i>progress dialog</i> en el proceso.
Salida obtenida	La aplicación se conecta con el robot automáticamente, mostrando un <i>progress dialog</i> en el proceso.
Comentarios	-
Medidas adoptadas	-

Tabla 10.2: Prueba de conexión con Zowi encendido por medio de Bluetooth.

Prueba	Conexión con Zowi apagado por medio de Bluetooth.
Entrada	Abrir ZowiApprende desde el menú de aplicaciones.
Salida esperada	La aplicación muestra una ventana emergente para reintentar la conexión después de mostrar un <i>progress dialog</i> en el durante unos segundos.
Salida obtenida	La aplicación muestra una ventana emergente para reintentar la conexión después de mostrar un <i>progress dialog</i> en el durante unos segundos.
Comentarios	-
Medidas adoptadas	-

Tabla 10.3: Prueba de conexión con Zowi apagado por medio de Bluetooth.

Prueba	Acceso al menú de juego libre
Entrada	Se hace clic en el botón verde de juego libre
Salida esperada	Se abre el menú de juego libre
Salida obtenida	Se abre el menú de juego libre
Comentarios	Las dimensiones son correctas y el fondo es vistoso y acorde al contexto
Medidas adoptadas	-

Tabla 10.4: Prueba de acceso al menú de juego libre.

Prueba	Acceso al menú de juego dirigido
Entrada	Se hace clic en el botón verde de juego dirigido
Salida esperada	Se abre el menú de juego dirigido
Salida obtenida	Se abre el menú de juego dirigido
Comentarios	La estructuración es simple y no añade complejidad en el uso
Medidas adoptadas	-

Tabla 10.5: Prueba de acceso al menú de juego dirigido.

10.1.2. Pruebas de la sección de juego libre

Prueba	Acceso a la actividad de bloques lógicos
Entrada	Se hace clic en el botón correspondiente a la actividad
Salida esperada	Se abre la actividad de bloques lógicos
Salida obtenida	Se abre la actividad de bloques lógicos
Comentarios	Sería aconsejable modificar el texto del enunciado
Medidas adoptadas	Se ha modificado el texto adecuándolo a las recomendaciones de la maestra

Tabla 10.6: Prueba de acceso a la actividad de bloques lógicos.

Prueba	Resolución incorrecta de la actividad de bloques lógicos
Entrada	Se hace girar a Zowi desobedeciendo el enunciado y se corrige la actividad
Salida esperada	Zowi se muestra triste y se permite continuar resolviendo el ejercicio
Salida obtenida	Zowi se muestra triste y se permite continuar resolviendo el ejercicio
Comentarios	Zowi despierta sentimientos de cariño El botón de corrección no proporciona feedback de que ha sido pulsado
Medidas adoptadas	Se ha mejorado el comportamiento de ese y otros botones cuya apariencia no cambiaba al ser pulsados

Tabla 10.7: Prueba de resolución incorrecta de la actividad de bloques lógicos.

Prueba	Resolución correcta de la actividad de bloques lógicos
Entrada	Se hace girar a Zowi de acuerdo a lo especificado en el enunciado y se corrige la actividad
Salida esperada	Zowi se muestra contento y se abre una ventana emergente con las opciones para repetir la actividad, o para volver al menú de juego libre
Salida obtenida	Zowi se muestra contento y se abre una ventana emergente con las opciones para repetir la actividad, o para volver al menú de juego libre
Comentarios	-
Medidas adoptadas	-

Tabla 10.8: Prueba de resolución correcta de la actividad de bloques lógicos.

Prueba	Repetición de la actividad una vez ha sido completada correctamente
Entrada	Se hace clic en el botón “¡Otra vez!”
Salida esperada	Se reinicia la actividad
Salida obtenida	Se reinicia la actividad
Comentarios	-
Medidas adoptadas	-

Tabla 10.9: Prueba de repetición de la actividad de bloques lógicos.

Prueba	Vuelta al menú de juego libre tras haber completado la actividad correctamente
Entrada	Se hace clic en el botón “¡Volver al menú!”
Salida esperada	Se vuelve al menú de juego libre
Salida obtenida	Se vuelve al menú de juego libre
Comentarios	-
Medidas adoptadas	-

Tabla 10.10: Prueba de vuelta al menú de juego libre.

Prueba	Vuelta al menú principal tras haber vuelto al menú de juego libre
Entrada	Se hace clic en el botón “Atrás” de la barra de navegación
Salida esperada	Se vuelve al menú principal
Salida obtenida	Se vuelve a la actividad
Comentarios	-
Medidas adoptadas	Android ha almacenado en el stack las siguientes pantallas, en este orden: menú principal, menú de juego libre, actividad, menú principal. Por lo tanto, pulsar el botón “Atrás” implica volver a la actividad en lugar de al menú principal. No ha habido tiempo para corregir este comportamiento

Tabla 10.11: Prueba de vuelta al menú principal.

10.1.3. Pruebas de la sección de juego dirigido

Prueba	Acceso a las actividades de la cuadrícula
Entrada	Se pulsa alguno de los iconos del menú que lleva a las distintas actividades de la cuadrícula
Salida esperada	Se abre la actividad de la cuadrícula
Salida obtenida	Se abre la actividad de la cuadrícula
Comentarios	Sería recomendable cambiar algunos textos de los enunciados
Medidas adoptadas	Se han modificado los textos de acuerdo a las recomendaciones de la maestra

Tabla 10.12: Prueba de acceso a actividades de la cuadrícula.

Prueba	Selección de movimientos en las actividades de la cuadrícula
Entrada	Se seleccionan movimientos al azar
Salida esperada	Se almacena la dirección en la pila de movimientos y se colorea la siguiente casilla a la que caminaría el robot
Salida obtenida	Se almacena la dirección en la pila de movimientos y se colorea la siguiente casilla a la que caminaría el robot
Comentarios	La dificultad podría ser demasiado elevada para alumnos de Educación Infantil
Medidas adoptadas	-

Tabla 10.13: Prueba de selección de movimientos en actividades de la cuadrícula.

Prueba	Encuentro con obstáculos en las actividades de la cuadrícula
Entrada	Se traza una ruta en cuyo camino haya algún obstáculo
Salida esperada	Se abre una ventana emergente indicando que Zowi ha encontrado un obstáculo
Salida obtenida	Se abre una ventana emergente indicando que Zowi ha encontrado un obstáculo
Comentarios	Sería conveniente modificar el texto de la ventana emergente
Medidas adoptadas	Se ha modificado el texto de la ventana emergente

Tabla 10.14: Prueba de encuentro con obstáculos en actividades de la cuadrícula.

Prueba	Desplazamientos de Zowi fuera de la cuadrícula
Entrada	Se traza una ruta que se salga de la cuadrícula y se corrige la actividad
Salida esperada	En la pila de movimientos se siguen almacenando direcciones aunque teóricamente sean fuera de la cuadrícula Zowi camina hasta llegar al límite y en ese instante se muestra una ventana emergente
Salida obtenida	En la pila de movimientos se siguen almacenando direcciones aunque teóricamente sean fuera de la cuadrícula Zowi camina hasta llegar al límite y en ese instante se muestra una ventana emergente
Comentarios	-
Medidas adoptadas	-

Tabla 10.15: Prueba de desplazamientos fuera de la cuadrícula.

Prueba	Borrado de movimientos previamente establecidos en la pila.
Entrada	Se borran todos los movimientos posibles la pila.
Salida esperada	Se elimina la flecha de la pila de movimientos y se descolorea el cuadrado correspondiente de la cuadrícula. Si no hay más movimientos, se muestra una alerta en pantalla. Si Zowi ya ha recorrido parte de la ruta, esos movimientos no se borran de la pila
Salida obtenida	Se elimina la flecha de la pila de movimientos y se descolorea el cuadrado correspondiente de la cuadrícula. Si no hay más movimientos, se muestra una alerta en pantalla. Si Zowi ya ha recorrido parte de la ruta, esos movimientos no se borran de la pila
Comentarios	-
Medidas adoptadas	-

Tabla 10.16: Prueba de borrado de movimientos de la pila de direcciones.

Prueba	Asignación de elementos a contenedores en las actividades de arrastrar a un contenedor (semillas, momentos de alimentación...).
Entrada	Se arrastran los elementos a su contenedor correspondiente.
Salida esperada	Si la asignación es correcta, Zowi se muestra alegre. Si la asignación es errónea, Zowi se muestra triste. Si se completa la actividad, se muestra una ventana emergente de éxito.
Salida obtenida	Si la asignación es correcta, Zowi se muestra alegre. Si la asignación es errónea, Zowi se muestra triste. Si se completa la actividad, se muestra una ventana emergente de éxito.
Comentarios	-
Medidas adoptadas	-

Tabla 10.17: Prueba de asignación de elementos a contenedores en las actividades de arrastrar a un contenedor.

Prueba	Asignación de acciones a su columna correspondiente en la actividad de columnas a izquierda y derecha.
Entrada	Se arrastran las semillas a las macetas.
Salida esperada	Si la asignación es correcta, Zowi se muestra alegre. Si la asignación es errónea, Zowi se muestra triste. Si se completa la actividad, se muestra una ventana emergente de éxito.
Salida obtenida	Si la asignación es correcta, Zowi se muestra alegre. Si la asignación es errónea, Zowi se muestra triste. Si se completa la actividad, se muestra una ventana emergente de éxito.
Comentarios	-
Medidas adoptadas	-

Tabla 10.18: Prueba de asignación de acciones en la actividad de columnas a izquierda y derecha.

Prueba	Resolución de cuentas en la actividad de operaciones simple.
Entrada	Se resuelven las 6 operaciones disponibles.
Salida esperada	Si la respuesta es correcta, Zowi se muestra alegre y se deshabilita el botón de “Corregir”. Si la respuesta es incorrecta, Zowi se muestra triste. Si se completa la actividad, se muestra una ventana emergente de éxito.
Salida obtenida	Si la respuesta es correcta, Zowi se muestra alegre y se deshabilita el botón de “Corregir”. Si la respuesta es incorrecta, Zowi se muestra triste. Si se completa la actividad, se muestra una ventana emergente de éxito.
Comentarios	-
Medidas adoptadas	-

Tabla 10.19: Prueba de resolución de cuentas en la actividad de operaciones simple.

Prueba	Mostrado de cuentas en la actividad de operaciones compleja.
Entrada	Al apretar el botón “Enseñar”, se muestran los operandos y el operador de la operación en la boca de Zowi.
Salida esperada	La información aparece, de derecha a izquierda, en la boca de Zowi.
Salida obtenida	Sólo aparece la parte inicial de la operación, para a continuación mostrar una boca feliz.
Comentarios	-
Medidas adoptadas	Se comprobó el correcto funcionamiento del algoritmo depurando directamente con Zowi, y se determinó que fallo provenía de la prematura finalización del mismo al recibir información por medio de Bluetooth. Se solucionó el fallo y se pudo probar la actividad con éxito.

Tabla 10.20: Prueba de mostrado de cuentas en la actividad de operaciones compleja.

Prueba	Resolución de cuentas en la actividad de operaciones compleja.
Entrada	Se introducen los valores mostrados en la boca de Zowi y el resultado de la cuenta.
Salida esperada	Si la respuesta es correcta, Zowi muestra un tic en la boca y se deshabilita el botón de “Corregir”. Si la respuesta es incorrecta, Zowi muestra una “x”. Si se completa la actividad, se muestra una ventana emergente de éxito.
Salida obtenida	Sólo aparece la parte inicial de la operación, para a continuación mostrar una boca feliz bloqueándose la interfaz de la aplicación.
Comentarios	El texto “Enseñar” destaca poco en el fondo azul del botón.
Medidas adoptadas	Se modificó el color del fondo para que el contraste con la letra fuera mayor.

Tabla 10.21: Prueba de resolución de cuentas en la actividad de operaciones compleja.

Prueba	Corrección de la actividad de la pirámide alimenticia.
Entrada	Una vez arrastrados los alimentos a su hueco, se pulsa el botón “Corregir” y se muestra la tablet a Zowi para que este “corrija” la actividad.
Salida esperada	Al situar la tablet delante de los ojos del robot, la actividad se corrige.
Salida obtenida	Al situar la tablet delante de los ojos del robot, la actividad se corrige.
Comentarios	Parece que es Zowi el que corrige realmente el ejercicio, por lo que los niños pueden disfrutar mucho de esta interacción
Medidas adoptadas	-

Tabla 10.22: Prueba de corrección de la actividad de la pirámide alimenticia.

Prueba	Resolución de la actividad de la pirámide alimenticia.
Entrada	Se arrasta cada alimento a su hueco correspondiente, corrigiendo en distintas etapas y alternando respuestas correctas con incorrectas hasta la completa resolución.
Salida esperada	Si la respuesta es correcta pero no se ha completado la pirámide, Zowi se muestra triste. Si la respuesta es incorrecta, Zowi se muestra triste y el alimento se traslada a su posición original. Si se completa la actividad, se muestra una ventana emergente de éxito.
Salida obtenida	Si la respuesta es correcta pero no se ha completado la pirámide, Zowi se muestra triste. Si la respuesta es incorrecta, Zowi se muestra triste y el alimento se traslada a su posición original. Si se completa la actividad, se muestra una ventana emergente de éxito.
Comentarios	-
Medidas adoptadas	-

Tabla 10.23: Prueba de resolución de la actividad de la pirámide alimenticia.

Prueba	Aleatoriedad de contenido en la actividad de puzzles.
Entrada	Se entra en repetidas ocasiones en la actividad.
Salida esperada	Tanto la imagen como su forma cambian al entrar varias veces en la actividad.
Salida obtenida	Tanto la imagen como su forma cambian al entrar varias veces en la actividad.
Comentarios	La aleatoriedad amplía la vida útil del ejercicio.
Medidas adoptadas	-

Tabla 10.24: Prueba de aleatoriedad de contenido en la actividad de puzzles.

Prueba	Escalado de imágenes en la actividad de puzzles.
Entrada	Se mantiene pulsada cada pieza del puzzle para corroborar que escala hasta su posición adecuada (aquella que le permite encajar en el recuadro).
Salida esperada	Las imágenes crecen lo necesario como para encajar en el recuadro.
Salida obtenida	Las imágenes crecen lo necesario como para encajar en el recuadro.
Comentarios	-
Medidas adoptadas	-

Tabla 10.25: Prueba de escalado de imágenes en la actividad de puzzles.

Prueba	Resolución de la actividad de puzzles.
Entrada	Se arrastran las piezas a su correspondiente posición en la superficie del puzzle.
Salida esperada	Si la resolución es correcta, Zowi se muestra contento y aparece una ventana emergente de éxito. Si la resolución es incorrecta, Zowi se muestra triste y se puede continuar resolviendo el puzzle .
Salida obtenida	Tanto la imagen como su forma cambian al entrar varias veces en la actividad.
Comentarios	-
Medidas adoptadas	-

Tabla 10.26: Prueba de resolución de la actividad de puzzles.

Prueba	Resolución de la actividad de memory.
Entrada	Se arrastran las piezas a su correspondiente posición en la superficie del puzzle.
Salida esperada	<p>Si la resolución es correcta, Zowi se muestra contento y aparece una ventana emergente de éxito.</p> <p>Si la resolución es incorrecta, Zowi se muestra triste y se puede continuar buscando las parejas.</p> <p>Si se completa la actividad, se muestra una ventana emergente de éxito.</p>
Salida obtenida	Tanto la imagen como su forma cambian al entrar varias veces en la actividad.
Comentarios	La actividad es bastante compleja para los alumnos de Educación Infantil
Medidas adoptadas	-

Tabla 10.27: Prueba de resolución de la actividad de memory.

Prueba	Resolución de la actividad de memory.
Entrada	Se arrastran las piezas a su correspondiente posición en la superficie del puzzle.
Salida esperada	<p>Si la resolución es correcta, Zowi se muestra contento y aparece una ventana emergente de éxito.</p> <p>Si la resolución es incorrecta, Zowi se muestra triste y se puede continuar buscando las parejas.</p> <p>Si se completa la actividad, se muestra una ventana emergente de éxito.</p>
Salida obtenida	<p>Si la resolución es correcta, Zowi se muestra contento y aparece una ventana emergente de éxito.</p> <p>Si la resolución es incorrecta, Zowi se muestra triste y se puede continuar buscando las parejas.</p> <p>Si se completa la actividad, se muestra una ventana emergente de éxito.</p>
Comentarios	La actividad es bastante compleja para los alumnos de Educación Infantil.
Medidas adoptadas	-

Tabla 10.28: Prueba de resolución de la actividad de memory.

Prueba	Funcionamiento de la actividad de la lupa.
Entrada	Se arrasta el dedo por la pantalla para descubrir todas las imágenes disponibles.
Salida esperada	Zowi emite pitidos a distinta frecuencia y período en función de la cercanía de una foto.
Salida obtenida	Zowi emite pitidos a distinta frecuencia y período en función de la cercanía de una foto.
Comentarios	La actividad es muy llamativa visualmente
Medidas adoptadas	-

Tabla 10.29: Prueba de funcionamiento de la actividad de la lupa.

Prueba	Mostrado de casa en la actividad de guía.
Entrada	Al iniciar la actividad, se sitúa la tablet delante de Zowi.
Salida esperada	El robot indica cuál es su casa difuminando el resto hasta que casi no sean perceptibles. Se muestra una ventana emergente con instrucciones para hacer que Zowi se desplace.
Salida obtenida	El robot indica cuál es su casa difuminando el resto hasta que casi no sean perceptibles. Se muestra una ventana emergente con instrucciones para hacer que Zowi se desplace.
Comentarios	-
Medidas adoptadas	-

Tabla 10.30: Prueba de mostrado de casa en la actividad de guía.

Prueba	Guía de Zowi con el sensor de ultrasonidos en la actividad de guía.
Entrada	Una vez mostrada la casa de Zowi, se pone la mano o un objeto delante para que el robot lo siga.
Salida esperada	Zowi sigue el objeto más rápido cuanto más lejos esté este último.
Salida obtenida	Zowi sigue el objeto más rápido cuanto más lejos esté este último.
Comentarios	-
Medidas adoptadas	-

Tabla 10.31: Prueba de guía de Zowi con el sensor de ultrasonidos en la actividad de guía.

Prueba	Actividad de los dictados musicales.
Entrada	Se pulsa el botón de “Comenzar” para alguno de los 3 dictados musicales.
Salida esperada	Zowi lleva el ritmo marcado en el pentagrama.
Salida obtenida	Zowi lleva el ritmo marcado en el pentagrama.
Comentarios	-
Medidas adoptadas	-

Tabla 10.32: Prueba de la actividad de los dictados musicales.

Prueba	Actividad de la cuadrícula de colores.
Entrada	Se define el número de cuadriculas de cada color.
Salida esperada	Zowi se muestra contento o triste en función de si la respuesta es correcta o no.
Salida obtenida	Zowi se muestra contento o triste en función de si la respuesta es correcta o no.
Comentarios	-
Medidas adoptadas	-

Tabla 10.33: Prueba de la actividad de los cuadrícula de colores.

Capítulo 11

Implementación del sistema

11.1. Alternativas tecnológicas y software utilizado

Aunque a lo largo de este escrito se han detallado los lenguajes de programación y herramientas utilizadas, no se han expuesto los motivos que han llevado a tomar esas decisiones. Y a pesar de que en algunos casos o bien no hubo opción o bien no hubo duda alguna, cabe destacar que **las pautas seguidas en este Proyecto no son las únicas posibles ni viables**.

11.1.1. Sistema operativo

Los motivos por los que se puede llegar a apostar por Android en un proyecto son muy variados. Se podría justificar esta decisión basándose, por ejemplo, en la **cuota de mercado del sistema de Google**, que domina con amplia ventaja en lo que a porcentaje de dispositivos se refiere. Además, la inmensa mayoría de estos últimos se ofrecen con un **precio de venta muy inferior a su rival más directo, el iPhone**.

No obstante, y dado que la maestra que usará ZowiApprende dispone de una tablet **BQ M10**, en ningún momento hubo dudas con respecto a cuál sería la opción escogida al respecto.

En lo que respecta a Zowi, el hecho de albergar una plataforma Arduino eliminó por completo la toma de decisiones al respecto.

11.1.2. Lenguaje de programación

A pesar de que en multitud de ocasiones Android es directamente asociado con **Java**, este **no es el único lenguaje de programación en el que se puede desarrollar una aplicación**.

La que es posiblemente la alternativa más interesante a día de hoy es **Kotlin**, lenguaje oficial de Android -junto con el ya comentado- desde el anuncio de Google en el I/O de 2017. Con una curva de aprendizaje más sencilla y la misma potencia de Java, constituye una opción más moderna y permite aprovechar cualquier fragmento de código ya existente.

Sin embargo, el desconocimiento de Kotlin, la falta de oficialidad hasta una etapa tardía y al experiencia previo decantaron la balanza a favor del lenguaje más tradicional.

11.1.3. IDE

En el caso de Android, se hizo uso del IDE oficial proporcionado por la empresa de la gran G: Android Studio. Se había hecho uso de Eclipse con anterioridad, pero de nuevo, **la oficialidad del primero y los problemas de rendimiento del segundo dejaron la elección en bandeja.**

El caso de Arduino fue razonablemente más complicado, al ser la alternativa oficial, el Arduino IDE, extremadamente simple y carecer de funcionalidad básica en caso de haberse acostumbrado a programas como el IntelliJ IDEA o el Visual Studio. Resulta razonablemente parecido, en lo que a experiencia de uso se refiere, al Notepad++ o similares.

Por lo tanto, y tras evaluar el plugin de Arduino para Eclipse, **se optó por utilizar el editor de texto Atom junto con el paquete PlatformIO**. En conjunto, proporcionan toda la funcionalidad necesaria y aconsejable para que la tarea de desarrollar se lleve a cabo de forma satisfactoria.

11.1.4. Otro software

La finalización de ZowiApprende se ha basado en el software mencionado, pero también **se ha hecho uso de multitud de programas que han contribuido a mejorar la calidad general del resultado.**

Creación de esquemas y mockups

Tras la búsqueda de la mejor opción posible por Internet, se optó por las alternativas expuestas a continuación:

- **Mockups:** dado que los bocetos debían representar una aplicación Android, **se descargó e instaló WireframeSketcher**. Cuenta por defecto con multitud de componentes adecuados a este sistema operativo y la facilidad de uso es notable desde el primer momento.
- **Esquemas:** **la mayoría se han elaborado con Dia**, un software gratuito que ofrece bastante funcionalidad. Sin embargo, y teniendo en cuenta que se han encontrado algunas limitaciones en el transcurso del desarrollo, también se ha hecho uso de las aplicaciones online [Lucidchart](#) -versión gratuita- y [websequencediagrams](#).

Creación y edición de imágenes

Para la edición de las imágenes mostradas en este escrito o en la propia aplicación, **se utilizó el programa gratuito GIMP**. Constituye una alternativa realmente potente y libre a Photoshop, y cumple con creces para la inmensa mayoría de tareas diarias.

Esta funcionalidad se ha complementado con Inkscape, un editor profesional de vectores gráficos que ha permitido crear algunas imágenes de cero -tarea para la que no está pensada GIMP-. Además, con protagonismo en la sección correspondiente a [al algoritmo para caminar de Zowi](#), **se aprovechó la potencia de Autocad para el diseño de los esquemas angulares** mostrados en las diversas representaciones.

Control de versiones

Con el objetivo de mantener un histórico y salvaguardar el código, **se ha hecho uso de Git y del repositorio GitHub** desde el propio nacimiento del Proyecto. La mejora y evolución del código de ambos subsistemas se puede comprobar en esta página: github.com/marcojonudo.

11.2. Manual de instalación

Dados los objetivos marcados al comienzo del proyecto, que se pueden ver en la sección **Objetivos y alcance**, **no se pretendía subir la aplicación al Google Play Store**. Por lo tanto, y centrándose en el estado actual del Proyecto, **se debe utilizar un método de instalación distinto**, cuyos pasos se detallan más adelante.

El procedimiento, a grandes rasgos, **consiste habilitar la instalación de aplicaciones externas** -no disponibles en la Store-. De esta forma, y habiendo descargado previamente el archivo de instalación de ZowiApprende, es posible llevar completar el proceso y utilizarla como si se hubiera obtenido a través de los medios habituales.

Cabe destacar que para seguir con éxito el tutorial expuesto **es requisito indispensable contar con un dispositivo con Android como sistema operativo**. Además, si bien es posible instalar ZowiApprende cumpliendo esta condición, **el uso de la misma y la conexión con Zowi estarán limitados por la disponibilidad de este robot**.

1. Descarga de los archivos necesarios

El primer paso consiste en la **obtención del archivo apk**. Este formato es a Android lo que el exe a Windows: constituye un fichero cuya ejecución desemboca en la instalación de un determinado software.

Aunque en los anexos proporcionados en la entrega de los documentos del TFM se puede encontrar ZowiApprende.apk, también se puede descargar desde el siguiente enlace:

https://drive.google.com/open?id=1A_vURE3xkN_5zkjMDCckLdL0HYmufnHg

2. Habilitación de la instalación de aplicaciones de terceros en Android

El sistema operativo de Google no permite por defecto la instalación de archivos apk. No obstante, se puede modificar este comportamiento desde los ajustes del dispositivo.

Para ello, y siguiendo el flujo mostrado en la Figura 11.1, **se debe acceder al apartado “Seguridad” y hacer clic encima de “Orígenes desconocidos”**.

Aceptando las condiciones mostradas en la ventana emergente, **el dispositivo queda listo para continuar el proceso**.

3. Transferencia del archivo apk al dispositivo

En caso de que la descarga no se haya llevado a cabo con el propio terminal, **se debe copiar la aplicación (en formato apk) desde el dispositivo de origen a aquel en el que se pretenda instalar ZowiApprende**.

Se puede copiar en cualquier directorio.

4. Instalación del apk

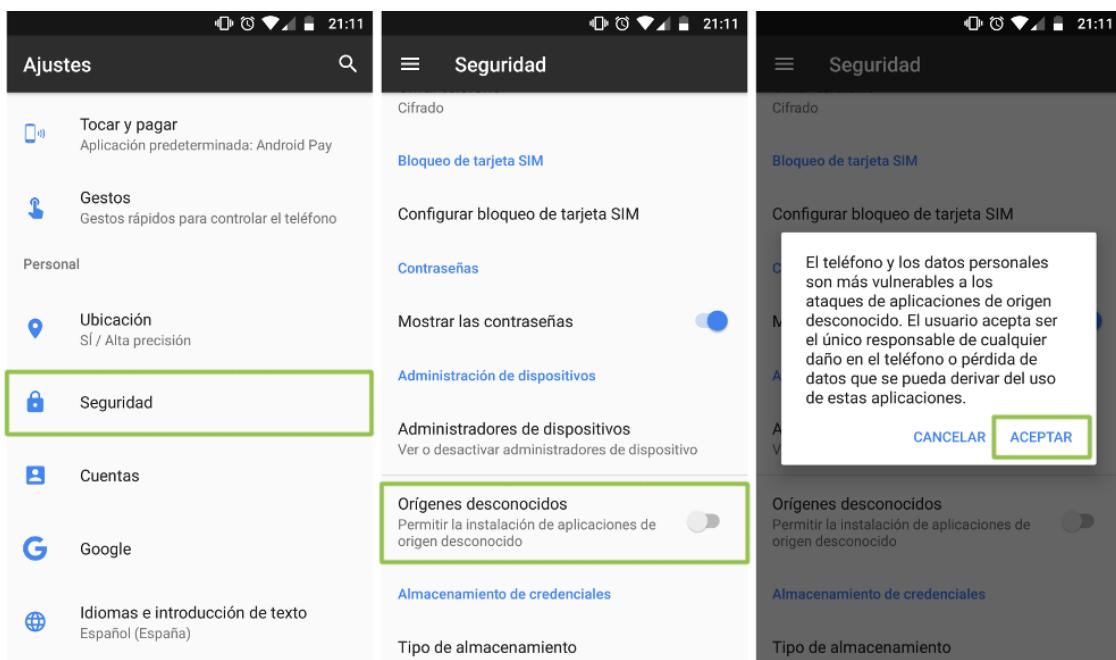


Figura 11.1: Habilitación de los “Orígenes desconocidos” en Android.

Una vez se encuentre el apk en la memoria del dispositivo en cuestión, se debe acceder con el explorador de archivos de preferencia (por ejemplo, [Es File Explorer](#) o [Solid Explorer File Manager](#)). En el caso expuesto en la Figura11.2, se ha usado el disponible por defecto en el sistema operativo.

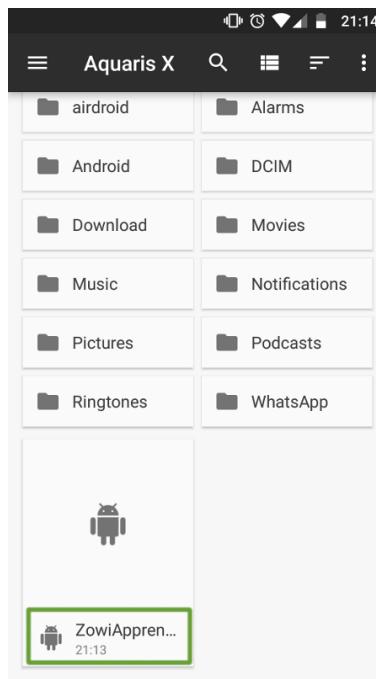


Figura 11.2: “ZowiApprende.apk” en un explorador de archivos.

Presionando encima del archivo, se abre la ventana de instalación de aplicaciones. Los últimos pasos para completar con éxito todo el procedimiento consisten en seguir la sucesión de capturas mostradas en la Figura 11.3.

En el último paso se puede pulsar el botón “Listo” y acceder a ZowiApprende desde el menú de aplicaciones, pero la opción “Abrir” permite hacerlo inmediatamente.

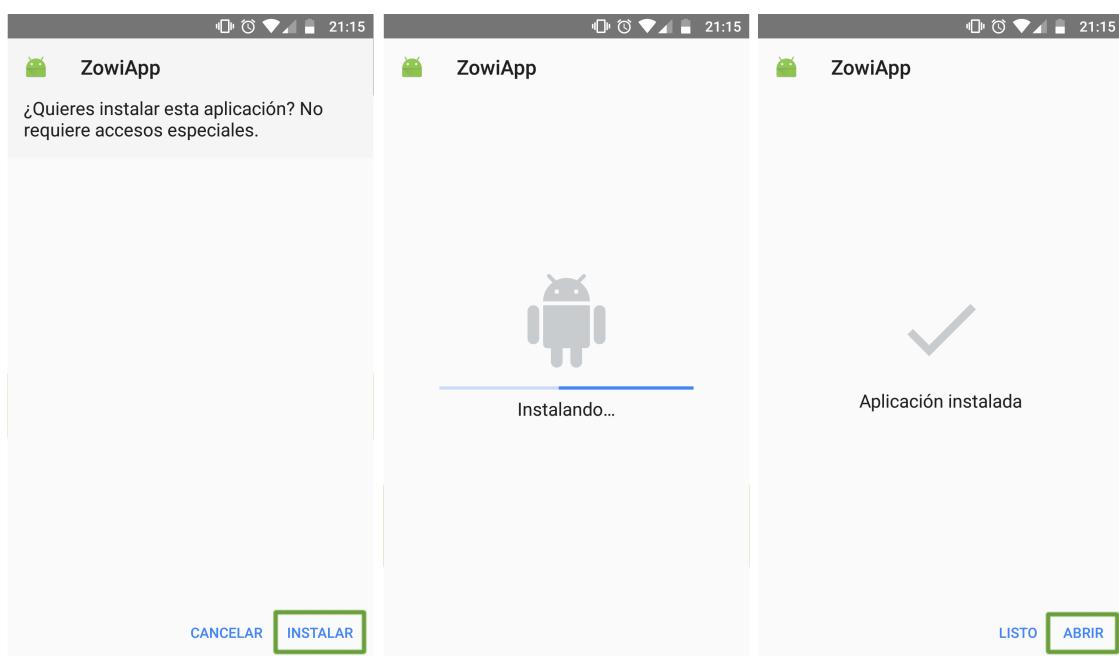


Figura 11.3: Instalación de una aplicación externa al Google Play Store.

11.3. Problemas y obstáculos enfrentados

Como cualquier otro tipo de Proyecto con un objetivo definido, **a lo largo del desarrollo se ha enfrentado una serie de problemas** a los que o se ha dado solución, o se han sorteado o definido como irresolubles. A continuación **se comenta cada uno de ellos más detalladamente**, con el fin de analizar cuál ha sido la respuesta y evaluar cómo se puede mejorar en el futuro.

11.3.1. Carga de contenido

Como se ha explicado con anterioridad, y tras analizar e idear lo que se convertiría más adelante en ZowiApprende, **se persiguió que el contenido de la aplicación fuera lo menos limitado posible** -tanto en variedad de actividades como en contenido-.

Por lo tanto, se optó por el enfoque final, relativamente similar al de un framework. **Cambiar los textos e imágenes es tarea sencilla al separar la presentación de la estructura interna**, de modo que aumentar o modificar aspectos de contenido es rápido y sencillo. Además, en lugar de hacer uso de fotos o cadenas simples, **se hizo uso de arrays y de un índice aleatorio que se genera al iniciar un determinado ejercicio**. De este modo, es plausible garantizar un ciclo de vida lo más largo posible.

Este proceso se consigue por medio de un fichero XML, donde está definida la información de cada actividad. Una parte representativa de su contenido se puede ver a continuación:

```

1 <string-array name="guided_activities_details">
2     <!-- TEMA 1 -->
3     <item>{"type": "GRID", "description": \"Traza el camino y llévalo hasta la
4         casilla musical\", "gridSize": 1, "cells": [7, 2], "images":
5             [grid_zowi_pointer, grid_musical_cell], "externalLink":
6                 \"https://www.youtube.com/watch?v=gKkrMgGzN7E\"}</item>
```

```

7   <item>{"type": "GRID", "description": \"Llévalo a comer. ¡Que no se entretega  

8     jugando ni descansando!\", "gridSize": 1, "cells": [9, 8, 2, 1], "images":  

9       [grid_zowi_pointer, grid_toy, unit_4_bed, grid_meal], "externalLink": \"\"}<!--item--&gt;<br/>
10  <item>{"type": "GRID", "description": \"Lleva a Zowi a darse un baño, ¡y ten  

11    cuidado con el jabón!\", "gridSize": 1, "cells": [8, 5, 2], "images":  

12      [grid_zowi_pointer, grid_soap, grid_bath], "externalLink": \"\"}<!--item--&gt;<br/>
13  <item>{"type": "SEEDS", "description": \"Planta las semillas en sus hoyos  

14    correspondientes\", "seedsImages": [[unit_1_seeds_seed_yellow,  

15      unit_1_seeds_seed_yellow, unit_1_seeds_seed_yellow, unit_1_seeds_seed_yellow],  

16      [unit_1_seeds_seed_orange, unit_1_seeds_seed_orange, unit_1_seeds_seed_orange,  

17        unit_1_seeds_seed_orange], [unit_1_seeds_seed_pink, unit_1_seeds_seed_pink,  

18          unit_1_seeds_seed_pink, unit_1_seeds_seed_pink], [unit_1_seeds_seed_brown,  

19            unit_1_seeds_seed_brown, unit_1_seeds_seed_brown, unit_1_seeds_seed_brown]],  

20      "containerImages": [unit_1_seeds_container_yellow, unit_1_seeds_container_orange,  

21        unit_1_seeds_container_pink, unit_1_seeds_container_brown], "correction":  

22          [YELLOW, ORANGE, PINK, BROWN]}<!--item--&gt;<br/>
23  <item>{"type": "COLUMNS", "description": \"Coloca en cada columna lo que debemos  

24    hacer antes y después de comer!\", "leftTitle": \"Antes\",  

25      "rightTitle": \"Después\", "images": [[unit_1_columns_before_set_table,  

26        unit_1_columns_before_cook, unit_1_columns_before_wash_hands],  

27        [unit_1_columns_after_brush_teeth, unit_1_columns_after_clean_table,  

28          unit_1_columns_after_washing_up]], "correction": [LEFT, RIGHT]}<!--item--&gt;<br/>
29  <item>{"type": "OPERATIONS", "description": \"Resuelve las sumas y restas y  

30    consigue todos los dientes que faltan!\", "operationsType": 1, "image":  

31      "unit_1_operations_zowi_teeth", "operationsImages": []}<!--item--&gt;<br/>
32  

33  ...  

34  

35  <!-- TEMA 5 -->  

36  <item>{"type": "GRID", "description": \"Llévalo a la casilla musical!\",  

37      "gridSize": 2, "cells": [15, 5], "images": [grid_zowi_pointer,  

38        grid_musical_cell], "externalLink":  

39          \"https://www.youtube.com/watch?v=501DDBnHrCw\"}<!--item--&gt;<br/>
40  <item>{"type": "MUSIC", "description": \"Toca el instrumento siguiendo el dictado  

41    musical!\", "images": [unit_5_music_dictation_1, unit_5_music_dictation_2,  

42      unit_5_music_dictation_3, unit_5_music_dictation_4, unit_5_music_dictation_5]}  

43  </item>  

44  <item>{"type": "COLOURED_GRID", "description": \"Lleva a Zowi al colegio. Elige  

45    el camino más corto!\", "cells": [[2, 13], [1, 10], [8, 11]], "images":  

46      [unit_5_coloured_grid_destiny, grid_zowi_pointer], "colouredCells":  

47          [[GREEN, GREEN, BLANK, RED, GREEN, BLANK, BLUE, RED, GREEN, BLUE, BLUE, RED,  

48            GREEN, BLANK, RED, RED], [BLUE, BLANK, GREEN, GREEN, BLUE, RED, RED, GREEN,  

49              BLUE, BLUE, BLANK, GREEN, BLANK, BLUE, BLUE, BLANK], [GREEN, GREEN, GREEN,  

50                GREEN, GREEN, BLANK, BLANK, GREEN, BLANK, BLUE, RED, RED, RED,  

51                  RED]]}<!--item--&gt;<br/>
52  <item>{"type": "OPERATIONS", "description": \"Resuelve las sumas y restas con los  

53    objetos del colegio!\", "operationsType": 2, "image":  

54      "unit_1_operations_zowi_teeth", "operationsImages": [unit_5_operations_sharpener,  

55        unit_5_operations_rubber, unit_5_operations_blackboard, unit_5_operations_pencil,  

56        unit_5_operations_scissors, unit_5_operations_glue]}<!--item--&gt;<br/>
57  <item>{"type": "PUZZLE", "description": \"Construye el rompecabezas del colegio!\",  

58      "images": [[unit_5_puzzle_school_1_1, unit_5_puzzle_school_2_1],  

59        [unit_5_puzzle_school_1_2, unit_5_puzzle_school_2_2], [unit_5_puzzle_school_1_3,  

60          unit_5_puzzle_school_2_3]]}<!--item--&gt;<br/>
61  <item>{"type": "COLUMNS", "description": \"Arrastra cada acción a su columna!\",

```

```

62         "leftTitle": \"Lo que debemos hacer en el aula\", "rightTitle": \"Lo que no
63             debemos hacer en el aula\", "images": [[unit_5_columns_clean_up,
64                 unit_5_columns_help_others, unit_5_columns_play, unit_5_columns_queue],
65                 [unit_5_columns_argue, unit_5_columns_shout, unit_5_columns_furious,
66                 unit_5_columns_unfriendly]], "correction": [LEFT, RIGHT]}</item>
67 </string-array>
68
69 <string-array name="free_activities_details">
70     <item>{"type": "LOGIC_BLOCKS", "description": \";Identifica la imagen
71         correspondiente al tema!\", "images": [triangulo_rojo_grande,
72             cuadrado_verde_grande, triangulo_rojo_pequeno, cuadrado_azul_grande,
73             cuadrado_azul_pequeno, cuadrado_verde_pequeno, circulo_amarillo_grande,
74             circulo_amarillo_pequeno]}</item>
75 </string-array>

```

11.3.2. Evento TouchEvent en imágenes

Dado que ZowiApprende se usará en aulas de Educación Infantil, las imágenes son las grandes protagonistas del Proyecto. Muchas de las actividades diseñadas requieren arrastrarlas por la pantalla, por lo que **el correcto desempeño en este aspecto es crucial**.

En Android, **una vista sólo puede desplazarse dentro de la superficie de su padre**. Por lo tanto, distintos ImageViews anidados dentro de un contenedor no podrán rebasar los límites de este último, pudiendo no ser lo suficientemente amplios.

Para solventar este inconveniente, se crearon distintos diseños en XML usando el *layout Constraint*. En la carga de la actividad, **se obtienen las coordenadas de los elementos y se crean vistas ubicadas en la misma posición, pero con un parente distinto**. Este parente constituye la raíz de la jerarquía de elementos, por lo que así es posible utilizar gestos a lo largo de toda la superficie disponible.

11.3.3. Carga de imágenes asíncrona con Picasso

Una versión inicial de ZowiApprende **cargaba las imágenes de forma síncrona**, estableciéndolas como *background* o recurso en función de cada caso particular. A pesar de ser funcional, la aplicación no era eficiente, y **podía generar errores de memoria en caso de que los archivos tuvieran más resolución de la estrictamente necesaria o su número fuera elevado**.

Por lo tanto, y tras evaluar qué librerías existen con este cometido -como Glide-, se implantó Picasso para delegar esta responsabilidad. A pesar de que la primera está recomendada por la propia Google, **generó numerosos problemas al crear conflicto con la etiqueta (*tag*) de las vistas**. Al no depender este aspecto del desarrollador de este TFM sino de los de Glide, se decidió utilizar la segunda mencionada.

Sin embargo, esta refactorización no estuvo exenta de inconvenientes. El uso de **ConstraintLayout permite expandir los elementos del *layout* hasta ocupar todo el espacio disponible**, lo que implica que **en función del dispositivo en el que se ejecute la app, este tendrá un tamaño y relación de aspecto desitintos**. Picasso

carga por defecto los *bitmaps* escalándolos proporcionalmente para ajustarse al lado corto, de forma que mantenga su *ratio* y no se pierda información.

Este hecho implica que, **al no tener los contenedores de imágenes un tamaño predefinido**, siempre sobresalgan parte del ImageView frente al propio *bitmap*. Si no se define *background*, la repercusión visual es nula, pero **no se puede olvidar que muchos ejercicios se basan en el *drag and drop***. Por lo tanto, los límites del contenedor son determinantes de cara a evaluar una correcta ubicación de los elementos.

Inicialmente, se cargaba un *bitmap* para a continuación redefinir el tamaño del ImageView asociado; **la carga asíncrona, sin embargo, impide la correcta ejecución de este planteamiento**, ya que en el instante de redimensionar el contenedor **todavía no hay una imagen cargada en él**.

La solución fue basarse en las dimensiones originales de la foto y calcular previamente -y no a posteriori- cuál tendrá que ser, una vez se cargue todo el contenido, el tamaño final del contenedor.

Además, fueron de especial interés varios aspectos de la configuración de **Picasso**, que al mostrar las imágenes permitieron optimizar el rendimiento al máximo.

```

1 Picasso.with(context).load(resourceId).fit()
2     .centerInside().into(imageView);
3
4 Picasso.with(gameParameters).load(resourceId).fit()
5     .centerInside().memoryPolicy(MemoryPolicy.NO_CACHE).into(imageView);

```

Por defecto (primera instrucción), **la librería guarda en caché información de los recursos**, evitando tiempos de espera en su aparición si estos se usan de forma recurrente. Es la opción utilizada al implementar esta característica inicialmente, pero **guardar estos datos para todas y cada una de las actividades** (cuando por norma general, no se van a abrir una cantidad exagerada de veces) **resultaba contraproducente**. En adición, el retardo al abrir los *layouts* es mínimo, por lo que se optó por modificar ligeramente esta metodología.

Finalmente, **sólo se cachearon las imágenes visualizadas en los menús**. Constituyen el eje central de la navegabilidad de ZowiApprende, y por lo tanto, se consideraron el único lugar en el que merecía la pena implementar esta funcionalidad.

11.3.4. Configuración de IMUs

Aunque finalmente se determinó que el uso de un IMU para compensar la imprecisión al caminar no era la mejor alternativa, **su investigación e implementación conllevó un considerable porcentaje de las horas dedicadas al proyecto**.

En el caso del MPU6050, la obtención de datos del acelerómetro y del giroscopio resultó sencilla. La configuración que requiere este proceso es mínima, y la recopilación de datos bastante inmediata y fácilmente interpretable. **Los problemas**, no obstante, **tuvieron su origen en el uso del magnetómetro del MPU9050**.

En este sensor se requiere una configuración de registros previa menos comprensible que en el caso anterior, y cuenta con un factor añadido que añade complejidad al sistema: **la calibración**. Y es que si bien esta también se lleva a cabo a la hora de conocer la

aceleración o la velocidad de giro, la medición **del campo magnético requiere unas mecánicas distintas**: mover y girar el IMU en un espacio tridimensional -normalmente, haciendo la forma de un ocho en el aire-. Sin embargo, **a pesar de ejecutar estos movimientos correctamente, no se conseguía recabar unos datos bien calibrados**, desconociendo si este hecho se debía a una mala calibración o a una configuración errónea.

Tras comprar un sensor nuevo, y comprobar que los datos obtenidos distaban de los del adquirido previamente, se consiguió definir unas cifras de calibración que permitían obtener medidas bastante fiables.

No obstante, **el MPU9050 no deja de ser un sensor pensado para Arduino, no para la placa modificada por BQ** que constituye el cerebro de Zowi. De esta forma, no está especialmente adaptado, habiendo tenido que hacer uso de conexiones poco profesionales. Sin embargo, existen alternativas diseñadas para funcionar con la infraestructura disponible, como la que se puede ver en www.elecfeet.com/estore/octopus-9dof-module.html.

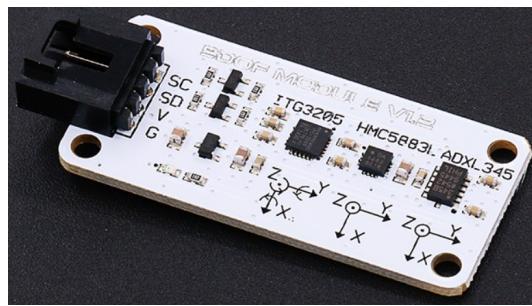


Figura 11.4: IMU 9DOF Octopus.

Este Octopus 9DOF, sin embargo, no constituyó una mejora sustancial. **Dispone de muy poca documentación, y cualquier error encontrado al adaptar el código habría resultado extremadamente costoso de solucionar.**

Por lo tanto, y sumando los inconvenientes detallados en la sección correspondiente al [nuevo algoritmo para caminar](#), se terminó desechar el trabajo realizado. Como idea y concepto, el uso de un IMU resulta muy prometedora, pero no se consideró apropiado incluirlo en este Proyecto dadas las características de Zowi.

11.3.5. Hilos bluetooth

Las primeras aproximaciones al envío de comandos se llevaron a cabo en clases enlazadas a la interfaz de ZowiApprende. Atendiendo a la Figura 9.15, en el primer tipo de comunicación no habría inconvenientes, al enviarse la información y no esperar una respuesta.

En el segundo tipo, no obstante, la aplicación Android espera una respuesta del robot, **bloqueando la interfaz si el envío de datos se realiza desde el lugar inadecuado**. Para evitar este fallo, se decidió **crear un hilo cada vez que se esperara algún feedback de Zowi**.

Una segunda versión de este concepto consistió en **crear una clase manejadora de hilos, de forma que toda la funcionalidad al respecto** -creación, recepción de datos, destrucción, etc- **estuviera centralizada y fuera manejable y escalable**. No obstante, con el fin de evitar tener que proporcionar un feedback desde el hilo a la interfaz (con el

correspondiente fallo al intentar modificar la UI desde el primero), **se esperaba a que su ejecución terminase para continuar el flujo de la actividad.**

```

1 Thread zowiSeeScreenThread = ThreadHandler.createThread(ThreadType.SIMPLE_FEEDBACK);
2 zowiSeeScreenThread.start();
3
4 sendDataToZowi(ZowiActions.ZOWI_CHECKS_ANSWERS);
5
6 try {
7     zowiSeeScreenThread.join();
8
9     //...
10 }
11 catch (InterruptedException e) {
12     e.printStackTrace();
13 }
14 }
```

Esta decisión provocaba el mismo desenlace que el envío de comandos directamente desde la interfaz: **el bloqueo de la misma.**

De este modo, y como solución definitiva, **se implementó un *AsyncTask* intermedio, donde se pudiera esperar a la destrucción del hilo y enviar información al primer plano** de ZowiApprende sin inconvenientes.

```
1 new AsyncTaskHandler(this, ActivityType.GUIDE).execute(ZowiActions.ZOWI_CHECKS_ANSWERS);
```

11.3.6. Personalización de las piezas de Zowi mediante impresión 3D

En los últimos años hemos asistido al **auge de la creación de piezas mediante impresoras 3D**. Como con todas las ramas de la tecnología, este ámbito ha pasado de consistir en planteamientos puramente teóricos a resultar de increíble utilidad en sectores como la sanidad. No sólo se pueden diseñar figuras, ceniceros o logotipos, sino que incluso **es posible la creación de prótesis** cuyo precio sería, de otro modo, desorbitado. En este TFM se ha explorado este potencial, y **se plantean a continuación la experiencia y resultados recabados, así como las dificultades en su consecución.**

Zowi, como se ha comentado en la sección [Carcasas 3D](#), **admite el uso de piezas 3D personalizadas**. En la Figura 3.5 se pueden ver tres alternativas llevadas a cabo por la propia BQ, donde la cabeza del robot se ha personalizado tanto en forma como en color.

Este recurso, no obstante, tiene ciertas limitaciones. **Requiere del acceso a una impresora 3D, así como del conocimiento necesario para realizar el diseño correspondiente.** Estos dos aspectos son restrictivos de por sí, pero el mayor impedimento puede ser la falta de interés por parte del cuerpo de maestros. **La cabeza de Zowi por defecto** tiene un aspecto magnífico, lo que sumado a las pegatinas que trae de fábrica hace que **cumpla con creces en las aulas**.

No obstante, y como se ha plasmado anteriormente haciendo referencia al concepto de este Proyecto, es interesante **darle un giro de tuerca al conjunto para poder explotar todo su potencial**. En este caso, y dado que se ha llevado a cabo una ampliación de

sensores, se decidió diseñar una carcasa que se amoldara mejor a los nuevos integrantes del sistema. Consiste en tres modificaciones con respecto al diseño base:

- **“Orejas” para luces LED:** para dotar a Zowi de luz, se incorporaron dos diodos LED controlables por código, que se ilumina en algunos puntos concretos de la funcionalidad añadida. Por lo tanto, se optó por **aumentar los huecos disponibles en los laterales de la cabeza y realizar en ellos tanto un agujero como unos soportes**. El objetivo fue evitar que estos componentes bailaran en el interior del robot una vez enchufados en la placa base.



Figura 11.5: Nuevas orejas de Zowi.

- **Gorra:** en la sección correspondiente al [sensor de infrarrojos](#), se mencionó la idea llevada a cabo con respecto a la gorra. La imagen se puede ver en la Figura 11.7, y muestra la nueva estructura y la disposición de los sensores.

En la parte inferior de la visera se sitúa el espejo, que **refleja la señal del emisor al suelo y de vuelta al receptor**.

Con respecto al IMU, cuando se pensaba que esa idea era viable **se pretendía imprimir una pieza apropiada para su correcta sujeción**. Inicialmente, las pruebas se llevaron a cabo pegando el MPU9250 en la parte inferior de la superficie de la cabeza de Zowi -paralelo al suelo-, pero esta no es una opción profesional para presentar de cara al público. Por lo tanto, se ideó un **soporte que permitiera colocar el sensor de forma más segura y sin depender tanto de adhesivos** que se pueden despegar con el tiempo, confirmando además la **ausencia de movimiento del dispositivo con respecto al robot**. No obstante, dado que las investigaciones al respecto no llegaron a buen puerto, **se desechó este diseño por resultar inútil**.

En definitiva, **las variaciones con respecto a la carcasa original han proporcionado todo lo necesario para poder adaptar los nuevos componentes a la perfección**. El razonamiento es análogo al expuesto en otras secciones: esta ampliación no es necesaria, pero muy útil en determinadas circunstancias y que vuelve a dejar patente el potencial de este robot.

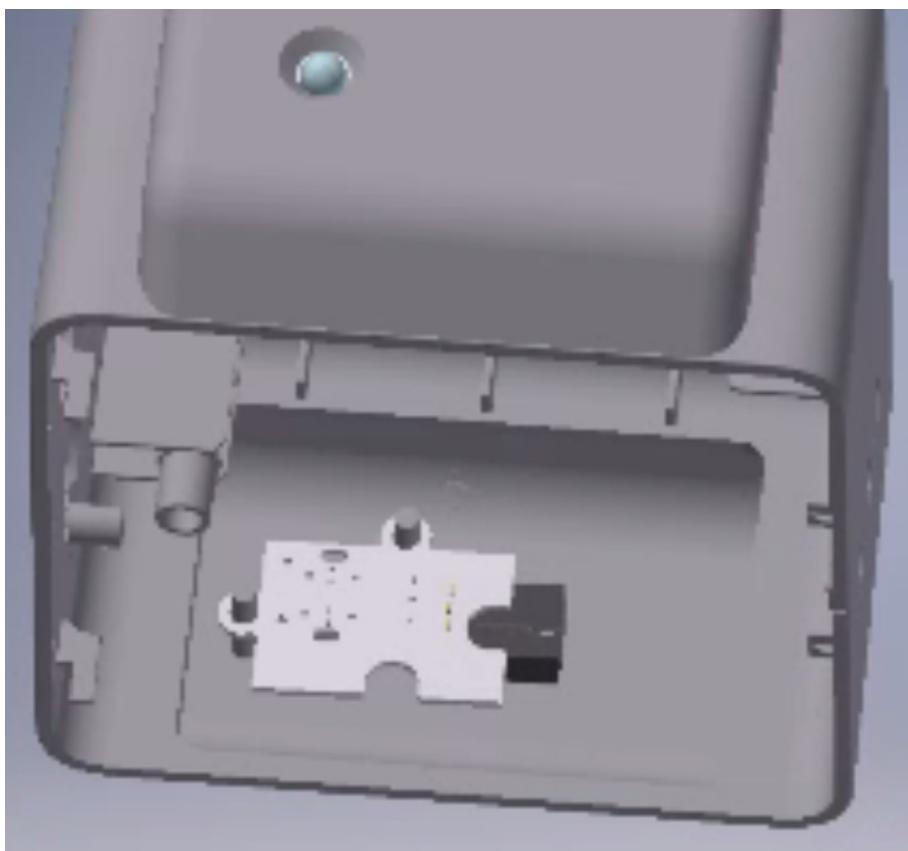


Figura 11.6: Soporte para el sensor dentro de las orejas de Zowi.

11.3.7. Adaptación a requisitos de edades tan tempranas

Un desarrollador acostumbrado a trabajar para un público o clientes adultos puede adquirir ciertas costumbres y buenas prácticas qué, dado el contexto de este Proyecto, no son aplicables en este TFM.

Los niños constituyen un conjunto muy particular en cuanto a necesidades, requisitos, hábitos y pautas de conducta, lo que requiere un **cambio en las aplicaciones enfocadas a ellos**. Acciones que a priori pueden resultar intuitivas, podrían implicar demasiada dificultad para la etapa de Educación Infantil, englobando en este concepto aspectos tanto de funcionalidad como de interfaz.

En algunas ocasiones, se tomaron decisiones desde el punto de vista del **programador**, sin valorar de antemano qué repercusión podrían tener en los usuarios de ZowiApprende. **La mayoría de ellas han sido solventadas con el tiempo**, analizando y evaluando distintas alternativas con un objetivo común; no obstante, **la falta de trabajo previo con personas de esta edad hizo que en algunos casos, se desconociera cuál podría ser la mejor forma de gestionar un determinado procedimiento**.

Las pruebas en colegio habrían sido de gran ayuda, pero dada la imposibilidad de haberlas realizado, se optó por **colaborar estrechamente con una maestra del ámbito correcto para adecuar todas las actividades lo máximo posible**.

11.3.8. Mejora del algoritmo de caminar

La mejora de la forma de caminar de Zowi ha constituido, sin lugar a dudas, la tarea a la que más tiempo se ha dedicado, combinando **investigación y pruebas**.

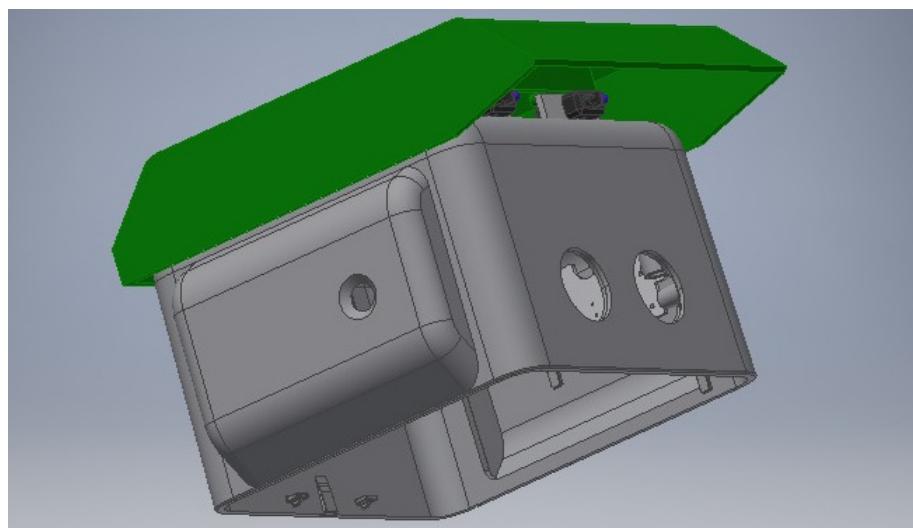


Figura 11.7: Gorra y disposición de los sensores de infrarrojos.

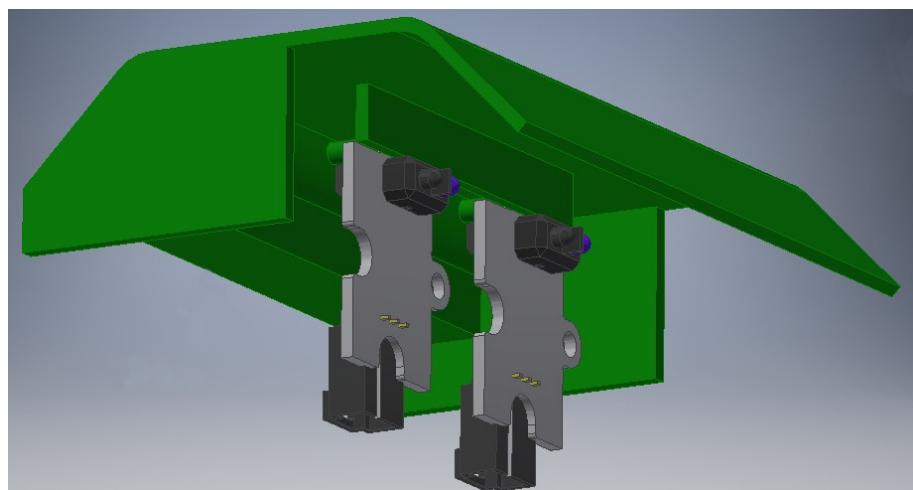


Figura 11.8: Gorra y disposición de los sensores de infrarrojos.

Como se ha mencionado en repetidas ocasiones, **el robot tiende a girarse al caminar**. Este hecho, unido a la falta de retroalimentación del algoritmo -el movimiento de los servos se basa en funciones y valores predefinidos-, provoca que sea **imposible controlar su posición una vez ha iniciado el movimiento**.

Las alternativas propuestas se han analizado en la sección [Nuevo algoritmo para caminar](#), por lo que no se entrará en detalle al respecto. No obstante, cabe destacar algunos aspectos que se han evaluado a lo largo de los meses de desarrollo y que merece la pena plasmar en este apartado.

La pregunta a contestar, a la vista de la falta de resultados conseguida, sería la siguiente: **¿por qué no se ha podido encontrar una solución tecnológica viable?** La respuesta se ha de aclarar en varios puntos:

- **Falta de conocimientos de la plataforma Arduino:** al ser esta la primera experiencia con este ecosistema, el comienzo del Proyecto estuvo marcado por la **carenza en experiencia al respecto**.

La adaptación a la filosofía de Arduino requirió cierto tiempo, y lo más costoso resultó ser el descubrimiento de sus límites; es decir, **conocer qué se puede y no**

se puede hacer. De este modo, en un primer momento no se pensó en el uso de un sensor inercial, ya que ni siquiera se planteaba esta posibilidad.

Muchas de las decisiones que en su día implicaron semanas o meses de investigación y aprendizaje, no requerirían actualmente más de unos días. Aunque habría sido ideal disponer de más tiempo, haber tenido más conocimientos o haber conocido a un experto en la materia, se puede concluir que **se consiguió el fin último de la educación: aprender en el proceso.**

- **Ausencia de un entorno de pruebas adecuado:** muchas de las pruebas llevadas a cabo consistieron en el **uso de sensores pensados para Arduino -no para la placa de BQ-**, con protoboards, soldaduras u otros sistemas relativamente precarios. Y si bien se pueden conseguir avances, **se ha echado en falta un entorno más amigable para llevar a cabo toda clase de tests.**

Realmente, esta dificultad tiene sentido, ya que no se trata de programación web o de alguna aplicación únicamente software. **El hecho de interactuar con hardware implica tener que realizar trabajos más manuales que el desarrollo como tal**, siendo necesaria la compra de sensores, su adecuación a la placa, etc.

- **Estructura de fábrica limitada:** los componentes de Zowi, dado su precio, **no conforman lo mejor en lo que a precisión respecta.** De esta forma, en la práctica los servos no giran siempre la misma cifra, lo que sumado al inconveniente de que el robot resbala en el suelo dificulta enormemente esta tarea.

La posibilidad de modificar la cabeza mediante impresión 3D tiene mucho potencial, pero es necesario tener acceso a uno de estos productos y requiere una inversión considerable de tiempo y diseño.

Este punto, no obstante, tiene relación con el anterior. En el caso del espejo y del sensor de infrarrojos, **se efectuaron pruebas** antes de la instalación final de la nueva cabeza, **pero no tan fiables como en el sistema una vez montado.** Se puede jugar con un espejo casero y con el componente enchufado a la placa base, pero **hasta que no se dispone de una carcasa con los huecos necesarios no es posible determinar la viabilidad del conjunto**, lo que supone una clara limitación.

En conclusión, **el no haber finalizado con éxito este apartado constituye el punto, a título personal, más decepcionante del Proyecto.** Es cierto que Zowi presenta limitaciones prácticamente insalvables, pero a pesar de eso, **las propuestas del IMU y del sensor de infrarrojos con el espejo se consideraban razonablemente sencillas y viables;** los problemas derivados de Arduino y otros factores influyeron negativamente en el correcto desempeño de las mismas.

Sin embargo, cabe destacar que **se considera que ambas ideas tienen potencial, y la realización de los objetivos factible** en caso de aplicar el suficiente tiempo a tomar medidas, invertir en mejores sensores y aumentar la eficiencia de los algoritmos.

11.4. Objetivos no alcanzados

En esta sección se detallan **aquellos objetivos que no han podido ser cumplidos**, así como los motivos que lo impidieron.

11.4.1. Mejora del algoritmo de caminar

Por los motivos expuestos un poco más arriba, en [Mejora del algoritmo de caminar](#), la consecución de este requisito no ha llegado a buen puerto.

No obstante, se ha implementado la actividad de la cuadrícula utilizando un mecanismo mucho más sencillo ya descrito, por lo que este punto no se considera completamente inalcanzado.

Capítulo 12

Conclusiones

12.1. Posibles mejoras

El hecho de haber finalizado el Proyecto no implica que no puedan existir **ampliaciones que lo mejoren**; de este modo, tras evaluar y analizar el entorno de Zowi, Android y todo lo que ambos engloban, se puede determinar **una serie de factores que podrían constituir la continuación del presente TFM**.

12.1.1. Más actividades de juego libre

Dado que **la mayor parte del tiempo se empleó tanto en la investigación del nuevo método para caminar**, como en la creación de actividades de juego dirigido -más importantes-, **no ha sido posible diseñar muchos ejercicios de la sección complementaria**.

Existen **multitud de propuestas que habrían enriquecido la experiencia final en este aspecto**, como distintas ideas con los bloques lógicos, el uso de la regleta de Cousinet o la aplicación del ábaco.

12.1.2. Uso de imágenes libres

En la mayoría de trabajos elaborados hasta la fecha, la totalidad de imágenes no creadas habían seguido el mismo proceso: buscar por Internet, escoger y descargar. Sin embargo, este flujo constituye un **problema si, en el caso de una aplicación o algún otro tipo de software, esas imágenes tienen licencia y su uso no es gratuito**.

A tal fin, existen **fotos que o bien no tienen un propietario reconocido, o bien se publican bajo alguna licencia que permite su aprovechamiento siempre y cuando se identifique la fuente**. En el caso de ZowiApprende, se han intentado explotar ambas alternativas, con resultados poco satisfactorios.

El desarrollo de este Proyecto requiere multitud de distintas imágenes que se adecuen al contexto de la temática. Y si encontrar alguna que cumpla los requisitos mencionados es considerablemente más complicado de lo habitual, **encontrar todas las incluidas en el TFM se consideró un desperdicio de recursos**.

De este modo, **se combinan con fines no lucrativos imágenes con y sin licencia**, pudiendo por tanto existir problemas de cara a la comercialización de la aplicación y a su publicación en el Google Play Store.

12.1.3. Optimización en la carga y manejo de imágenes

Tras utilizar las funciones nativas de Android para la carga de imágenes, **se optó por el uso de una librería externa**; la interfaz no sólo presentaba un comportamiento poco fluido, sino que no se efectuaba una gestión de caché y algunos detalles importantes se llevaban a cabo manualmente de forma innecesaria.

Esta decisión implicó una mejora sustancial a nivel general, aunque aún así se podría intentar una mejora al respecto. **No se tiene un control preciso de lo que Picasso hace en segundo plano**, y dadas la resolución, el tamaño y los formatos de las imágenes -png o jpg-, **se considera que el funcionamiento general de ZowiApprende podría ser mejor**.

Sería interesante analizar **cómo se comporta el sistema en lo que respecta a porcentaje de memoria o procesador consumidos**, haciendo una comparativa para comprobar a nivel teórico cómo y cuánto mejora la aplicación usando una librería.

12.1.4. Registros de la actividad de los alumnos y uso de perfiles de usuario

Evaluar el rendimiento de los pequeños de la clase en la resolución de actividades puede permitir **recavar información de mucho provecho para los maestros**. Los tiempos de finalización, el porcentaje de aciertos o el número necesario de movimientos son algunos de los parámetros que se podrían obtener gracias a un sistema de logging.

El uso de perfiles de usuario está estrechamente ligado a este concepto, ya que **los registros se podrían asociar a cada uno de los niños de la clase**. De este modo, se podrían consultar a posteriori en una aplicación web -entre otras muchas alternativas- especialmente diseñada para tal fin.

Dado que esta ampliación se considera muy interesante, **se han diseñado los mockups** que constituirían la base de la misma.



Figura 12.1: Pantalla de selección de usuario de ZowiApprende.

En la Figura 12.1 se muestra la pantalla de selección del alumno, ofreciendo la de la

Figura 12.3 la opción de trabajar de forma conjunta.



Figura 12.2: Pantalla de selección de más usuarios.

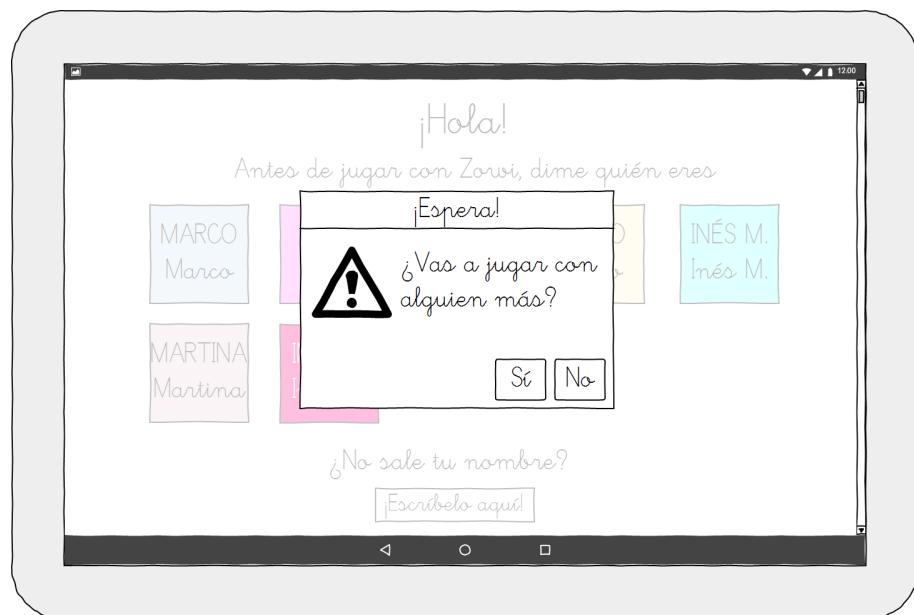


Figura 12.3: Popup para añadir más usuarios.

12.1.5. Soporte a sistemas operativos modernos

Android 4.4 KitKat sigue siendo utilizado, pero su extensión a día de hoy ya constituye un porcentaje anecdótico del total. Y este hecho no es el más importante: Lollipop, Marshmallow, Nougat y ahora Oreo proporcionan mejoras a nivel estético, funcional y de seguridad.

La tablet BQ M10 HD ha quedado obsoleta en lo que respecta al software, y **sería tremendamente aconsejable dar un soporte adecuado a las versiones mayoritarias a día de hoy.**

12.1.6. Pruebas en colegios

Uno de los aspectos fundamentales en la etapa final de cualquier Proyecto consiste en evaluar su desempeño con los propios usuarios. Independientemente del ámbito en el que este se desarrolle, su calidad no va a ir ligada a la opinión del programador, sino a la persona que haga uso del resultado en el día a día.

Dado que el alcance de este TFM no englobaba incluir en la fase de test a los propios niños en las aulas de Educación Infantil, se considera un siguiente paso lógico y acorde al desarrollo llevado a cabo previamente. Su feedback podría proporcionar ideas directas de mejora, retroalimentando el sistema para adaptarse mejor a lo que se espera de él.

Si este paso hubiera estado definido dentro de los objetivos, habría sido recomendable efectuar no una visita al colegio para probar el resultado final, sino varias. De este modo, habría sido posible enfocar el futuro a corto y medio plazo de ZowiApprende en función de las opiniones recogidas a lo largo del tiempo de trabajo.

12.1.7. Subida de ZowiApprende a la Play Store

Ligado a los dos puntos anteriores, estar disponible en la tienda de aplicaciones de Android supone un requisito indispensable para conseguir que un determinado software llegue a la mayoría de gente posible.

Este paso es posterior a todos los demás, ya que implica el fin de la etapa de desarrollo planteada hasta el momento. Posteriormente, y una vez se disponga de una base de usuarios aportando feedback, comenzaría la etapa de soporte. En ella, además de implementar las mejoras comentadas en sucesivas actualizaciones, se solucionarían los bugs o problemas de compatibilidad derivados del uso en multitud de dispositivos.

12.2. Reflexión final

Zowi es un robot educativo con potencial. Cuenta con funciones muy interesantes a las que se puede sacar mucho partido, como el sensor de ultrasonidos, el micrófono o la matriz de LEDs que hace la función de boca; sin embargo, las sensaciones después de comprarlo, analizarlo tanto por fuera como por dentro y hacer ampliaciones de sensores son agridulces.

No se puede olvidar que en la inmensa mayoría de casos, Zowi va a ser adquirido para su uso exclusivo con la aplicación de BQ y sus funciones de fábrica. No todos los usuarios tienen conocimientos de programación ni tiempo para aplicarlos, más aún cuando la mayoría de ellos pertenecen a un sector como el educativo, donde el desarrollo de software no es ni una prioridad ni una necesidad.

Por lo tanto, se corre el riesgo de que este robot acabe en una estantería habiendo sido una anécdota temporalmente divertida e, indudablemente, demasiado cara para lo que llegó a ofrecer en las aulas. Este razonamiento se ha visto confirmado en la práctica gracias al caso expuesto por la maestra de Gijón: los niños habían llegado, tras las etapas de descubrimiento y juego, al aburrimiento y el desinterés. Se pueden hacer aquí dos divisiones importantes:

- **Hardware de Zowi:** sin duda, el punto más positivo del conjunto. La estética del robot provoca una primera impresión inmejorable, motivando la empatía y el cariño de los alumnos hacia él. Su constitución similar a la humana, sus ojos y su boca parecidos a los nuestros, sumado a la cantidad de gestos y sonidos disponibles, hacen que **más pronto que tarde se gane el favor de los niños.**

Los componentes internos, por supuesto, también son importantes, pero no se puede olvidar que su contexto de uso son las aulas de Educación Infantil. En esta etapa priman aspectos que más adelante pueden quedar relegados, y si bien el aspecto externo puede interesar poco a un desarrollador experimentado, es crucial cuando los usuarios son personas que rondan los 4 ó 5 años.

Si a este hecho se le suma la **gran cantidad de pegatinas disponibles en la caja** para personalizar la cabeza del robot, y la **posibilidad de crear piezas completamente nuevas con impresión 3D**, el aspecto exterior se sitúa como el punto más favorable del conjunto.

- **Software de Zowi:** en contraposición a lo expuesto anteriormente, **el software es donde Zowi necesita mejorar si quiere posicionarse como una alternativa real en la educación.** Las funciones por defecto son numerosas y su código base es muy interesante, pero se hace evidente una falta total de utilidad.

¿De qué sirve implementar 20 gestos distintos, si los niños descubren 10 y se aburren? ¿De qué sirve poder controlar los pasos del robot desde un Android, si esta acción no implica nada más allá del simple hecho de poder hacerlo? **Si de verdad se pretende que este robot sea utilizado por niños para aprender, se necesita un software que lo posibilite**, un software que les permite aprender números, letras, hábitos, valores y todo aquello que consideren los especialistas en esta materia.

Mejorar la funcionalidad por defecto depende, a día de hoy, de programadores externos con tiempo y ganas de darle un giro de tuerca al sistema de fábrica. Que esta opción esté disponible es fantástico, pero esta responsabilidad no debería recaer únicamente en personas ajenas a BQ. El hecho de que se requiera no sólo modificar el código Arduino, sino también diseñar actividades docentes y desarrollar una aplicación Android, no hace sino aumentar los obstáculos para que la acogida de Zowi en las aulas esté a la altura de su objetivo: enseñar.

Las carencias expuestas son lo que se ha pretendido solucionar con la nueva ZowiApprende, así como con la investigación y las ampliaciones expuestas en las secciones correspondientes. No se trata únicamente de entender el código, mejorarlo y centrarse en el proyecto técnico como un fin en sí mismo. **Se trata de que los objetivos son enseñar y educar.** En la mayoría de ámbitos, entre ellos el social y el laboral, el concepto de la tecnología como medio y no como fin se aplica constantemente, si bien se tiene tan asumido que no se piensa en ello como tal. Se usa la tecnología, se usan los ordenadores, se usan los móviles, pero a veces cuesta diferenciar cuándo son ellos los que llevan la batuta y cuándo somos nosotros.

Si contextualizamos en el **ámbito de la enseñanza y la educación**, todo este planteamiento cobra más importancia todavía. **Se trata de una de las primeras etapas**

por las que todas las personas tienen que pasar. Todos aquellos que más adelante van a usar la tecnología en lo referente a lo social o al trabajo van a ser educados, de forma que las enseñanzas en estas edades tan tempranas van a repercutir en todo lo demás. Los años de educación obligatoria son el factor común en todos nosotros, y utilizar la tecnología como medio para mejorar puede repercutir directamente en su desarrollo. **No sólo porque se puede aprender más, sino sobre todo porque se pude hacer mejor, de forma innovadora, original y más divertida.** Y aprender divirtiéndose -o divertirse aprendiendo- no requiere esfuerzo.

Bibliografía

- [1] “Arduino”, [en línea]. Disponible en la web: <https://www.arduino.cc/>
- [2] “Robótica BQ”, [en línea]. Disponible en la web: <https://www.bq.com/es/mundo-maker>
- [3] “Robot Zowi”, [en línea]. Disponible en la web: <https://www.bq.com/es/zowi>
- [4] “DIWO (Do It With Otheres)”, [en línea]. Disponible en la web: <http://diwo.bq.com/product/zowi/>
- [5] “Personaliza tu Zowi con diseño e impresión 3D”, [en línea]. Disponible en la web: <http://diwo.bq.com/zowi-personalizar-impresion-3d/>
- [6] “Codi Oruga”, [en línea]. Disponible en la web: <https://juegosrobotica.es/codi-oruga/>
- [7] “Pleo V2 Reborn”, [en línea]. Disponible en la web: <https://www.arduino.cc/>
- [8] “Bee-Bot”, [en línea]. Disponible en la web: <https://www.ro-botica.com/Producto/BEE-BOT/>
- [9] “PDI”, [en línea]. Disponible en la web: https://es.wikipedia.org/wiki/Pizarra_interactiva
- [10] “ScratchJr”, [en línea]. Disponible en la web: <https://www.scratchjr.org/>
- [11] “Bitbloq”, [en línea]. Disponible en la web: <http://bitbloq.bq.com/>
- [12] “Zowi App”, [en línea]. Disponible en la web: <https://play.google.com/store/apps/details?id=com.bq.zowi&hl=es>
- [13] “Geekmomprojects”, [en línea]. Disponible en la web: <http://www.geekmomprojects.com/gyroscopes-and-accelerometers-on-a-chip/>
- [14] “Johnwilliamrussell”, [en línea]. Disponible en la web: <https://johnwilliamrussell.wordpress.com/>
- [15] “GitHub Kriswiner”, [en línea]. Disponible en la web: <https://github.com/kriswiner/MPU-6050/wiki/Affordable-9-DoF-Sensor-Fusion>
- [16] “luisllamas”, [en línea]. Disponible en la web: <http://www.luisllamas.es/2014/09/entradas-analogicas-en-arduino/>

- [17] “Openwebinars”, [en línea]. Disponible en la web:
<https://openwebinars.net/tutorial-arduino-entradas-analogicas-y-digitales/>
- [18] “Cplusplus”, [en línea]. Disponible en la web:
<http://wwwcplusplus.com/doc/tutorial/pointers/>
- [19] “Instructables”, [en línea]. Disponible en la web:
<http://www.instructables.com/id/Connect-Arduino-Uno-to-Android-via-Bluetooth/>
- [20] “learn.sparkfun”, [en línea]. Disponible en la web:
<https://learn.sparkfun.com/tutorials/mpu-9250-hookup-guide>
- [21] “engineering.stackexchange”, [en línea]. Disponible en la web:
<http://engineering.stackexchange.com/questions/3348/calculating-pitch-yaw-and-roll-from-mag-acc-and-gyro-data>
- [22] “Todopic”, [en línea]. Disponible en la web:
<http://www.todopic.com.ar/foros/index.php?topic=41981.0>
- [23] “Javighawk”, [en línea]. Disponible en la web:
<https://javighawk.wordpress.com/2015/01/25/testing-the-mpu-9150-imu/>
- [24] “StackOverflow”, [en línea]. Disponible en la web: <https://stackoverflow.com/>
- [25] “National Oceanic and Atmospheric Administration”, [en línea]. Disponible en la web:
<http://www.ngdc.noaa.gov/geomag-web/igrfwmm>
- [26] “Vectornav”, [en línea]. Disponible en la web:
<http://www.vectornav.com/support/library/magnetometer>
- [27] “Android Developers”, [en línea]. Disponible en la web:
<https://developer.android.com/index.html>
- [28] “Londatiga”, [en línea]. Disponible en la web:
<http://www.londatiga.net/it/programming/android/how-to-programmatically-pair-or-unpair-android-bluetooth-device/>
- [29] “Neteril”, [en línea]. Disponible en la web:
<http://blog.neteril.org/blog/2013/10/10/framelayout-your-best-ui-friend/>
- [30] “Lucidchart”, [en línea]. Disponible en la web: <https://www.lucidchart.com>
- [31] “Google Drive”, [en línea]. Disponible en la web:
https://www.google.com/intl/es_ALL/drive/
- [32] “Dropbox”, [en línea]. Disponible en la web: <https://www.dropbox.com/h>
- [33] Martínez de Carvajal Hedrich, Ernesto Tuneando a Zowi. España, 2016. 978-84-617-7409-8.