

**Universidad Nacional de la Patagonia San Juan Bosco**

**Facultad de Ingeniería**

**Tesina de grado:**

|  |
| --- |
| Desarrollo y construcción de un sistema autónomo robótico administrado por una aplicación web para exploración |

**Alumnos:**

**Mansilla Fernando Damián**

**Schlapp Agustín Pablo**

**Tutor**:

**Lic. Defossé Nahuel**

**Trelew**

**Año 2017**

Índice

[Capítulo 1 - Introducción 8](#_Toc502153545)

[**1.1 Objetivo general** 8](#_Toc502153546)

[1.1.1 Objetivos específicos 8](#_Toc502153547)

[1.1.2 Metodología 8](#_Toc502153548)

[**1.2 Motivación** 9](#_Toc502153549)

[**1.3 Desarrollos Propuestos** 10](#_Toc502153550)

[**1.4 Resultados Esperados** 10](#_Toc502153551)

[Capítulo 2 - La robótica 11](#_Toc502153552)

[**2.1 ¿Qué es la robótica?** 11](#_Toc502153553)

[**2.2 Estructura física de los robots** 13](#_Toc502153554)

[2.2.1 Poliarticulados 13](#_Toc502153555)

[2.2.2 Móviles 13](#_Toc502153556)

[2.2.3 Androides 14](#_Toc502153557)

[2.2.4 Zoomórficos 14](#_Toc502153558)

[2.2.5 Híbridos 14](#_Toc502153559)

[**2.3 Distintas tecnologías para la robótica educativa** 15](#_Toc502153560)

[**2.4 Microcontroladores y computadora de placa reducida (SBC)** 15](#_Toc502153561)

[**2.5. Comunicación entre distintas plataformas de cómputo** 17](#_Toc502153562)

[**2.6 ¿Qué es un SAR (Sistema Autónomo Robótico)?** 18](#_Toc502153563)

[**2.7 La robótica en la educación** 18](#_Toc502153564)

[Capítulo 3 – Arduino 20](#_Toc502153565)

[**3.1 ¿Qué es Arduino?** 20](#_Toc502153566)

[**3.2 ¿Qué es Processing?** 20](#_Toc502153567)

[**3.3** 22](#_Toc502153568)

[**Fritzing** 22](#_Toc502153569)

[**3.4 ¿Qué es Wiring?** 22](#_Toc502153570)

[**3.5 Entonces Arduino es…** 22](#_Toc502153571)

[**3.6 Plataforma Arduino** 23](#_Toc502153572)

[**3.7 Distintas plataformas para Arduino** 24](#_Toc502153573)

[**3.8 Diferencias entre distintas placas de la familia Arduino** 25](#_Toc502153574)

[**3.9 ¿Por qué usar Arduino?** 26](#_Toc502153575)

[3.9.1 La comunidad 26](#_Toc502153576)

[3.9.2 La sencillez del lenguaje de programación 27](#_Toc502153577)

[3.9.3 Es hardware de bajo costo 27](#_Toc502153578)

[**3.10 Incorporación de Arduino en las escuelas** 27](#_Toc502153579)

[3.10.1 Las tres erres 28](#_Toc502153580)

[**3.11 Actuadores y sensores** 28](#_Toc502153581)

[**3.12 Actuadores en el SAR** 29](#_Toc502153582)

[**3.13 Sensores en el SAR** 30](#_Toc502153583)

[**3.14 Módulos o *shields* en el SAR** 31](#_Toc502153584)

[Capítulo 4 – Raspberry Pi 32](#_Toc502153585)

[**4.1 ¿Qué es Raspberry Pi?** 32](#_Toc502153586)

[**4.2 Especificaciones técnicas de las distintas versiones** 32](#_Toc502153587)

[**4.3 Entrada/Salida de propósito general (GPIO)** 33](#_Toc502153588)

[**4.4 Sistemas Operativos compatibles** 34](#_Toc502153589)

[**4.5 Accesorios para Raspberry Pi** 35](#_Toc502153590)

[**4.5 ¿Por qué elegir Raspberry Pi?** 37](#_Toc502153591)

[Capítulo 5 - Aplicaciones Móviles 38](#_Toc502153592)

[**5.1 ¿Qué son las aplicaciones móviles?** 38](#_Toc502153593)

[**5.2 Las App’s** 38](#_Toc502153594)

[5.2.1 Las webApps 38](#_Toc502153595)

[5.2.2 Ventajas de las Web-App: 39](#_Toc502153596)

[5.2.3 Desventajas de las Web-Apps 39](#_Toc502153597)

[**5.3 ¿Cuáles son los distintos sistemas operativos para dispositivos móviles?** 39](#_Toc502153598)

[**5.4 Android** 40](#_Toc502153599)

[**5.5 Aplicaciones móviles multiplataforma** 41](#_Toc502153600)

[5.5.1 Diferencias entre aplicaciones y web móviles 41](#_Toc502153601)

[5.5.2 Aplicaciones Nativas 41](#_Toc502153602)

[5.5.3 Aplicaciones Web 42](#_Toc502153603)

[5.5.4 Aplicaciones Híbridas 43](#_Toc502153604)

[5.5.5 Aplicación híbrida: Web App 43](#_Toc502153605)

[5.5.6 Aplicación híbrida: app interpretada 43](#_Toc502153606)

[**5.6 Investigación en Desarrollo de Apps y WebApps** 44](#_Toc502153607)

[5.6.1 Android Studio 44](#_Toc502153608)

[5.6.2 App inventor 44](#_Toc502153609)

[5.6.3 Intel XDK 45](#_Toc502153610)

[5.6.4 Ionic 45](#_Toc502153611)

[5.6.5 Cordova 45](#_Toc502153612)

[5.6.6 Meteor 45](#_Toc502153613)

[5.6.7 Meteor y Cordova 45](#_Toc502153614)

[Capítulo 6 – Stack MEAN 47](#_Toc502153615)

[**6.1 ¿Qué es MEAN?** 47](#_Toc502153616)

[**6.2 ¿Cómo se compone MEAN?** 47](#_Toc502153617)

[6.2.1 Mongo 47](#_Toc502153618)

[6.2.2 Express 47](#_Toc502153619)

[6.2.3 Angular 47](#_Toc502153620)

[6.2.4 Node 48](#_Toc502153621)

[**6.3 Otros complementos** 48](#_Toc502153622)

[6.3.1 Twitter Bootstrap 48](#_Toc502153623)

[6.3.2 Compodoc 48](#_Toc502153624)

[6.3.3 Json 48](#_Toc502153625)

[Capítulo 7 – Librería Johnny-five y el protocolo Firmata 50](#_Toc502153626)

[**7.1 ¿Qué es Johnny-five?** 50](#_Toc502153627)

[**7.2 Instalación** 50](#_Toc502153628)

[**7.3 Arduino Firmata** 51](#_Toc502153629)

[**7.4 Instalación Firmata** 51](#_Toc502153630)

[Capítulo 8 - Python en Raspberry Pi 53](#_Toc502153631)

[**8.1 ¿Qué es Python?** 53](#_Toc502153632)

[**8.2 Python con Raspberry Pi** 53](#_Toc502153633)

[**8.3 Raspberry PI – GPIO** 54](#_Toc502153634)

[Capítulo 9 - Análisis y selección de tecnologías para desarrollo del SAR 56](#_Toc502153635)

[**9.1 Primer análisis** 56](#_Toc502153636)

[**9.2 Selección tecnologías hardware** 56](#_Toc502153637)

[9.2.1 ¿Por qué Arduino? 56](#_Toc502153638)

[¿Por qué Raspberry? 57](#_Toc502153639)

[**Comparativa entre Arduino Mega, Arduino Nano y Raspberry Pi 3 Model b** 57](#_Toc502153640)

[**Cámara V2 de Raspberry Pi** 58](#_Toc502153641)

[**Módulos de Arduino** 58](#_Toc502153642)

[Capítulo 10 – Ensamblado del robot móvil 60](#_Toc502153643)

[**Componentes** 60](#_Toc502153644)

[**Estructura** 62](#_Toc502153645)

[Capítulo 11 – Desarrollo del SAR 64](#_Toc502153646)

[Esquema general del SAR (lógico /físico) 64](#_Toc502153647)

[Desarrollo de la aplicación (front-end) 64](#_Toc502153648)

[Desarrollo del servidor (back-end) 64](#_Toc502153649)

[Esquema de la arquitectura lógica 65](#_Toc502153650)

[Capturas de pantalla de la aplicación 66](#_Toc502153651)

[Funcionamiento de la App 67](#_Toc502153652)

[Como se construyen las estadísticas 67](#_Toc502153653)

[FALTA DETERMINAR LAS MUESTRAS 67](#_Toc502153654)

[Parámetros de configuración 67](#_Toc502153655)

[**FALTA FINALIZARLA** 67](#_Toc502153656)

[Ilustración 1 - Esquema básico de un robot 11](#_Toc502152469)

[Ilustración 2 - Ejemplo de robot poliarticulado 12](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152470)

[Ilustración 3 - Ejemplo de robot móvil 12](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152471)

[Ilustración 4 - Androide Asimo de Honda 13](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152472)

[Ilustración 5 - Robot Zoomórfico caminador 13](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152473)

[Ilustración 6 - Robot móvil-poliarticulado 13](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152474)

[Ilustración 7 - Arquitectura de un microcontrolador 15](#_Toc502152475)

[Ilustración 8 - Logo Arduino 19](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152476)

[Ilustración 9 - Logo de Processing 19](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152477)

[Ilustración 10 - Wiring IDE 20](#_Toc502152478)

[Ilustración 11 - C++ Blink ejemplo 20](#_Toc502152479)

[Ilustración 12 - Entorno Fritzing 21](#_Toc502152480)

[Ilustración 13 - Ejemplo serie 22](#_Toc502152481)

[Ilustración 14 - Niveles de entrada a la plataforma Arduino 23](#_Toc502152482)

[Ilustración 15 - Arduino Uno 24](#_Toc502152483)

[Ilustración 16 - Logotipo comunidad open-source de Arduino 26](#_Toc502152484)

[Ilustración 17- Representación actuadores y sensores 28](#_Toc502152485)

[Ilustración 18 - Actuadores y sensores compatibles con Arduino 29](#_Toc502152486)

[Ilustración 19- Representación de sensores 30](#_Toc502152487)

[Ilustración 20 - Logo oficial de Raspberry Pi 31](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152488)

[Ilustración 21 - Raspberry Pi 2 y sus GPIOs 32](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152489)

[Ilustración 22 - Interfaces de Raspberry Pi 33](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152490)

[Ilustración 23 - Cámara Raspberry Pi V2 34](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152491)

[Ilustración 24 - Pantalla táctil de Raspberry Pi 34](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152492)

[Ilustración 25 - Adafruit Prototyping Pi 35](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152493)

[Ilustración 26 - Pidrive 35](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152494)

[Ilustración 27 - Pi TFT 35](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152495)

[Ilustración 28 - Aplicaciones móviles 37](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152496)

[Ilustración 29 - App nativa vs Web App 37](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152497)

[Ilustración 30 – WebApps – Diseño multipropósito 38](#_Toc502152498)

[Ilustración 31 - Arquitectura de Android 39](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152499)

[Ilustración 32 - Logo de Android 39](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152500)

[Ilustración 33 - Cuadro comparativo nativas 41](#_Toc502152501)

[Ilustración 34 - Cuadro comparativo - Aplicaciones Web 41](#_Toc502152502)

[Ilustración 35 - Comparativa aplicaciones híbridas 42](#_Toc502152503)

[Ilustración 36 - Herramientas para desarrollo de apps 43](#_Toc502152504)

[Ilustración 37 - Logo del motor V8 47](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152505)

[Ilustración 38 - Arquitectura de interacción MEAN 47](#_Toc502152506)

[Ilustración 39 - Logo de JSON 48](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152507)

[Ilustración 40 - JSON pegamento de tecnologias 48](#_Toc502152508)

[Ilustración 41 - Página oficial de Johnny-Five 49](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152509)

[Ilustración 42 - Remote Wiring de Windows 50](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152510)

[Ilustración 43 - Sketch de Arduino 50](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152511)

[Ilustración 44 - Código StandardFirmata 51](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152512)

[Ilustración 45 - Código ConfigurableFirmata 51](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152513)

[Ilustración 46 - Logo de Python 52](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152514)

[Ilustración 47 - Menú de Raspbian 52](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152515)

[Ilustración 48 - Raspbery Pi 2 y sus GPIOs 53](#_Toc502152516)

[Ilustración 49 - Raspberry Pi 3 59](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152517)

[Ilustración 50 - Arduino Mega 59](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152518)

[Ilustración 51 - Arduino Nano 59](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152519)

[Ilustración 52 - Motores CC 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152520)

[Ilustración 53 - Sensor de ultrasonido 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152521)

[Ilustración 54 - Portapilas 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152522)

[Ilustración 55 - Módulo Puente H 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152523)

[Ilustración 56 - Mini-protoboard 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152524)

[Ilustración 57 - Sensor de Temperatura 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152525)

[Ilustración 58 - MQ7 CO 60](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152526)

[Ilustración 59- GPS 61](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152527)

[Ilustración 60 - Cámara V2 61](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152528)

[Ilustración 61 - Panel Solar Power Bank 61](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152529)

[Ilustración 62 - Diseño estructura SketchUp 61](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152530)

[Ilustración 63 - Impresión 3D 61](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152531)

[Ilustración 64 - Nivel 2 descubierto 62](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152532)

[Ilustración 65 - RM Vista Lateral 62](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152533)

[Ilustración 66 - Esquema general del SAR 63](#_Toc502152534)

[Ilustración 67 - Logo PM2 63](file:///C:\Users\Agustin\Documents\Agustin\Universidad\Tesina\Proyecto%20de%20tesina\sar\Capitulos\Tesina%20Mansilla-Schlapp%20Correcion%20hasta%20Pag%2030.docx.docx#_Toc502152535)

[Ilustración 68 - Arquitectura lógica del SAR 64](#_Toc502152536)

[Ilustración 69 - Aplicación Web 65](#_Toc502152537)

# Capítulo 1 - Introducción

## **1.1 Objetivo general**

Se pretende desarrollar un prototipo de un Sistema Autónomo Robótico (SAR), gestionado por un software definido como agente inteligente (que responda al modelo basado en objetivos[[1]](#footnote-1))para la exploración y análisis del medio ambiente.

### 1.1.1 Objetivos específicos

* Desarrollar una aplicación web multiplataforma que mediante comunicación inalámbrica permita el control del Robot móvil.
* Ensamblar un robot móvil integrando las plataformas Arduino y Raspberry Pi con diversos módulos y software.
* Investigar protocolos existentes y evaluar la necesidad de diseño de protocolos de comunicación para el control y procesamiento de datos entre el microcontrolador y la aplicación.
* Ensamblar físicamente e integrar a nivel de software los distintos componentes (sensores y actuadores) al SAR.
* Extender la aplicación para interactuar con la información que brinda el SAR de los sensores.

### 1.1.2 Metodología

El SAR se creará mediante las plataformas Arduino y Raspberry Pi. El robot poseerá motores como actuadores para desplazarse sobre la superficie a explorar y diversos sensores que permitan tomar muestras del ambiente explorado. Todos estos componentes se ensamblarán sobre distintas piezas estructurales para conformar el robot móvil o RM.

El RM estará en un estado receptivo, donde se le otorga el control a una aplicación web, la cual contará con una interfaz de usuario que facilitará la comunicación con el SAR. La aplicación permitirá manipular el desplazamiento del RM sobre la superficie y obtener las muestras del ambiente según se soliciten, en otras palabras, la lectura de los sensores.

La comunicación entre el SAR y la aplicación se realizará por medio de señales inalámbricas de radiofrecuencia. Se mantendrá una arquitectura de diseño denominada cliente/servidor, donde el cliente es el dispositivo que ejecuta la aplicación y el servidor es el SAR.

## **1.2 Motivación**

Las nuevas tendencias de hardware comomicrocontroladores, Smartphones y nuevos dispositivos programables, requieren contar con un nuevo esquema de diseño donde se puedan integrar las distintas tecnologías relacionadas (robótica, redes, plataformas móviles, etc.) en un área de conocimiento específica, para lograr una integración de saberes y disminuir la curva de aprendizaje de personas que se introducen en estas temáticas.

Para esto se necesita incursionar en el investigación y desarrollo en los ámbitos de la computación, control, mecánica y electrónica. Los cuales dieron paso a la robótica como técnica que combina diversas disciplinas, logrando un alto impacto en la sociedad en diversos ámbitos.

En la actualidad es muy popular la utilización de teléfonos móviles inteligentes (*smartphones*). De estos dispositivos, un segmento mayoritario se basa en el sistema operativo Android, presentado por Google en el 2007.

Android está basado en Linux y utiliza Java como lenguaje de desarrollo de aplicaciones. Por otro lado, Arduino, introducido en el año 2005, es una plataforma de hardware libre para electrónica orientado a la computación física (Phisical Computing).

Arduino aprovecha ciertas características de C++ para permitir el desarrollo de pequeños programas o sketches con conocimientos básicos de programación y electrónica. Esta simplicidad, sumado al bajo coste de las placas ha otorgado a la plataforma una gran popularidad.

Tanto Java como C++ han sido lenguajes utilizados en las actividades de laboratorio de varias cátedras de la Licenciatura por lo cual consiste en una motivación para llevar a cabo esta tesina.

Los nuevos avances en interoperabilidad de las distintas plataformas de las áreas de robótica y programación tanto en hardware como software, brindan un excelente recurso en materia de educación de nivel medio y superior permitiendo agilidad en el desarrollo de proyectos educativos con escaso conocimiento en dichas áreas. Es por ello que se necesita un estándar o prototipo de dónde partir, que se encuentre testeado con una biblioteca de funciones inmersas en el mismo y una arquitectura moldeable a distintas temáticas. Este prototipo base es el denominado SAR que se quiere desarrollar. En síntesis, el objetivo del SAR es crear un instrumento didáctico para la comprensión e incentivación de los alumnos en las distintas áreas mencionadas (robótica e informática).

## **1.3 Desarrollos Propuestos**

* Diseño y desarrollo del software necesario para el funcionamiento del SAR.
* Ensamblado de un prototipo hardware basado en Arduino y Raspberry Pi, integrado por distintos módulos compatibles con dichas plataformas.
* Diseño y desarrollo de una aplicación web que permita controlar el RM cuya interfaz integre la visualización de valores recolectados por los sensores integrados al SAR y generación de estadísticas a partir de estos datos.
* Selección de un medio de comunicación inalámbrica (Radiofrecuencia) que permita la interrelación entre la aplicación móvil y el SAR.

## **1.4 Resultados Esperados**

Al finalizar la tesina esperamos haber construido el robot móvil a partir de la integración de las diversas plataformas previamente mencionadas, conformando el denominado SAR.

Se espera aportar conocimiento significativo para futuros proyectos que requieran la utilización de protocolos de comunicación inalámbricos entre aplicaciones móviles y microcontroladores.

Tanto el desarrollo del software como el hardware serán liberados para contribuir a un mejor proceso de enseñanza de la informática y robótica en principio en el nivel medio.

Un resultado esperable es que el SAR en su conjunto sea fácilmente extensible y por lo tanto se prevé que otros continúen la evolución del producto y sea utilizado como base para nuevos proyectos relacionados con la robótica y aplicaciones móviles.

Otro resultado esperado es que los anexos referentes a la utilización de módulos sean de utilidad para la enseñanza de electrónica en nivel medio.

# Capítulo 2 - La robótica

 En este capítulo se va a abordar el concepto de la robótica desde el punto de vista de su utilidad en áreas relacionadas con la informática, para el ámbito educativo. Se introducen diversas estructuras robóticas, como también distintas plataformas que facilitan la aplicación de esta ciencia, dando soporte didáctico, en la actualidad. Además, se distinguen los conceptos de microcontrolador y computadora de placa reducida, detallando ventajas, desventajas y formas de comunicación de cada uno de ellos. Finalmente, se define que es un sistema autónomo robótico (el cual, como se mencionó en el capítulo anterior, es el desarrollo propuesto por esta tesina) concluyendo con el impacto de la robótica en la educación.

## **2.1 ¿Qué es la robótica?**

A lo largo de la historia el ser humano ha sentido fascinación por las máquinas que puedan imitar las figuras y movimientos de seres animados. El poder desarrollar sistemas electromecánicos que simulen o realicen actividades típicas de seres vivos, ofrece la sensación de tener un propósito propio, lo cual fue un motivador para su estudio.

A este tipo de maquinaria se la denomina Robot. Según la RIA (Robotic Industries Association):

*“Un robot es un manipulador funcional reprogramable, capaz de mover material, piezas, herramientas o dispositivos especializados mediante movimientos variables programados, con el fin de realizar tareas diversas.”*

Una de las grandes diferencias entre los robots y el resto de las máquinas es la versatilidad que adquieren los mismos al poder variar su propósito modificando su programación. Todas las tareas que realizan los robots están basadas en la manipulación de su entorno.

Se le considera robótica a la ciencia y técnica encargada del diseño, construcción y aplicación de robots. Esta ciencia involucra diversas disciplinas tales como la mecatrónica, electrónica, mecánica, e informática, entre otras.

Actualmente la robótica ha ido evolucionando rápidamente, dando lugar a innovaciones tecnológicas destacadas para la historia de la humanidad, logrando un alto impacto socio-económico. Hoy en día, la robótica no es solo utilizada en los ámbitos industriales o militares, sino que podemos ver a robots en variadas áreas como por ejemplo en la medicina o en la educación.



Ilustración - Esquema básico de un robot

En la imagen (Ilustración 1 - Esquema básico de un robot) se puede apreciar el esquema básico del funcionamiento de un robot, detallando los componentes que pueden tener (Actuadores, sensores y un sistema de control).

La robótica está constituida por tres grandes temas como lo son; la *percepción*, la *planificación* y la *manipulación*. En conjunto permiten el desarrollo de robots con un gran índice de autonomía, logrando acciones básicas que realiza un ser humano al ejecutar ciertas tareas. Cuando una persona ha detectado una necesidad, los primeros pasos que realiza es estudiar su entorno con alguno de sus cinco sentidos (*percepción*); luego toma la decisión de realizar acciones con determinados movimientos (*planificación*) para que, finalmente, las ejecute de modo secuencial (*manipulación*).

Podemos identificar elementos y acciones relacionados con cada etapa de la secuencia antes descripta:

Percepción:

* Sensores
* Tratamiento de información
* Procesamiento de información

Planificación:

* Trayectorias
* Tareas
* Planificación de tareas
* Toma de decisiones

Manipulación:

* Mecánica
* Actuadores
* Sistema de control
* Sistema de programación

## **2.2 Estructura física de los robots**

La estructura es definida por el tipo de configuración general de las distintas piezas que conforman al Robot. Es difícil establecer una clasificación estricta de los mismos que resista un análisis riguroso. La subdivisión de los Robots, con base en su arquitectura, se podría hacer dentro de alguno de los siguientes grupos: poliarticulados, móviles, androides, zoomórficos e híbridos.

### 2.2.1 Poliarticulados

Se les denomina robots poliarticulados a aquellos que en su mayoría son sedentarios o de desplazamientos muy limitados y tanto su forma como configuración pudiera ser muy diversa. En este grupo entrarían aquellos robots estructurados para mover sus componentes terminales (Ej.: sus actuadores) en un espacio determinado de trabajo con una simetría específica. Ejemplos, podrían ser los robots industriales, cartesianos y/o manipuladores. En la ilustración anterior (Ilustración 2 - Ejemplo de robot poliarticulado) se muestra un brazo robótico como ejemplo de un robot poliarticulado.

Ilustración - Ejemplo de robot poliarticulado

### 2.2.2 Móviles

Estos robots se caracterizan, primordialmente, por su capacidad de desplazamiento. Su forma, por lo general, se basa en diseños típicos de vehículos como los automóviles. Su objetivo prioritario suele ser recorrer un determinado camino guiándose por la información de su entorno, obtenida a través de sus sensores. Pueden ser dotados de un cierto nivel de inteligencia (gracias a su programación) e incluso sortear obstáculos. En la imagen (Ilustración 3 - Ejemplo de robot móvil) se visualiza un robot móvil que cuenta con 4 ruedas y motores para su desplazamiento, y a su vez con un brazo manipulado por servo motores.

Ilustración - Ejemplo de robot móvil

### 2.2.3 Androides



Se les llama androide a los robots que intentan simular y/o reproducir la forma y comportamiento cinemático de seres vivos. Todavía no cuentan con alguna aplicación práctica específica, sino más que, para el estudio y la experimentación. La imagen (Ilustración 4 - Androide Asimo de Honda) muestra el androide ASIMO creado por la compañía japonesa Honda en el año 2000.

Ilustración - Androide Asimo de Honda

### 2.2.4 Zoomórficos

Los Robots zoomórficos, se caracterizan principalmente por sus sistemas de locomoción que tienen como objetivo imitar a los diversos seres vivos, como se puede apreciar en la imagen (Ilustración 5 - Robot Zoomórfico caminador) un robot con forma canina. A pesar de la disparidad morfológica de sus posibles sistemas de locomoción se suelen distinguir entre dos categorías principales: caminadores y no caminadores. El grupo de los no caminadores está muy poco evolucionado. Los Robots zoomórficos caminadores multípedos son muy numerosos y están siendo objeto de experimentos en diversos laboratorios con vistas al desarrollo posterior de verdaderos vehículos terrenales, pilotados o autónomos, capaces de evolucionar en superficies muy accidentadas. Las aplicaciones de estos Robots apuntan a su utilización en el campo de la exploración espacial y en el estudio de los volcanes.

Ilustración - Robot Zoomórfico caminador

### https://lh4.googleusercontent.com/Iop1qqdMsk7UnEMkQs6-v938nAD7qo8OVTlpS-kQ6kgmjNjhegpQ9YcBiHqOy3RBTBYb5whkIafhH6t6Bfsxk6ALuxxxNW5ErbhPGpIyAI2Y3ZQJCFjVwj3AkZABWm4fRvTY4zdO2.2.5 Híbridos

Los robots híbridos se les consideran a aquellos a los cuales es difícil clasificar dentro de las mencionadas anteriormente o bien es la combinación de algunas de ellos. En esta imagen (Ilustración 6 - Robot móvil-poliarticulado), se puede observar un robot móvil con variados actuadores para la manipulación de objetos y que además su forma es similar a la de un escorpión.

Ilustración - Robot móvil-poliarticulado

## **2.3 Distintas tecnologías para la robótica educativa**

Sin duda alguna, en los últimos años, las arquitecturas más destacadas para la enseñanza y desarrollo de robótica a nivel educativo han sido las plataformas **Arduino**[[2]](#footnote-2)y **Raspberry Pi[[3]](#footnote-3)**. Gracias a su costo accesible y disponibilidad de versiones, estas tecnologías son utilizadas en las diversas disciplinas relacionadas con la robótica educativa. En el caso de Arduino, presenta una notable ventaja dentro de este ámbito dado que la compañía que lo fábrica (del homónimo Arduino) libera su hardware y a su vez ofrece una amplia variedad de modelos para usos múltiples (se brindará más detalle sobre esta tecnología en el siguiente capítulo). Por otro lado, Raspberry Pi es un computador reducido creado con el objetivo de la enseñanza de la informática, cuenta con notables capacidades de procesamiento en relación a su bajo costo.

La gran ventaja de estas arquitecturas con respecto a las que se mencionan a continuación, es su gran soporte y compatibilidad, dada la amplia comunidad que las utiliza.

Existen otras tecnologías para el desarrollo de la robótica tales como; la plataforma **Intel Galileo**, similar a Raspberry Pi pero desarrollada por Intel, es también un computador reducido certificado por Arduino que integra la arquitectura Intel X86; **BeagleBone**, es una placa computadora de hardware libre diseñada como plataforma de evaluación y de prototipos para ingenieros profesionales; **Nanode**, es un placa de microcontrolador de código abierto, similar a Arduino, que cuenta con un módulo Wifi incorporado, su objetivo es el de la experimentación en Iot (Internet de las cosas).

## **2.4 Microcontroladores y computadora de placa reducida (SBC)**

Un **microcontrolador** es un circuito integrado programable, por lo general montado sobre una PCB (placa de circuito impreso), con la capacidad de ejecutar órdenes cargadas en su memoria. Su velocidad de procesamiento es limitada comparada con un CPU dado que su objetivo es el de funcionar como controlador. Son utilizados en periféricos informáticos, electrodomésticos, control de sistemas mecánicos, etc.

Puede ser muy común pensar que un microcontrolador es igual a un microprocesador, pero esto no es así, de hecho, difieren en muchos aspectos. La principal diferencia es su funcionalidad, dado que, para utilizar un microprocesador en alguna aplicación real, se debe conectar con diversos componentes tales como memorias o buses de transmisión de datos.

Aunque el microprocesador se considera una máquina de computación poderosa, no está preparado para la comunicación con los dispositivos periféricos que se le conectan. Para que el microprocesador se comunique con algún periférico, debe interactuar con un microcontrolador (cómo por ejemplo en el caso de un mouse, disco rígido o una cámara web). Por ende, se puede decir que, el CPU requiere del microcontrolador para la comunicación con el resto del hardware. Así era en el principio y esta práctica sigue vigente en la actualidad.

Por otro lado, al microcontrolador se lo diseña de tal manera que tenga todos los componentes integrados en el mismo chip, como se puede apreciar en la siguiente imagen (Ilustración 7 - Arquitectura de un microcontrolador). No necesita de otros componentes especializados para su operación, porque todos los circuitos necesarios, que de otra manera correspondan a los periféricos, ya se encuentran incorporados. De esta forma se ahorra tiempo y espacio al momento de su utilización.

Es por estas razones que han tenido grandes repercusiones para el desarrollo de la robótica.

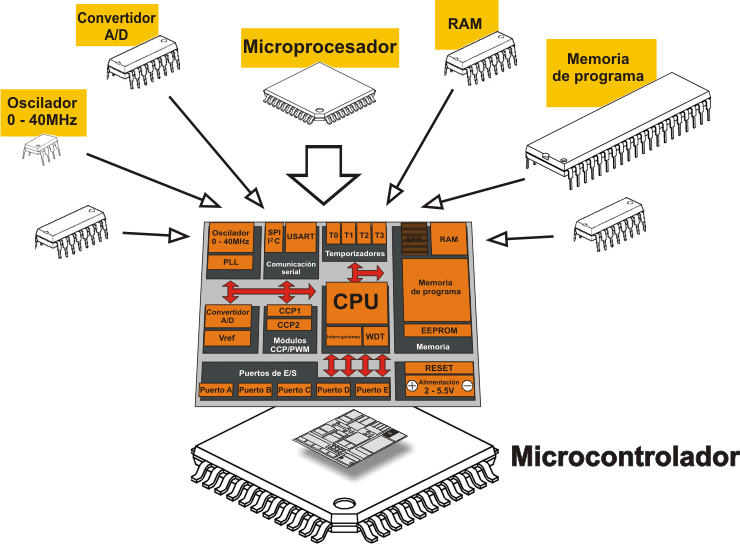


Ilustración - Arquitectura de un microcontrolador

Una **computadora de placa reducida** (SBC, *Single Board Computer*),en cambio, es una computadora completa que integra todos los componentes necesarios, que definen a la misma, en un solo circuito (la placa madre o *motherboard*) con la particularidad de que la misma es de un tamaño mucho más reducido que el de una computadora tradicional. Ejemplos típicos de este tipo de computadoras son las plataformas Arduino y Raspberry Pi.

En el caso de Arduino, dentro de su placa se integra un microcontrolador para el procesamiento de sus órdenes programadas, en cambio, Raspberry Pi integra un microprocesador con capacidades de ejecutar un sistema operativo con interfaz gráfica.

## **2.5. Comunicación entre distintas plataformas de cómputo**

Existen diversos medios de comunicación entre las PCs y las SBCs o de dispositivos de cómputo entre sí, a continuación, se listan algunos de ellos:

* Serial: La comunicación serie o serial es una interfaz de comunicación de datos digitales que nos permite establecer transferencia de información entre varios dispositivos. Un puerto es el nombre genérico con que denominamos a las interfaces, físicas o virtuales, que permite esta comunicación entre dispositivos. Un puerto serie envía la información mediante una secuencia de bits. Para ello se necesitan al menos dos conectores para realizar la comunicación de datos, RX (recepción) y TX (transmisión). Las placas Arduino actuales cuenta con un puerto USB para realizar este tipo de comunicación y es su principal interfaz para conectarlos a una PC donde cargar la secuencia de órdenes que luego ejecutará.
* Inalámbricas: Los medios de comunicación inalámbricos, para computadoras, han evolucionado de forma exponencial desde su aparición. Su gran ventaja, como su nombre lo dice, es que no necesitan de un medio de propagación físico (como los cables) para la transmisión de los datos, sino que, para el envío de los mismo se utiliza la modulación de ondas electromagnéticas a través del espacio. Existen diversos tipos, con grandes diferencias en cuanto a velocidades y rangos de alcance. En cuanto para la robótica podemos encontrar dispositivos que nos permitan conectar computadoras de placas reducidas con diversos computadores por medio de:
  + *Radiofrecuencia*: Existen módulos compatibles con Arduino, como el módulo de radiofrecuencia RF 433Mhz, que nos permiten conectar dos dispositivos de este tipo entre sí de forma inalámbrica a través de radiofrecuencia.
  + *Infrarrojo*: Las redes de luz infrarroja están limitadas por el espacio, se utilizan por lo general en dispositivos que se encuentran en un mismo espacio físico como un cuarto o un piso. Utilizan luz infrarroja tanto como para la transmisión como para la recepción de datos.
  + *Bluetooth*: Es una especificación industrial que permite crear redes inalámbricas de área personal (WPAN), mediante un enlace de radiofrecuencia que trabaja en la banda ISM (Industrial Scientific and Medical) de 2.4 GHz posibilitando la transmisión de voz y datos.
  + *Wifi*: Este mecanismo de comunicación inalámbrica es el más popular entre computadoras de hoy en día. A su vez, es una marca de la Alianza Wi-fi la cual certifica que los dispositivos cumplan con los estándares IEEE 802.11 vigentes relacionados a redes inalámbricas de área local.

## **2.6 ¿Qué es un SAR (Sistema Autónomo Robótico)?**

Se le considera SAR o sistema autónomo robótico a aquellos robots que presentan cierto grado de autonomía (Inteligencia artificial[[4]](#footnote-4)) y que, cuentan con la capacidad de testear su entorno (por medio de sensores) para decidir qué acciones realizar (por medio de actuadores). Por ende, se puede decir que, son sistemas dinámicos que consisten en un controlador electrónico acoplado a un cuerpo mecánico.

En el desarrollo propuesto por esta tesina, se diseñó y armó un sistema autónomo robótico móvil que posee un cierto grado de inteligencia, pero a su vez, permite ser manipulado desde una aplicación web.

## **2.7 La robótica en la educación**

En educación pueden diferenciarse dos tipos de uso de la programación y la robótica como apoyo en la clase: por un lado, la robótica y la programación como elemento educacional, y por otro, como elemento social.

Como elemento educacional, consiste en un conjunto de elementos físicos o de programación que motivan a los estudiantes a construir, programar, razonar de manera lógica y crear nuevas interfaces o dispositivos.

Mientras que, por otro lado, la programación y la robótica también es utilizada como elemento social, por ejemplo, a modo de juego o gamificación, de forma que sistemas autónomos o semiautónomos interactúan con humanos u otros agentes físicos o software en roles como entrenador, compañero, dispositivo tangible o registro de información.

El desarrollo de actividades educacionales basadas en robots o en programación pueden incrementar el compromiso por el aprendizaje en otras áreas como literatura o historia a través del juego y la motivación. Aún más, su uso puede mejorar el desarrollo ético, emocional y social en base al impacto que, por ejemplo, un robot con atribuciones sociales puede causar en los niños.

Otro beneficio, es su potencial educativo para niños con necesidades especiales tanto en las áreas cognitivas como psicosociales. La escalabilidad de las propuestas educativas basadas en robots, y su enorme potencial motivador, lo hacen especialmente útil en programas de refuerzo y de educación especial.

Una de las grandes controversias en estas áreas, es sobre los materiales que deben utilizarse en el aula. Algunos investigadores, como Cecilio Angulo (Profesor de la Universitat Politècnica de Catalunya y director del Grupo de Investigación en Ingeniería del Conocimiento), afirman que los dispositivos tangibles aumentan el nivel de inmersión porque los estudiantes están manipulando las cosas en un mundo real. Sin embargo, podemos encontrar otros estudios que entienden que los dispositivos no tangibles, como los elementos de programación, atraen más y evitan limitaciones a causa de la necesidad de un cuerpo físico en el espacio real. Por tanto, lo que parece lógico es un enfoque híbrido entre robótica y programación, donde una fusión entre lo físico y lo virtual proporciona más flexibilidad a los docentes y a los estudiantes.

La robótica y la programación en conjunto brindan una experiencia de aprendizaje particular respecto a otras áreas, porque la posibilidades ofrecidas por la utilización de computadoras se localiza no solo en una pantalla, sino también, en objetos tangibles, que comparten con los interesados en un espacio físico con la posibilidad de afectar su entorno. Aprender a través de la robótica aumenta el compromiso de los alumnos en actividades basadas en la manipulación, el desarrollo de habilidades motoras, la coordinación ojo-mano y una forma de entender las ideas abstractas. Además, las actividades basadas en robots proporcionan un contexto apropiado para el comportamiento cooperativo y el trabajo en equipo.[[5]](#endnote-1)

En Argentina, existen distintos centros de estudios relacionados con la robótica educativa, uno de los más renombrados es RoboGroup. Esta es una empresa nacional dedicada al diseño, fabricación y capacitación en robótica, que, según la misma, su objetivo es insertar la robótica como sistema interdisciplinario de aprendizaje en las entidades educativas de todos los niveles de nuestro país. Anualmente organiza campeonatos de robots para alumnos de colegios primarios y secundarios llamados Roboliga.

# Capítulo 3 – Arduino

## **3.1 ¿Qué es Arduino?**

Arduino es una plataforma y compañía, del mismo nombre, de electrónica "open-source" o de código abierto cuyos principios son contar con software y hardware fáciles de usar y que cualquiera pueda fabricar y mejorrar. Es decir, se propone como una plataforma sencilla de aprender para realizar proyectos interactivos para público no necesariamente con conocmientos técnicos.

Ilustración - Logo Arduino

Arduino se trata de una SBC con entradas y salidas, analógicas y digitales, la cual es programda bajo un entorno de desarrollo que inspirado en el entorno de programación **Processing**. En la imagen (Ilustración 8 - Logo Arduino) se puede ver el logo oficial de la compañía.

## **3.2 ¿Qué es Processing?**

**Processing**es un lenguaje de programación y entorno de desarrollo integrado de código abierto basado en Java, de fácil utilización, y que sirve como medio para la enseñanza y producción de proyectos multimedia e interactivos de diseño digital. En la imagen (Ilustración 9 - Logo de Processing) se puede apreciar su logo.

Uno de los objetivos expresos de Processing es el de actuar como herramienta para que artistas, diseñadores visuales y miembros de otras comunidades ajenos a la programación, aprendan las bases de la misma a través de una realimentación gráfica inmediata y visual de los resultados obtenidos de su experiencia de programación.

Ilustración - Logo de Processing

El lenguaje de Processing se basa en Java, aunque hace uso de una sintaxis simplificada y de un una biblioteca sencilla para generación de gráficos.

Por otro lado, Wiring (Basado en Processing) viene con una librería de C/C++ la cual hace operaciones comunes de input/output mucho más fácil. Los programas de Wiring están escritos en C/C++, pese a que sus usuarios sólo necesiten definir dos funciones para hacer un programa ejecutable:

setup() – una función ejecutada sólo una vez en el inicio de un programa la cual puede ser usada para definir los ajustes iniciales de un entorno.

loop() – una función llamada repetidamente hasta que la placa es apagada.

Como podemos apreciar en la siguiente ilustración (Ilustración 10 - Wiring IDE) hacer un blink a un led es muy sencillo dado la abstracción que nos otorga la librería. Un blink es un parpadeo de un led conectado a la placa. Se lo considera el “hola mundo” de Arduino.

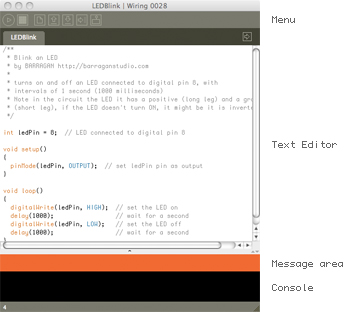


Ilustración 10 - Wiring IDE

Por otro lado, trabajando con C++ sin librerías quedaría como en la siguiente figura (Ilustración 11 - C++ Blink ejemplo) haciendo el mismo blink, programando directamente con la biblioteca AVR (control de los puertos de entrada y salida) y en C.[[6]](#endnote-2)



Ilustración 11 - C++ Blink ejemplo

## **3.3** **Fritzing**

El entorno Fritzing ayuda a los diseñadores y artistas a documentar sus prototipos interactivos y dar paso en la creación de prototipos físicos al producto real. Como podemos apreciar en la siguiente ilustración (Ilustración 12 - Entorno Fritzing), permite arrastrar componentes y generar un sketch.

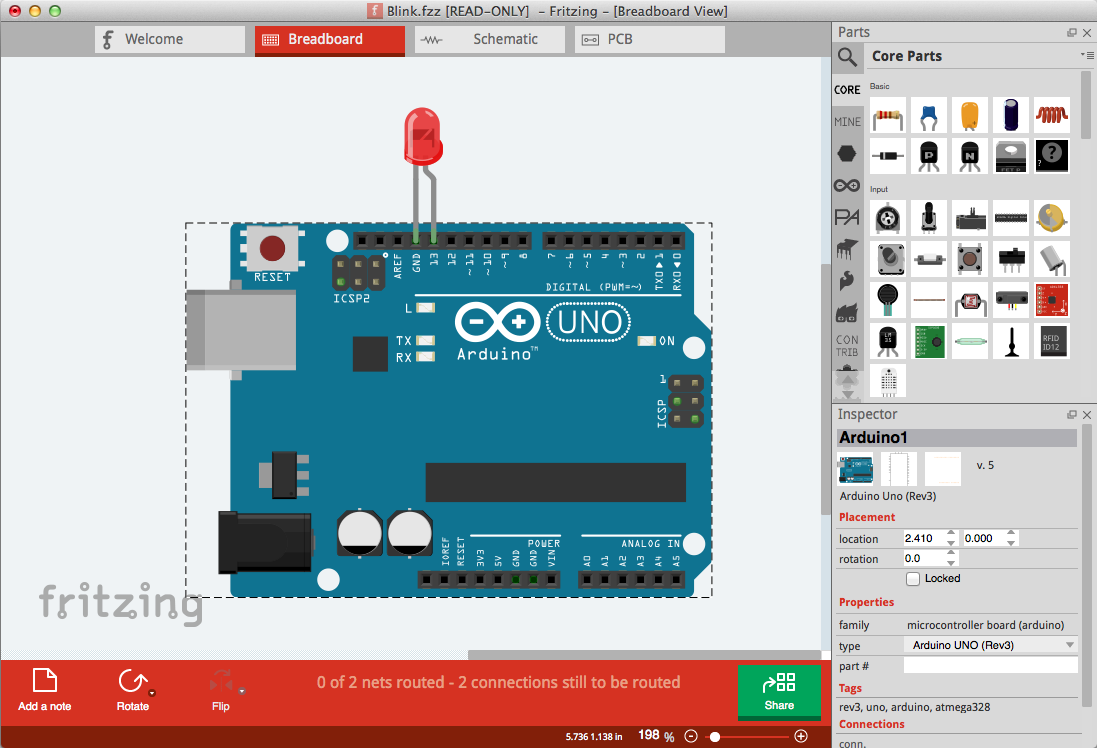


Ilustración 12 - Entorno Fritzing

## **3.4 ¿Qué es Wiring?**

Wiring es una plataforma de prototipado electrónico de fuente abierta compuesta de un lenguaje de programación, un entorno de desarrollo integrado (IDE), y un microcontrolador.

Esta plataforma permite escribir software para controlar dispositivos conectados a la tarjeta electrónica para crear toda clase de objetos interactivos, espacios o experiencias físicas que sienten y responden al mundo físico.

Este proceso se llama *sketching* con hardware; se explora una gran cantidad de ideas de forma muy rápida, se seleccionan las más interesantes, se afinan y producen prototipos en un proceso iterativo.

## **3.5 Entonces Arduino es…**

* Una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.
* Una plataforma de hardware abierto que facilita la programación de un microcontrolador. Los microcontroladores nos rodean en nuestra vida diaria, usan los sensores para escuchar el mundo físico y los actuadores para interactuar con el mismo. Los microcontroladores leen sobre los sensores y escriben sobre los actuadores.

## **3.6 Plataforma Arduino**

La plataforma consiste en una placa de circuito impreso con un microcontrolador, usualmente Atmel AVR, puertos digitales y analógicos de entrada/salida los cuales pueden conectarse a placas de expansión (*shields*), que amplían las características de funcionamiento de la placa Arduino. Asimismo, posee un puerto de conexión USB desde donde se puede alimentar la placa y establecer comunicación con el computador.

Los puertos serie que cuentan las distintas plataformas Arduino presentan puertos de serie como UART. La UART (universally asynchronous receiver/transmitter) es una unidad que incorporan ciertos procesadores, encargada de realizar la conversión de los datos a una secuencia de bits y transmitirlos o recibirlos a una velocidad determinada.

Por otro lado, también posee un nivel TTL (transistor-transistor logic). Esto significa que la comunicación se realiza mediante variaciones en la señal entre 0V y Vcc (donde Vcc suele ser 3.3V o 5V). Por el contrario, otros sistemas de transmisión emplean variaciones de voltaje de -Vcc a +Vcc (por ejemplo, los puertos RS-232 típicamente varían entre -13V a 13V).

Como podemos observar en la siguiente ilustración (Ilustración 13 - Ejemplo serie), se realiza una comunicación serie a (9600 bps) imprimiendo un contador. La zona marcada con rojo, es un botón que al presionarlo nos permite acceder a la terminal y ver el flujo serie seteando el clock correspondiente.

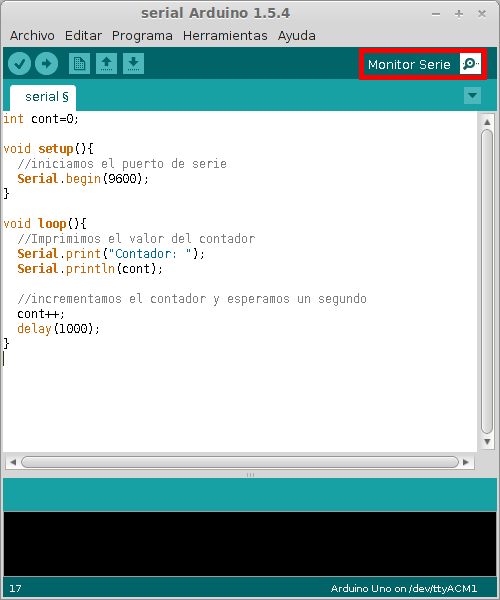


Ilustración 13 - Ejemplo serie

## **3.7 Distintas plataformas para Arduino**

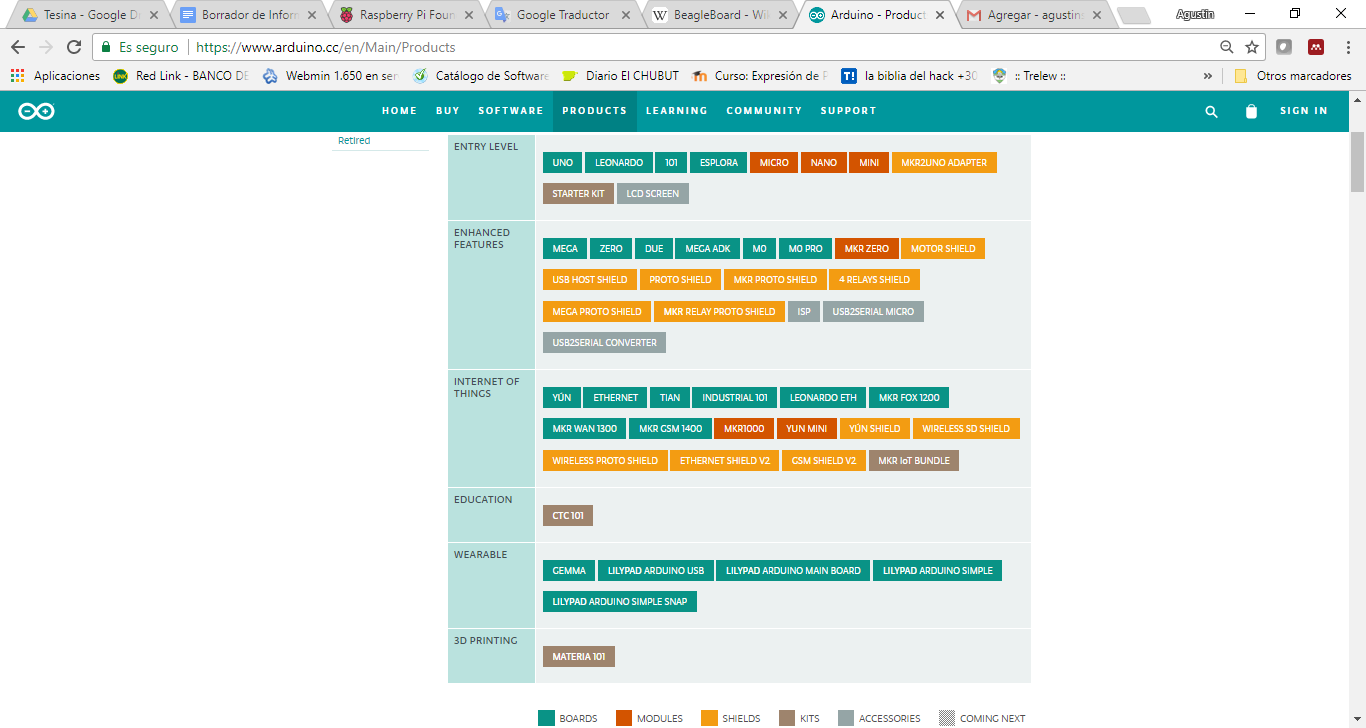


Ilustración 14 - Niveles de entrada a la plataforma Arduino

Existe una gran variedad de productos Arduino, la compañía los cataloga, como se puede aprecias en la imagen anterior (Ilustración 14 - Niveles de entrada a la plataforma Arduino), en distintos niveles según su utilidad[[7]](#endnote-3):

* Nivel de entrada: Son los más sencillos de utilizar, ideales para comenzar con la plataforma Arduino y realizar proyectos sencillos.
* Características mejoradas: Estas plataformas poseen características superiores, con respecto a las del nivel de entrada, están pensadas para proyectos más avanzados o de respuesta más rápida.
* Internet de las cosas: Estas placas vienen incorporadas con componentes que permitan realizar trabajos relacionados con la IoT (Internet de las cosas)[[8]](#footnote-5).
* Educación: En este caso, Arduino, ofrece un kit con herramientas y más de 25 proyectos, orientados a la educación, para realizar con sus plataformas.
* Usables: Estas plataformas están pensadas para “agregarle algo de electrónica” a prendas de vestir.
* Impresión 3D: Arduino ofrece una impresora 3D nombrada como Materia 101.

El hardware Arduino más sencillo consiste en una placa con un microcontrolador y una serie de puertos de entrada y salida. Los microcontroladores de 8 bits de AVR más utilizados en estas placas son el Atmega168, Atmega328, Atmega1280, y Atmega8 por su sencillez y bajo coste, aunque también se dispone de microcontroladores ARM, cómo el caso del CortexM3 de 32 bits. A pesar de que ARM y AVR son plataformas diferentes, al utilizar la IDE de Arduino, los programas se compilan y luego se ejecutan sin cambios en cualquiera de las plataformas. En la imagen (Ilustración 15 - Arduino Uno) se visualiza la distribución física de puertos y componentes de la versión Arduino Uno R3.

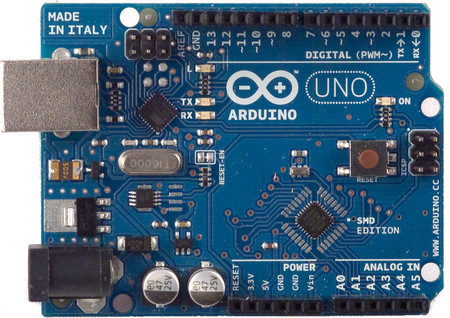


Ilustración - Arduino Uno

## **3.8 Diferencias entre distintas placas de la familia Arduino**

Una primera diferenciación entre los distintos modelos de Arduino la encontraremos en el voltaje o tensión de alimentación de las placas. Las basadas en CortexM3 operan con un voltaje de 3,3 voltios, mientras que la mayor parte de las placas basadas en AVR utilizan una tensión de 5 voltios. Esto de todas formas no es un factor decisivo en la elección de una placa, dado que existen conmutadores de tensión en muchos actuadores y sensores compatibles.

Los usos posibles que se le pueden dar a un Arduino, en forma general son:

* Aquellos en los que el Arduino es utilizado como microcontrolador, tiene un programa descargado desde un ordenador y funciona de forma independiente de éste, y controla y alimenta determinados dispositivos y toma decisiones de acuerdo al programa descargado e interactúa con el mundo físico gracias a sensores y actuadores.
* La placa Arduino hace de interfaz entre un ordenador (como podría ser una Raspberry Pi) u otro dispositivo, que ejecuta una determinada tarea, para traducir dicha tarea en el mundo físico a una acción (actuadores).

## **3.9 ¿Por qué usar Arduino?**

### 3.9.1 La comunidad

La comunidad Arduino se desarrolla y enriquece a partir del trabajo con la placa, de la experimentación, de la producción de conocimiento en torno a ella, y estas habilidades se comparten dentro de la comunidad, pudiendo cualquier persona tener acceso a ellas.

La página principal de Arduino da soporte por medio de secciones y dedica un blog histórico con novedades y proyectos que se encuentran en desarrollo.

Además, se crean sitios como Arduino playground (<http://playground.arduino.cc/>) que es una wiki donde todos los usuarios de Arduino pueden contribuir. Es el lugar donde publicar y compartir código, diagramas de circuitos, guías, manuales, cursos. Es la base de datos de conocimiento de la comunidad de Arduino. Este sitio a su vez tiene soporte a distintos lenguajes como el español (<https://playground.arduino.cc/Es/Es>).

Por otro lado, existe Arduino Hub, un lugar donde se comparten los proyectos, dando los distintos pasos para reproducirlo.

El manifiesto de la comunidad Arduino dice (traducción al español):

“Apoyar al ecosistema de hardware y software de open source Arduino, haciendo que los productos electrónicos sean abiertos y participativos.

Servir como un evangelista para Arduino, expandir el ecosistema de código abierto a estudiantes, fabricantes, desarrolladores, diseñadores, ingenieros y empresas dentro de sus comunidades locales.

Construir una red global de comunidades que diseñen y codifiquen proyectos, intercambien ideas, organicen actividades de colaboración y dicten cursos oficiales de Arduino, independientemente de su edad, sexo, idioma y capacidad técnica”[[9]](#footnote-6)

La siguiente imagen (Ilustración 16 - Logotipo comunidad open-source de Arduino) muestra el logotipo oficial de la comunidad open-source de Arduino.



Ilustración 16 - Logotipo comunidad open-source de Arduino

### 3.9.2 La sencillez del lenguaje de programación

Programar la placa es muy sencillo y accesible, y la ayuda por parte de la comunidad lo hace aún más fácil. Como se mostró en las ilustraciones (Ilustraciones 10 y 11), podemos apreciar librerías con un alto nivel de abstracción en cuanto al acceso a bajo nivel.

La definición de puertos con la sentencia a alto nivel, seteando los puertos del 1 al 7 en un Arduino UNO como salida sería:

PinMode(1,OUTPUT) ;

PinMode(2,OUTPUT) ;

…;

PinMode(7,OUTPUT) ;

en contraposición con:

DDRD = B11111110; [[10]](#footnote-7) // sets Arduino pins 1 to 7 as outputs, pin 0 as input

### 3.9.3 Es hardware de bajo costo

Lo único que “vale” en la placa son sus componentes, ya que no debemos pagar el costo de la licencia de su creador, por el hecho de ser hardware libre.

## **3.10 Incorporación de Arduino en las escuelas**

Gracias a la plataforma Arduino y su comunidad, se permite una vinculación con la dinámica de las escuelas a través de la creación de una red de trabajo colaborativo. Esto conlleva a:

* La utilización de medios multimediales, para conformar cursos y capacitaciones.
* Creación de proyectos articulares entre distintos espacios curriculares
* Uso de diversas herramientas que son Arduino-compatibles y gratuitas.
* Costos relativamente bajos en la adquisición de las distintas plataformas.

Todos estos puntos anteriores favorecen la introducción en el aula y clubes.

Por otro parte, la sencillez del lenguaje de programación de la placa, permite la rápida utilización por alumnos y docentes, no necesariamente del ámbito de la informática y la electrónica. Esto contribuye a la construcción colectiva del conocimiento, promoviendo la interdisciplinariedad escolar, donde docentes de distintas áreas articulan para crear proyectos.

Utilizando clubes de ciencia o proyectos específicos permite que el trabajo se apoye sobre un modelo pedagógico de aprendizaje en proceso, donde el sujeto que aprende es participante activo de ese proceso, desde la concepción de la idea hasta el producto final, incorporando conocimientos técnicos específicos.

Este tipo de actividades educativas hacen que la tecnología y su uso se pongan al servicio de la creatividad, el juego, la experimentación y la invención, con la posibilidad de ser adaptado al contexto en el que se inserta. Además, proporcionar la recuperación de la tecnología obsoleta existente en ellas **(3r: reducir, reciclar, reutilizar).**

### 3.10.1 Las tres erres

Las tres erres (3R) es una regla para cuidar el medio ambiente, específicamente para reducir el volumen de residuos o basura generada.

Cuando hablamos de reducir lo que estamos diciendo es que se debe tratar de simplificar el consumo de los productos directos.

Al decir **reutilizar**, nos estamos refiriendo a poder volver a utilizar los objetos y darles la mayor utilidad posible antes de que llegue el momento de desecharlos.

Por otro lado, **reciclar** consiste en el proceso de someter los materiales a una transformación en el cual se puedan volver a utilizar.

Esta definición se pretende aplicar en las escuelas haciendo un proceso de clasificación, selección y desoldando componentes electrónicos de placas en desuso y materiales que se han desechado en las instituciones o en hogares de los alumnos.

## **3.11 Actuadores y sensores**

Un **actuador** es un dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de una acción con la finalidad de generar un efecto sobre un proceso automatizado. Este recibe la orden de un regulador o controlador y en función a ella genera la orden para activar un elemento final de control, como por ejemplo un LED.

Un **sensor** es un objeto capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Las variables de instrumentación pueden ser, por ejemplo: intensidad lumínica, temperatura, distancia, aceleración, inclinación, presión, desplazamiento, fuerza, torsión, humedad, movimiento, pH, etc.

En conjunto, los sensores y actuadores, permiten la creación de distintos tipos de artefactos, que posibilitan comunicarse con el ambiente que los rodea, modificándolo (actuadores) o recibir estímulos (sensores).

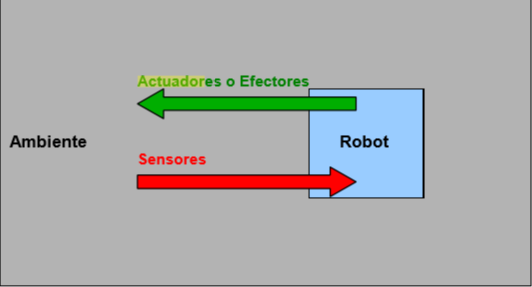


Ilustración - Representación actuadores y sensores

En esta imagen (Ilustración 17- Representación actuadores y sensores) se representa la toma de datos un robot puede capturar del ambiente que lo rodea mediante sus diversos sensores, y a su vez como podría interactuar con el mismo mediante actuadores.

## **3.12 Actuadores en el SAR**

Una de las ventajas que genero motivación para que el sector industrial comience a utilizar tecnologías como Arduino, fue la creación de actuadores compatibles con estas plataformas. Precisamente en el SAR se utilizarán:

* Motores de corriente continua
  + Para el desplazamiento del robot móvil
* LED
  + Para indicar estados del RM

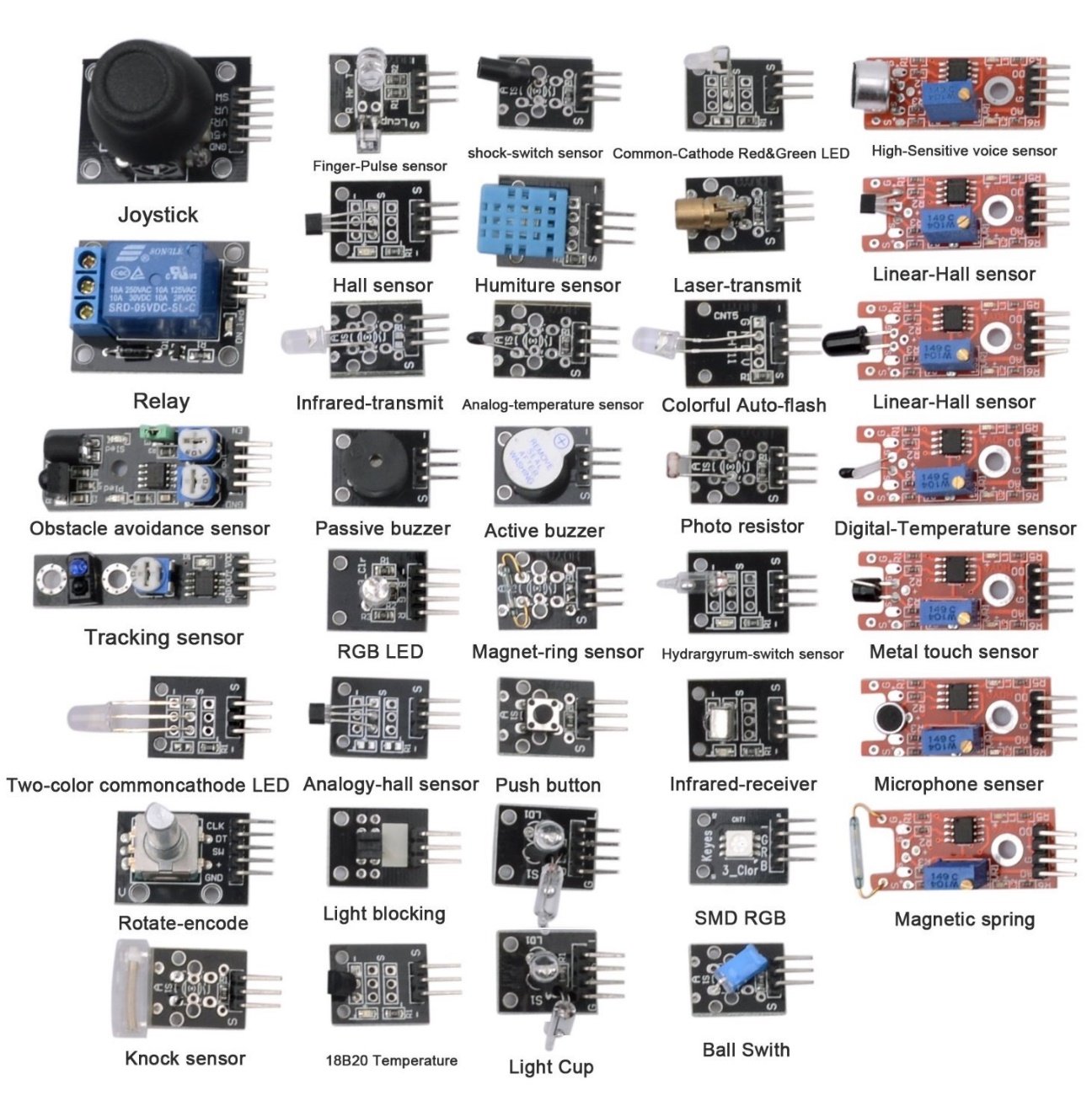


Ilustración - Actuadores y sensores compatibles con Arduino

## **3.13 Sensores en el SAR**

El SAR utiliza los siguientes sensores:

* Sensor ultrasónico HC-SR04
  + Para detectar objetos, y distancia entre el RM y elementos del ambiente
* Sensor de Temperatura KY-001
  + Incorporado para analizar la temperatura del ambiente
* Sensor de presencia de gases MQ-7
  + Detección de monóxido de carbono

Algunos de los sensores y actuadores se pueden apreciar en la ilustración anterior (Ilustración 18 - Actuadores y sensores compatibles con Arduino).

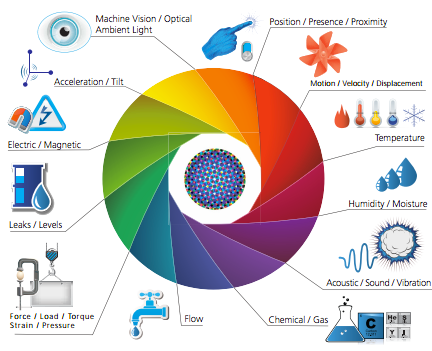


Ilustración - Representación de sensores

En esta imagen (Ilustración 19- Representación de sensores) se pueden apreciar los distintos factores de un entorno que pueden ser evaluadores con sensores mencionados anteriormente.

## **3.14 Módulos o *shields* en el SAR**

El SAR utiliza[[11]](#footnote-8):

* MotorShield L298
  + Para administración del puente H y gestión de los motores de CC
* Módulo bluetooth HC-05
  + Para la comunicación con dispositivos compatibles (móviles y/o computadoras)
  + Envío de órdenes
* Módulo GPS NEO-6
  + Para la geolocalización del RM
* Módulo ESP8266
  + Conectividad y transferencia de datos vía WIFI
  + Activación del Ad-hoc

A lo largo del desarrollo de la tesina se fueron implementando diversos casos de pruebas sobre los sensores, actuadores y módulos especificados en esta sección. Las pruebas se encuentran anexas en este documento.

# Capítulo 4 – Raspberry Pi

## **4.1 ¿Qué es Raspberry Pi?**

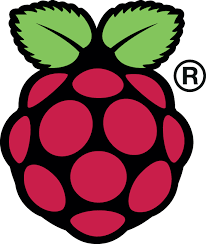
**Raspberry Pi** es un computador de placa reducida (SBC) desarrollado en Reino Unido por la Fundación Raspberry Pi. Su lanzamiento fue el 29 de febrero del 2012 con el *Raspberry Pi 1 Modelo A*. Su costo es relativamente bajo en relación a sus especificaciones técnicas (alrededor de U$D 25), dado que su objetivo primordial es el de estimular la enseñanza de la informática en las escuelas. Su logo oficial, como se muestra en la imagen (Ilustración 20 - Logo oficial de Raspberry Pi), no es más que una frambuesa.

Ilustración - Logo oficial de Raspberry Pi

## **4.2 Especificaciones técnicas de las distintas versiones**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Raspberry Pi 1 Modelo A** | | **Raspberry Pi 1 Modelo B** | | **Raspberry Pi 1 Modelo B+** | **Raspberry Pi 2 Modelo B** | **Raspberry Pi 3 Modelo B** |
| **SoC** | Broadcom BCM2835 ([CPU](https://es.wikipedia.org/wiki/CPU) + [GPU](https://es.wikipedia.org/wiki/GPU) + [DSP](https://es.wikipedia.org/wiki/Procesamiento_digital_de_se%C3%B1ales) + [SDRAM](https://es.wikipedia.org/wiki/SDRAM) + puerto USB) | | | | | Broadcom BCM2836 (CPU + GPU + DSP + SDRAM + Puerto USB) | Broadcom BCM2837 (CPU + GPU + DSP + SDRAM + Puerto USB |
| **CPU** | ARM 1176JZF-S a 700 MHz (familia ARM11) | | | | | 900 MHz quad-core ARM Cortex A7 | 1.2GHz 64-bit quad-core ARMv8 |
| **Juego de instrucciones** | RISC de 32 bits | | | | | | |
| **GPU** | Broadcom [VideoCore](https://es.wikipedia.org/w/index.php?title=VideoCore&action=edit&redlink=1) IV, OpenGL ES 2.0, MPEG-2 y VC-1 (con licencia), 1080p30 H.264/MPEG-4 AVC | | | | | | |
| **Memoria SDRAM** | 256 MiB compartidos con la GPU | 512 MiB compartidos con la GPU, desde el 15 de octubre del 2012 | | | | 1 GB compartidos con la GPU | |
| **Puertos USB 2.0** | 1 | 2 | | 4 | | | |
| **Entradas de vídeo** | Conector MIPI CSI que permite instalar un módulo de cámara desarrollado por la Fundación Raspberry Pi | | | | | | |
| **Salidas de vídeo** | Conector RCA (PAL y NTSC), HDMI (rev 1.3 y 1.4), interfaz DSI para panel LCD | | | | | | |
| **Salidas de audio** | Conector de 3.5 mm, HDMI | | | | | | |
| **Almacenamiento integrado** | SD, MMC, ranura para SDIO | | | MicroSD | | | |
| **Conectividad de red** | Ninguna | 10/100 Ethernet (RJ45) via hub USB | | | | | 10/100 Ethernet (RJ45) vía hub USB, Wifi 802.11n, Bluetooth 4.1 |
| **Periféricos de bajo nivel** | 8 x GPIO, SPI, I2C, UART | | | | | 17 x GPIO y un bus HAT ID | |
| **Consumo energético** | 500 mA (2.5 W) | 700 mA (3.5 W) | | 600 mA (3.0 W) | | 800 mA (4.0 W) | |
| **Fuente de alimentación** | 5 V vía Micro USB o GPIO header | | | | | | |
| **Dimensiones** | 85.60mm × 53.98mm | | | | | | |
| **SO soportados** | GNU/Linux: Debian (Raspbian), Fedora (Pidora), Arch Linux (Arch Linux ARM), Slackware Linux, SUSE Linux Enterprise Server for ARM.  RISC OS | | | | | | |

## **4.3 Entrada/Salida de propósito general (GPIO)**

Se le llama GPIO (En inglés, *General Purpose Input/Output*) a un pin genérico integrado a una placa o chip electrónico sin un fin específico, sino que, su “comportamiento” queda sujeto al usuario de dicha placa según algún tipo de lógica previamente cargada.

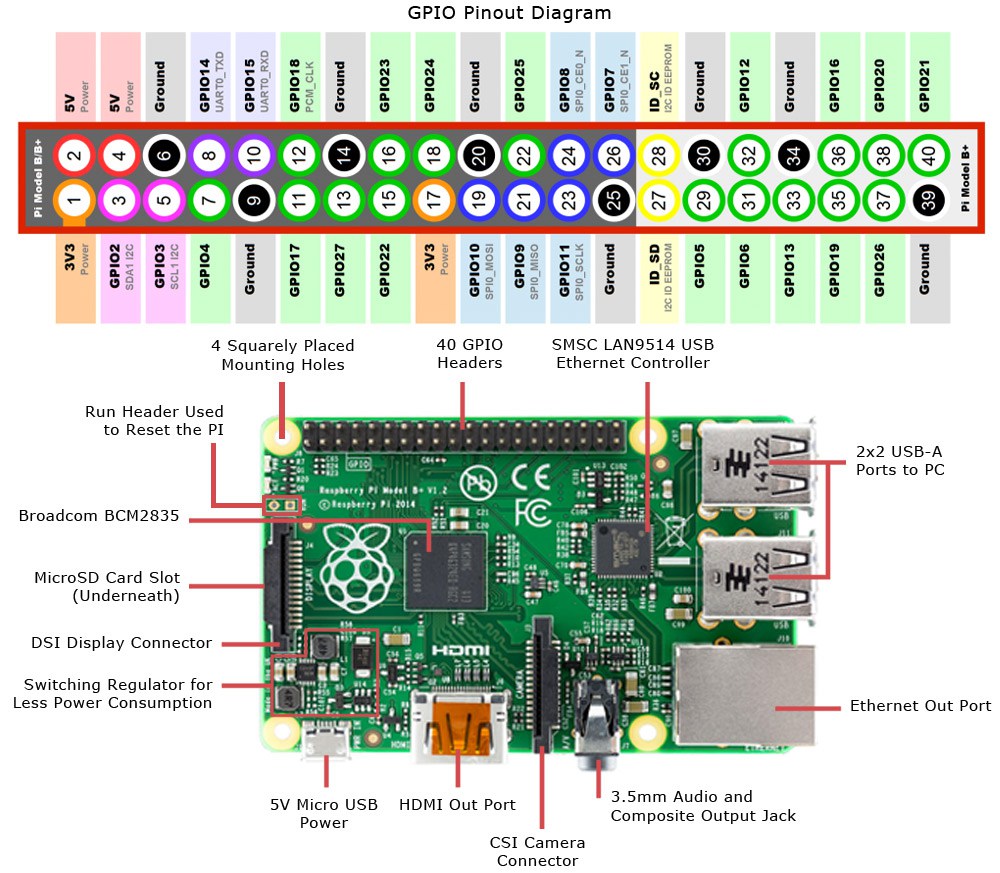
En la imagen (Ilustración 21 - Raspberry Pi 2 y sus GPIOs) se puede ver la Raspberry Pi 2 Modelo B de características bastante similares, en general, a la versión 3 de esta plataforma (utilizada en el desarrollo de esta tesina) y en detalle sus diversas interfaces. Un poco más arriba se pueden apreciar los distintos pines del tipo GPIO con los que cuenta esta plataforma (40 pines en total tanto la versión 2 como la 3)[[12]](#endnote-4).

Ilustración - Raspberry Pi 2 y sus GPIOs

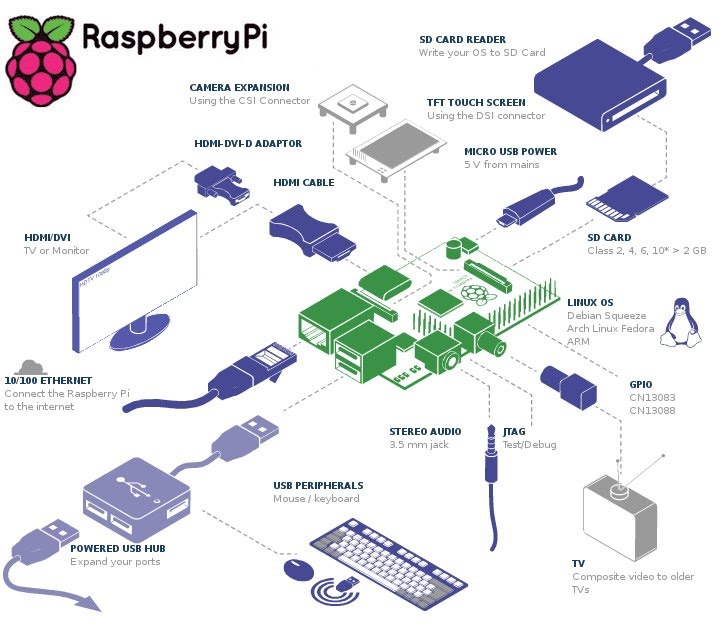
La siguiente imagen (Ilustración 22 - Interfaces de Raspberry Pi) ilustra los distintos periféricos que se pueden conectar a este computador.

Ilustración - Interfaces de Raspberry Pi

## **4.4 Sistemas Operativos compatibles**

Los computadores Raspberry Pi utilizan en su mayoría sistemas operativos basados en GNU/Linux compatibles con el mismo, alguno de ellos son los siguientes:

* Arch Linux
* Android
* Debian Whezzy
* Ubuntu Mate
* Google Chromium OS
* Raspbian

Este último (Raspbian), es una distribución derivada del sistema operativo Debian, la cual fue modificada y optimizada para el hardware de Raspberry Pi. Es la distribución por defecto recomendada por la Fundación Raspberry Pi para utilizarse en dicho computador.

Por otro lado, también existe una versión de Windows 10 desarrollada específicamente para sistemas embebidos, denominada Windows CE, compatible con esta plataforma (en particular con las Raspberrys Pi 2 y 3).

Los sistemas RISC OS también son compatibles con Raspberry Pi[[13]](#footnote-9).

## **4.5 Accesorios para Raspberry Pi**

Para poder operar la placa **Raspberry Pi**, es necesario contar con ciertos accesorios, como una fuente de alimentación de al menos 1A, un cable HDMI, una tarjeta de memoria microSD con el Sistema Operativo y un adaptador WIFI o un cable RJ45 para poder conectarla en red. Además, ya sea por estética o por protección existen variados gabinetes o carcasas para su resguardo.

Algunos de los accesorios más comunes compatibles para esta plataforma son los siguientes:

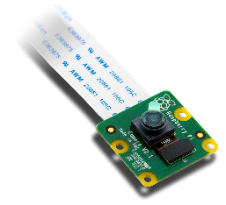
* *Cámara para Raspberry Pi V2*: Es una cámara de alta definición (HD) que se puede conectar a cualquier modelo de Raspberry para la captura de imágenes o videos en HD. Esta cámara posee un sensor de imagen IMX219PQ de Sony, el cual ofrece imágenes de video de alta velocidad y alta sensibilidad, además con enfoque fijo puede llegar a una resolución de hasta 8 megapíxeles. En la imagen (Ilustración 21 - Cámara Raspberry Pi V2) se puede apreciar esta cámara.

Ilustración - Cámara Raspberry Pi V2

* *Pantalla táctil LCD para Raspberry Pi de 7”:* Es la pantalla táctil oficial de la plataforma (Ilustración 22 - Pantalla táctil de Raspberry Pi). Se trata de una pantalla táctil LCD capacitiva multitáctil (de hasta 10 puntos de contacto). El display de 7 pulgadas posee una resolución de 800x480 píxeles con una velocidad de refresco de 60 fps ( fotogramas por segundo) y color RGB de 24 bits. Se conecta a través de una placa adaptadora que se ocupa de la conversión de potencia y señal. Sólo se requieren dos conexiones a la Pi; la de energía a través del puerto GPIO del Pi y un cable de cinta que se conecta al puerto DSI (Display Serial Interface) presente en todo modelo de Raspberry Pi.

Ilustración - Pantalla táctil de Raspberry Pi

* *Kit de Placa de prototipado de Pi de Adafruit (Adafruit Prototyping Pi Plate Kit):* Se trata de una placa que se encastra en la parte superior de las Raspberry Pi, en la cual se pueden soldar componentes en su área de GPIO (entrada/salida de propósito general) y además cuenta en su centro con un área de protoboard[[14]](#footnote-10). En la imagen (Ilustración 23 - Adafruit Prototyping Pi) se puede ver esta placa empalmada sobre una Raspberry Pi

Ilustración - Adafruit Prototyping Pi

* *Western digital Pidrive:* Es un disco rígido (Ilustración 24 - Pidrive) exclusivo para esta plataforma, de una capacidad de 314 GB, creado por la marca homónima. Cuenta con una interfaz de conexión USB para comunicarse con la Raspberry Pi.

Ilustración - Pidrive

* *Pi TFT:* Es una pequeña pantalla táctil de 2.8 pulgadas del tipo resistiva (Ilustración 25 - Pi TFT), que se encastra en la parte superior del Raspberry. Su resolución es de 320x240 y color de 16 bits. Se le pueden soldar 4 botones de forma opcional para su manipulación.

Ilustración - Pi TFT

## **4.5 ¿Por qué elegir Raspberry Pi?**

Al igual que lo que se mencionó en el capítulo 3 con respecto a Arduino, la plataforma Raspberry Pi presenta una serie de ventajas, con respecto a otras arquitecturas similares, que se describen a continuación:

* **Comunidad**: Existe una vasta comunidad en variadas partes del mundo que trabaja, da soporte y utiliza esta plataforma para diversos proyectos[[15]](#footnote-11), que dado esto, se expanden con él tiempo. A su vez, como se mostró en el apartado anterior, se cuenta con una serie de accesorios que facilitan su uso.
* **Bajo costo**: Como se mencionó con anterioridad, esta SBC se puede conseguir a un bajo costo teniendo en cuenta las prestaciones que posee.
* **Desarrollada con finalidad educativa**: Como ya se comentó anteriormente, según sus creadores, esta plataforma fue desarrollada con fines educativos y existe una comunidad que constantemente aporta lo necesario para trabajar con ella en el aula.
* **Interfaces y GPIO:** Cuenta con una variedad de interfaces para la conexión de distintos periféricos (HDMI, USB, Ethernet, Wifi, Bluetooth) y a su vez, los modelos más actuales (la versión 3), vienen con 40 pines del tipo GPIO, lo que lo convierte en un SBC muy versátil en cuanto a su utilidad.
* **Prestaciones**: Explicado todo lo anterior en este capítulo, podemos concluir que esta plataforma cumple con las prestaciones necesarias pretendidas en el desarrollo de esta tesina.

<https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/>

# Capítulo 5 - Aplicaciones Móviles

## **5.1 ¿Qué son las aplicaciones móviles?**

“Una aplicación móvil es un programa que se puede descargar y/o acceder directamente desde un aparato móvil”

## **5.2 Las App’s**

“Una aplicación móvil o app es una aplicación informática diseñada para ser ejecutada en teléfonos inteligentes, tabletas y otros dispositivos móviles y que permite al usuario efectuar una tarea concreta de cualquier tipo: profesional, de ocio, educativas, de acceso a servicios, etc; Facilitando las gestiones o actividades a desarrollar”

Al ser aplicaciones residentes en los dispositivos, están escritas mayormente en Java (Android) , Objective-C (IOS) y Windows Phone (C#). Su funcionamiento y recursos se encaminan a aportar una serie de ventajas tales como:

Ilustración - Aplicaciones móviles

* Un acceso más rápido y sencillo a la información necesaria sin necesidad de los datos de autenticación en cada acceso.
* Un almacenamiento de datos personales que, a priori, es de una manera segura. Esto quiere decir (Android) que utiliza un sistema de permisos heredado de Linux y conocido como la Sandbox. Este sistema limita en gran medida el acceso al sistema de archivos e impide que los procesos puedan acceder a los recursos de otros procesos, como la memoria y la CPU. Sin embargo, existen procesos con privilegios elevados que pueden acceder a recursos del dispositivo sin problemas, principalmente servicios del sistema y aplicaciones preinstaladas en los dispositivos.[[16]](#endnote-5) Los permisos son declarados en un archivo denominado AndroidManifest.xml . Estos permisos definen los recursos del dispositivo que van a utilizar y que el usuario debe aceptar antes de instalarlas, como el acceso a la cámara, leer o escribir en el almacenamiento externo, el acceso a internet, etc. Android impide que las apps utilicen recursos no manifestados, provocando un fallo en las mismas o su cierre, aunque esto solo sucede durante el desarrollo de las aplicaciones y no en su distribución. [[17]](#endnote-6)
* Una gran versatilidad en cuanto a su utilización o aplicación práctica. Dada la facilidad que presenta a la hora de la utilización del usuario, dado que necesita instalar la aplicación y mantenerla actualizada. No necesita de un navegador para poder acceder.
* La atribución de funcionalidades específicas. Esto se debe a que las app nativas acceden a recursos hardware con mayor facilidad (cámara, contactos, memoria, notificaciones push, etc.)
* Mejorar la capacidad de conectividad y disponibilidad de servicios y productos (usuario-usuario, usuario-proveedor de servicios).

### 5.2.1 Las webApps

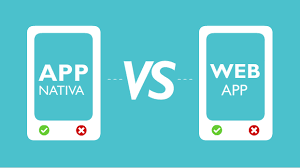
****Una web app no es más que una versión de página web optimizada y adaptable, a cualquier dispositivo móvil independientemente del sistema operativo que utilice. Esta optimización es posible gracias al lenguaje HTML5, combinado con hojas de estilo CSS3, que permiten dicha adaptabilidad, denominada “Responsive Web Design”.

Ilustración - App nativa vs Web App

### 5.2.2 Ventajas de las Web-App:

* No ocupa espacio de memoria en los dispositivos (no es una app).
* No requiere actualizaciones ya que al ser una página web siempre se accede a la última versión.
* No consume recursos dado que no instala servicios en segundo plano y además no consume espacio dado que no es necesario instalar la aplicación para su uso.
* El proyecto de implementación de una Web-App es más económico que el de una App.

### 5.2.3 Desventajas de las Web-Apps

* No permite la promoción y distribución a través de los markets (Playstore, Nokia store, App Store, Windows Phone Apps)
* Requiere de una conexión entre el cliente y el servidor (por ejemplo por internet o una LAN).
* Menor usabilidad, al no poder aprovecharse de los recursos del propio dispositivo (geolocalización, notificaciones “push”).
* Carece de un icono de lanzamiento específico. Es necesario un navegador

Ilustración – WebApps – Diseño multipropósito

## **5.3 ¿Cuáles son los distintos sistemas operativos para dispositivos móviles?**

Un sistema operativo móvil o SO móvil es un conjunto de programas de bajo nivel que permite la abstracción de las peculiaridades del hardware específico del teléfono móvil y provee servicios a las aplicaciones móviles, que se ejecutan sobre él. Al igual que los PCs que utilizan Windows, Linux o Mac OS, los dispositivos móviles tienen sus sistemas operativos como Android, IOS o Windows Phone, entre otros.

A medida que los teléfonos móviles crecen en popularidad, los sistemas operativos con los que funcionan adquieren mayor importancia. La cuota de mercado de sistemas operativos móviles en el primer trimestre de 2016 era el siguiente:

* Android 84,1 %
* iOS 14,8 %
* Windows Phone 0,7 %
* BlackBerry OS 0,2 %
* Otros 0,2 %

Android tiene la mayor cuota, desde enero 2011, con más de la mitad del mercado, experimentó un creciente aumento y en solo dos años (2009 a comienzos de 2011) ha pasado a ser el SO móvil más utilizado.

Es por esto, que en principio se pensó desarrollar la App que se comunica con el SAR y por ende al RM, en Android.

## **5.4 Android**

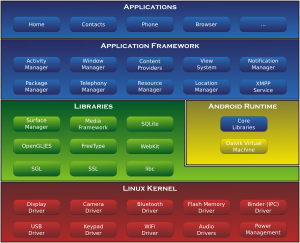
Es sin duda el líder del mercado móvil en sistemas operativos, está basado en Linux, diseñado originalmente para cámaras fotográficas profesionales, luego fue vendido a Google y modificado para ser utilizado en dispositivos móviles como los teléfonos inteligentes y luego en tablets. Actualmente se encuentra en desarrollo para usarse en netbooks y PCs; además de la creación de la Open Handset Alliance, compuesto por 78 compañías de hardware, software y telecomunicaciones dedicadas al desarrollo de estándares abiertos para celulares, esto le ha ayudado mucho a Google a masificar el S.O, hasta el punto de ser usado por empresas como HTC, LG, Samsung, Motorola entre otros.

Ilustración - Arquitectura de Android



Las aplicaciones para Android se escriben y desarrollan en Java aunque con unas APIs propias, por lo que las aplicaciones escritas en Java para PC y demás plataformas ya existentes no son compatibles con este sistema.

Ilustración - Logo de Android

## **5.5 Aplicaciones móviles multiplataforma**

### 5.5.1 Diferencias entre aplicaciones y web móviles

Las aplicaciones comparten la pantalla del teléfono con las webs móviles, pero mientras las primeras tienen que ser descargadas e instaladas antes de usar, a una web puede accederse simplemente usando Internet y un navegador compatible con la aplicación web. Pero, no todas pueden verse correctamente desde una pantalla generalmente más pequeña que la de un ordenador de escritorio.

Las que se adaptan especialmente a un dispositivo móvil se llaman “web responsiva” y son ejemplo del diseño líquido (adaptativo), ya que se puede pensar en ellas como un contenido que toma la forma del contenedor, mostrando la información según sea necesario.

### 5.5.2 Aplicaciones Nativas

Una aplicación nativa es la que se desarrolla de forma específica para un determinado sistema operativo, llamado Software Development Kit o SDK. Cada una de las plataformas, Android, iOS o Windows Phone, tienen un sistema diferente, por lo que si se desea que una app esté disponible en todas las plataformas se deberán de crear varias apps con el lenguaje del sistema operativo seleccionado.

* Las apps para iOS se desarrollan con lenguaje Objective-C
* Las apps para Android se desarrollan con lenguaje Java, sobre API’s
* Las apps en Windows Phone se desarrollan en .Net

Cuando hablamos de desarrollo móvil casi siempre nos estamos refiriendo a aplicaciones nativas. La principal ventaja con respecto a los otros dos tipos, es la posibilidad de acceder a todas las características del hardware del móvil: cámara, GPS, agenda, dispositivos de almacenamiento y la falta de necesidad de estar conectado a internet para que funcione. Por otro lado, existe la promoción de las Apps por medio de las tiendas de aplicaciones (app store). Las aplicaciones nativas presentan un mayor rendimiento y aprovechamiento del hardware de cada smartphone o dispositivo móvil.

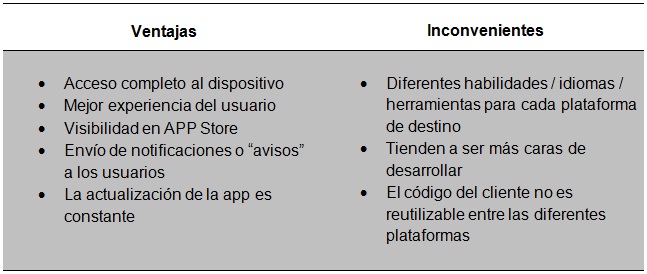


Ilustración - Cuadro comparativo nativas

### 5.5.3 Aplicaciones Web

Una aplicación web o webapp es la desarrollada con lenguajes muy conocidos por los programadores, como pueden ser HTML, Javascript y CSS.

La principal ventaja con respecto a la nativa es la posibilidad de programar independientemente del sistema operativo en el que se usará la aplicación. De esta forma se pueden ejecutar en diferentes dispositivos sin tener que crear varias aplicaciones.

Las aplicaciones web se ejecutan dentro del propio navegador web del dispositivo a través de una URL.

La gran diferencia con una aplicación nativa es que no necesita instalación por lo que no pueden estar visibles en app store y la promoción y comercialización debe realizarse de forma independiente. De todas formas, se puede crear un acceso directo o link para acceder a la aplicación en el dispositivo.

Las apps web móviles son siempre una buena opción si nuestro objetivo es adaptar la web a formato móvil.



Ilustración - Cuadro comparativo - Aplicaciones Web

### 5.5.4 Aplicaciones Híbridas

Una aplicación híbrida es una combinación de las dos anteriores. Las apps híbridas se desarrollan con lenguajes propios de las webabpp, es decir, HTML, Javascript y CSS por lo que permite su uso en diferentes plataformas, pero también dan la posibilidad de acceder a gran parte de las características del hardware del dispositivo. La principal ventaja es que es posible agrupar los códigos y distribuirla en app store.

PhoneGap es uno de los frameworks más utilizados por los programadores para el desarrollo multiplataforma de aplicaciones híbridas. Otro ejemplo de herramienta para desarrollar apps híbridas es Cordova. También existe la opción de servirse de cross-platforms frameworks, para desarrollar una web app.

### 5.5.5 Aplicación híbrida: Web App

Cosiste en diseñar la aplicación como si fuera una web app, y será el propio navegador el que la ejecute. Sin embargo, ésa es precisamente su gran virtud, pero también su principal inconveniente. Nos encontramos con que el código es válido para todos los dispositivos, pero la sensación del usuario y la apariencia no es tan buena como en la app nativa. Phonegap es el framework más popular en esta tecnología.

### 5.5.6 Aplicación híbrida: app interpretada

La aplicación interpretada significa que la aplicación es programada y luego cada terminal la traduce a su propio lenguaje de programación. Facilita el desarrollo de aplicaciones y reduce el esfuerzo considerablemente. Aunque el resultado no es idéntico a la nativa, la apariencia es bastante buena, y en muchas ocasiones puede ser la solución al problema del desarrollo de aplicaciones multiplataforma.

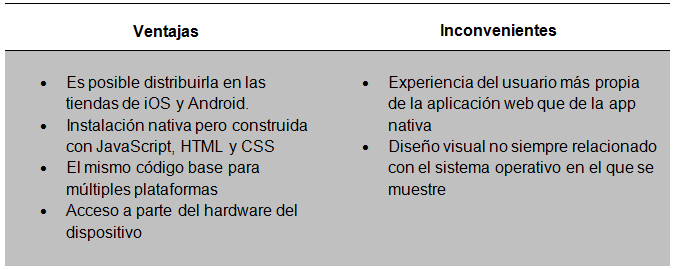


Ilustración - Comparativa aplicaciones híbridas

## **5.6 Investigación en Desarrollo de Apps y WebApps**

Al momento de seleccionar el tipo de App o webApp se investigó en las distintas tecnologías para seleccionar la adecuada

Ilustración - Herramientas para desarrollo de apps

### 5.6.1 Android Studio

Es el IDE oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA . Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan la productividad durante la compilación de apps para Android. Además, posibilita el uso de emuladores con distintos niveles y hardware específico para testear las app creadas.

### 5.6.2 App inventor

Es un entorno de desarrollo de software creado por Google Labs para la elaboración de aplicaciones destinadas al sistema operativo Android. El usuario puede, de forma visual y a partir de un conjunto de herramientas básicas, ir enlazando una serie de bloques para crear la aplicación. El sistema es gratuito y se puede descargar fácilmente de la web. Las aplicaciones creadas con App Inventor están limitadas por su simplicidad, aunque permiten cubrir un gran número de necesidades básicas en un dispositivo móvil.

Con Google App Inventor, se espera un incremento importante en el número de aplicaciones para Android debido a dos grandes factores: la simplicidad de uso, que facilitará la aparición de un gran número de nuevas aplicaciones; y Google Play, el centro de distribución de aplicaciones para Android donde cualquier usuario puede distribuir sus creaciones libremente.

Otra gran cualidad es la posibilidad de insertarlo en la educación dado su programación por medio de bloques gráficos, que resultan ser muy intuitivos en aquellas personas que se introducen.

Unas primeras apps para el SAR fueron realizadas bajo esta plataforma, concluyendo que tiene una curva de aprendizaje corta.

### 5.6.3 Intel XDK

Es un kit de desarrollo creado por Intel para crear aplicaciones nativas para los teléfonos celulares y las tabletas que utilizan tecnologías de la web como HTML5, CSS y Javascript. Las aplicaciones están compiladas vía on-line en la plataforma Cordova para hacer aplicaciones cross-platform, pensando en crear apps para IOT. Posee un emulador, App preview mediante QR, Drag and Drop y soporte de plantillas. Además es compatible y da soporte para Android, IOS, Windows Phone y entre otros.

### 5.6.4 Ionic

Es una herramienta (framework), gratuita y open source, para el desarrollo de aplicaciones híbridas, inicialmente pensado para móviles y tablets, basadas en HTML5, CSS y JS. Está construido con Sass y optimizado con AngularJS, permitiendo una mayor facilidad de desarrollo de aplicaciones, con una menor inversión económica y la creación de aplicaciones con una misma base de código.

### 5.6.5 Cordova

Apache Cordova es un entorno de desarrollo de aplicaciones móviles, originalmente creado por Nitobi y comprado por Adobe. Más tarde fue liberado como Apache Cordova. Permite, construir aplicaciones para dispositivos móviles utilizando CSS3, HTML5, y Javascript. Las aplicaciones resultantes son híbridas, lo que significa que no son ni una aplicación móvil nativa (porque toda la representación gráfica se realiza vía vistas de Web en vez del framework nativo) ni puramente basadas en web (porque no son solo aplicaciones web, sino que están empaquetadas como aplicaciones para su distribución y tienen acceso a las APIs nativas del dispositivo).

### 5.6.6 Meteor

Es una plataforma para crear aplicaciones web en tiempo real construida sobre Node.js. Meteor se localiza entre la base de datos de la aplicación y su interfaz de usuario y se encarga que las dos partes estén sincronizadas.

Como Meteor usa Node.js, se utiliza JavaScript en el cliente y en el servidor. Y más aún, Meteor es capaz de compartir código entre ambos entornos.

### 5.6.7 Meteor y Cordova

Meteor se integra con Cordova, por ende es posible tomar una aplicación existente y ejecutarla en un dispositivo iOS o Android de forma sencilla. Una aplicación de Cordova es una aplicación web escrita usando HTML, CSS y JavaScript como de costumbre, pero se ejecuta en una vista web incrustada en una aplicación nativa en lugar de en un navegador móvil independiente. Un beneficio importante de empaquetar su aplicación web como una aplicación de Cordova es que todos sus activos se incluyen con la aplicación. Esto asegura que su aplicación se cargue más rápido que una aplicación web que se ejecuta en un servidor remoto, lo que puede hacer una gran diferencia para los usuarios con conexiones lentas. Otra característica de la integración de Cordova en Meteor es la compatibilidad con hot code push, que le permite actualizar su aplicación en los dispositivos de los usuarios sin pasar por el proceso habitual de revisión de la tienda de aplicaciones. Cordova también abre el acceso a ciertas características nativas a través de una arquitectura de complemento. Los complementos permiten utilizar funciones que normalmente no están disponibles para aplicaciones web, como acceder a la cámara del dispositivo o al sistema de archivos local, interactuar con lectores de código de barras o NFC. Debido a que una aplicación de Cordova es una aplicación web, significa que utiliza elementos web estándar para crear su interfaz de usuario en lugar de confiar en componentes de interfaz de usuario nativos específicos de la plataforma.

# Capítulo 6 – Stack MEAN

## **6.1 ¿Qué es MEAN?**

“Es un framework o conjunto de subsistemas de software para el desarrollo de aplicaciones, y páginas web dinámicas, que están basadas, cada una de estas en el popular lenguaje de programación conocido como JavaScript. Gracias a esta característica el conjunto se integra exitosamente en una plataforma auto-suficiente”.



Ilustración 31 - Acrónimo MEAN

## **6.2 ¿Cómo se compone MEAN?**

### 6.2.1 Mongo

Es un sistema de base de datos NoSQL, que almacena los datos en estructuras o “documentos”, los cuales están definidos con la notación JSON (Notación simple de objeto tipo JavaScript), lo que permite una rápida manipulación y transferencia de los datos.

### 6.2.2 Express

Es un módulo de NodeJS, ofrece los métodos suficientes en JavaScript, para poder manejar las solicitudes o peticiones que se hacen por medio de los métodos del protocolo. También ofrece un sistema simple de enrutamiento (Routing), que dentro del mean stack es aprovechado en el back-end o en el lado del servidor.

### 6.2.3 Angular

Es un framework para aplicaciones web de TypeScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.

### 6.2.4 Node

Es la plataforma encargada del funcionamiento del servidor, y trabaja totalmente con JavaScript.

Ilustración - Logo del motor V8

Node.js utiliza por debajo el motor de JavaScript de Google, denominado V8, y provee de una arquitectura orientada a eventos (como la de los navegadores) así como una serie de APIs no-bloqueantes (asíncronas) que le proporcionan un rendimiento y una escalabilidad muy elevadas.

Se puede utilizar para crear cualquier tipo de lógica de aplicación, pero dado que incorpora un módulo para poder actuar como un servidor web, es especialmente popular para crear aplicaciones web. Actualmente lo emplean para sus aplicaciones multitud de empresas de todos los ámbitos, pero especialmente de Internet.

## **6.3 Otros complementos**

### 6.3.1 Twitter Bootstrap

El más popular framework que integra HTML, CSS, y JS para el desarrollo de proyectos adaptables a diferentes medios de reproducción (Responsive).

### 6.3.2 Compodoc

Documentador JavaScript estático.

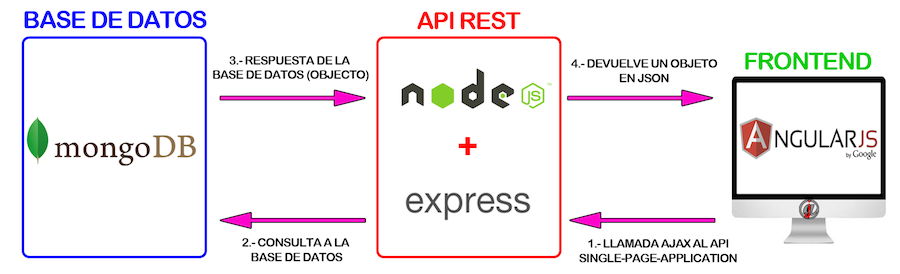


Ilustración - Arquitectura de interacción MEAN

### 6.3.3 Json

Es también a su vez una palabra formada por iniciales de palabras, y significa: JavaScript Simple Object Notation.

Como su propio nombre indica JSON (pronunciado "yeison") permite representar objetos (en realidad estructuras complejas) en forma de código JavaScript que luego podemos evaluar.

Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos es que es mucho más sencillo escribir un analizador sintáctico (parser) de JSON. En JavaScript, un texto JSON se puede analizar fácilmente usando la función eval(), lo cual ha sido fundamental para que JSON haya sido aceptado por parte de la comunidad de desarrolladores AJAX, debido a la ubicuidad de JavaScript en casi cualquier navegador web.

Ilustración - Logo de JSON

“JSON es el pegamento de todas las capas. Es el formato en el que se transfieren los datos entre todos los niveles de la aplicación: navegador, servidor web y servidor de datos”

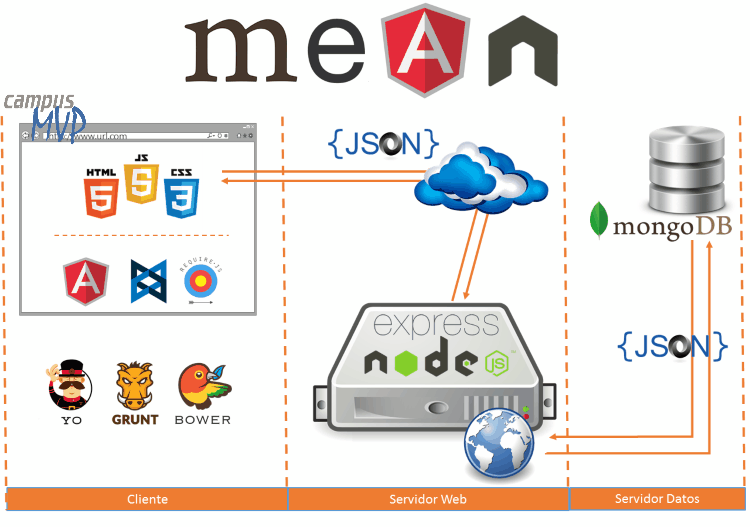


Ilustración - JSON pegamento de tecnologias

# Capítulo 7 – Librería Johnny-five y el protocolo Firmata

## **7.1 ¿Qué es Johnny-five?**

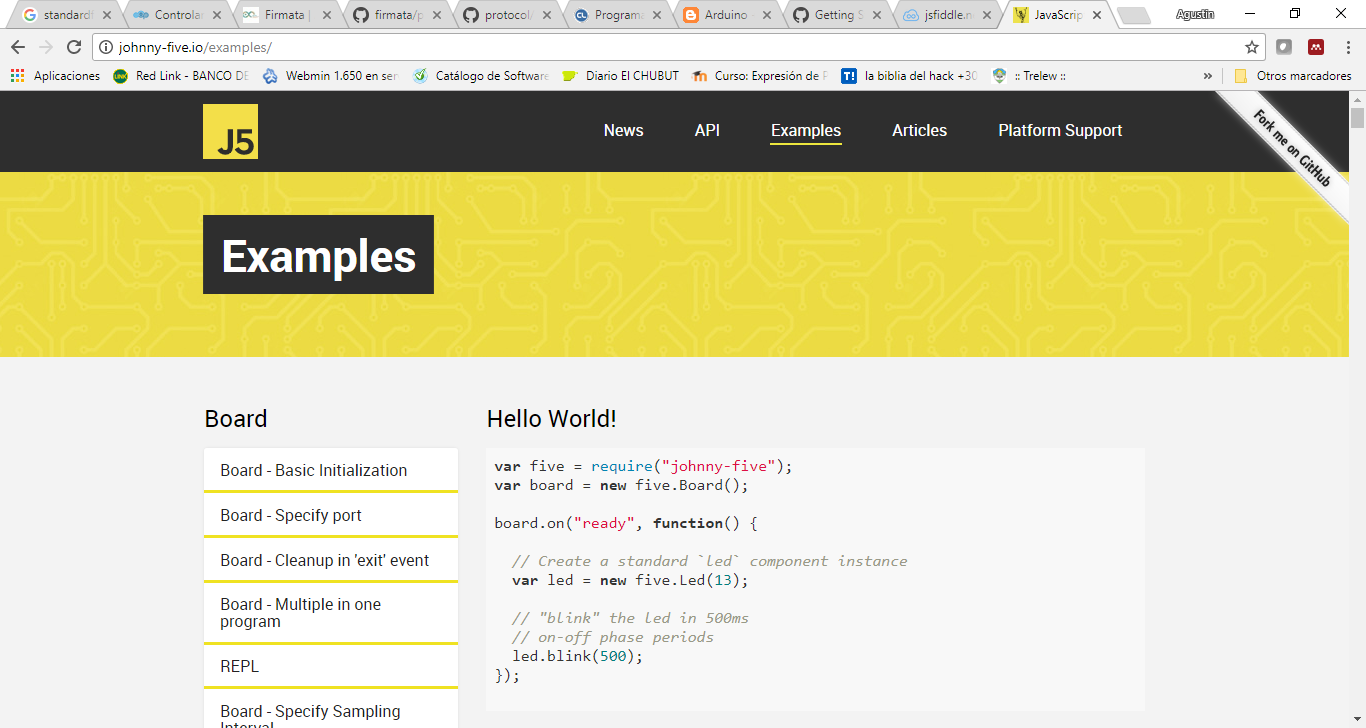
Johnny-five, más conocido en le jerga informática como J5, es un framework de programación robótica basado en javascript lanzado por la compañía de tecnología web Bocoup en el 2012, es de código abierto, por lo cual cuenta con una gran variedad de desarrolladores, ingenieros y colaboradores que están constantemente perfeccionando y agregando características nuevas a esta herramienta.

Ilustración - Página oficial de Johnny-Five

Es compatible y da soporte a la mayoría de los microcontroladores y SBC más populares como, los utilizados en el desarrollo de esta tesina, Arduino y Raspberry Pi (ambos en todas sus versiones). Además de placas como BeagleBone, SparkFun, ChipKit, Intel-Galileo, entre otras.

Posee una librería muy completa, con compatibilidad a una vasta cantidad de sensores y actuadores, y ejemplos útiles de usos e implementación de los mismos. Dichos ejemplos se encuentran disponibles en su página oficial con el esquema de conexión de los componentes a las distintas plataformas compatibles.

## **7.2 Instalación**

Para poder utilizar Johnny-Five, se debe contar con el Framework node.js (visto en el capítulo 6) instalado en el Sistema Operativo del computador.

La instalación de esta librería es sencilla, dado que la misma se puede agregar al paquete de librerías que se obtienen al crear un proyecto de node.js. Para ello se debe ejecutar el siguiente comando, en una consola, dentro del directorio del proyecto:

*npm install Johnny-five*

Como se puede apreciar, se utiliza el comando npm que no es más que el gestor de módulos y aplicaciones de node.js

## **7.3 Arduino Firmata**

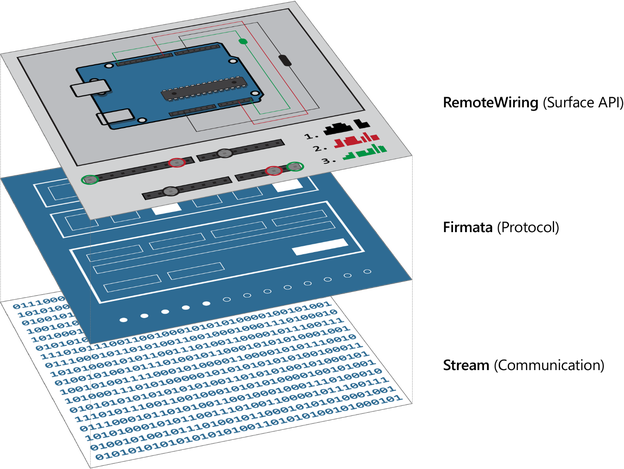


Ilustración - Remote Wiring de Windows

Firmata es un protocolo genérico utilizado para la comunicación con microcontroladores desde software instalado en una computadora. Este protocolo se puede implementar en cualquier arquitectura de microcontroladores, así como en cualquier paquete de software.

El objetivo de firmata es permitir controlar completamente un micricontrolador, por ejemplo Arduino, desde un programa instalado en una computadora, sin escribir código de Arduino.

Ventajas:

* Nuestro programa no está limitado por la memoria RAM y Flash de Arduino
* El software de control se puede programar en cualquier lenguaje, no solo C++, siempre que tenga soporte para firmata. Por ejemplo: Firmata: Processing, Visual Basic, Perl, C#, PHP, Java, **JavaScript**, Ruby y por su puesto Python.

Desventajas:

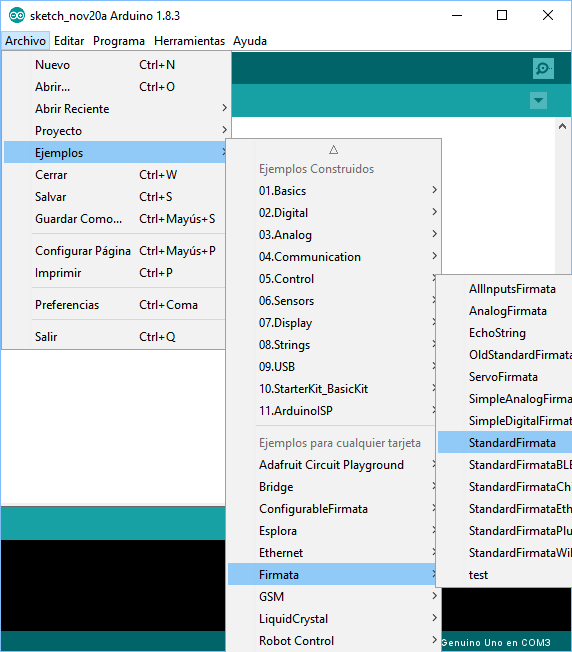
* Programas más restringidos, es posible que algunas operaciones complejas no sea posible hacerlas. Por ejemplo, el uso de interrupciones.
* El Arduino siempre debe estar conectado al computador. Debe tener comunicación para poder ejecutar acciones.

Ilustración - Sketch de Arduino

## **7.4 Instalación Firmata**

Para instalar Firmata se necesita tener conectado el chip Arduino a la computadora a través de un puerto USB. Luego de ello se debe seleccionar desde el Sketch de Arduino el código Firmata, dentro de su respectiva librería, según los dispositivos que se requieran comunicar con el mismo. Para ello se debe ir a Archivo 🡪 Ejemplos 🡪 Firmata.

En nuestro caso se utilizaron dos códigos Firmata:

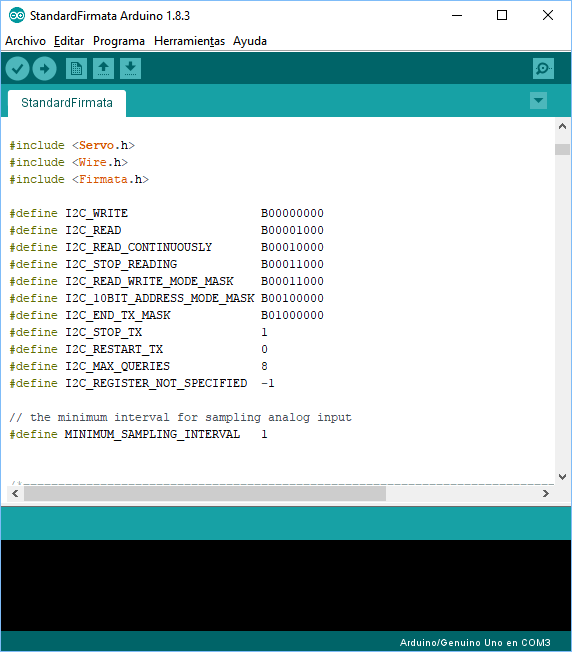


Ilustración - Código StandardFirmata

**StandardFirmata**: Es, como su nombre lo indica, el estándar del protocolo que permite la comunicación con la mayoría de los componentes compatibles con Arduino. En nuestro caso, es el utilizado para cargarlo dentro del Arduino Mega para manipular la mayoría de sensores y actuadores del SAR.

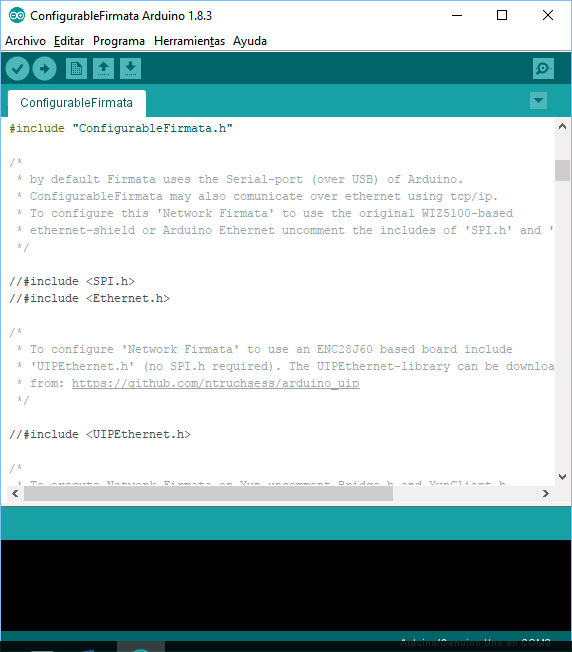


Ilustración - Código ConfigurableFirmata

**ConfigurableFirmata**: Esta versión personalizada del protocolo, es la utilizada para captar la temperatura mediante el sensor para dicho fin conectado al Arduino Nano. Permite separar las características del protocolo en clases individuales, haciendo más sencillo mezclar las características estándar del protocolo con otras personalizadas.

# Capítulo 8 - Python en Raspberry Pi

## **8.1 ¿Qué es Python?**

Se le llama Python a un lenguaje de programación multiparadigma y multiplataforma desarrollado en el año 1991 por Guido Van Rossum. A partir de marzo del 2001 la Python Software Foundation es la encargada de administrar este lenguaje. Su misión es la de fomentar el desarrollo de la comunidad Python.

Ilustración - Logo de Python

Este lenguaje de programación posee las siguientes características principales:

* Es de código abierto.
* Es multiparadigma, permite los estilos de programación orientado a objetos, imperativo y funcional.
* Permite otros paradigmas, a parte de los mencionados, con el uso de extensiones.
* Es interpretado.
* Utiliza tipado dinámico.
* Extensible.

## **8.2 Python con Raspberry Pi**

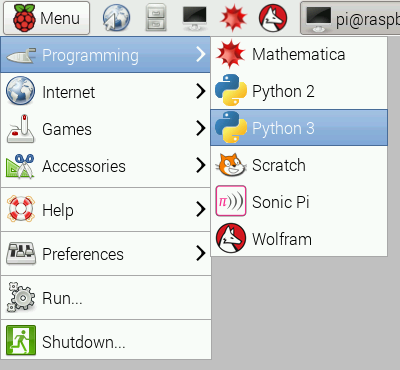
A Python se lo considera un lenguaje de programación fácil de aprender, además de ser muy popular y potente, es un lenguaje serio, usado en distintos ámbitos profesionales. Como se identificó en el apartado anterior, es un lenguaje de código abierto y multiplataforma, por lo que se puede utilizar en cualquier sistema con total libertad e incluso con fines comerciales si se quisiese.

Ilustración - Menú de Raspbian

Dicho esto, y por otros motivos, es que la Raspberry Pi Foundation lo ha seleccionado como el lenguaje de programación “estándar” para su plataforma Raspberry Pi. Lo que no significa que la misma se deba programar solo en Python. Raspberry Pi es una computadora que corre un sistema operativo, lo cual le da la facultad de correr diversos programas y por ende variados lenguajes.

Python es un lenguaje interpretado, que quiere decir que un “interprete” va leyendo las instrucciones y ejecutándolas en tiempo real. Un intérprete es un software encargado de convertir nuestras sentencias a código máquina. Puede instalarse cualquiera de los muchos intérpretes que existen para tu máquina, ya sea la Raspberry Pi o un ordenador personal.

El sistema operativo Raspbian, desarrollado específicamente para la Raspberry Pi, viene con las versiones de Python 2 y 3 pre- instaladas, con sus respectivos intérpretes.

## **8.3 Raspberry PI – GPIO**

La Raspberry Pi 3 cuenta con una cantidad de 40 pines del tipo GPIO (*General Purpose Input/Output, Entrada/Salida de propósito general*)

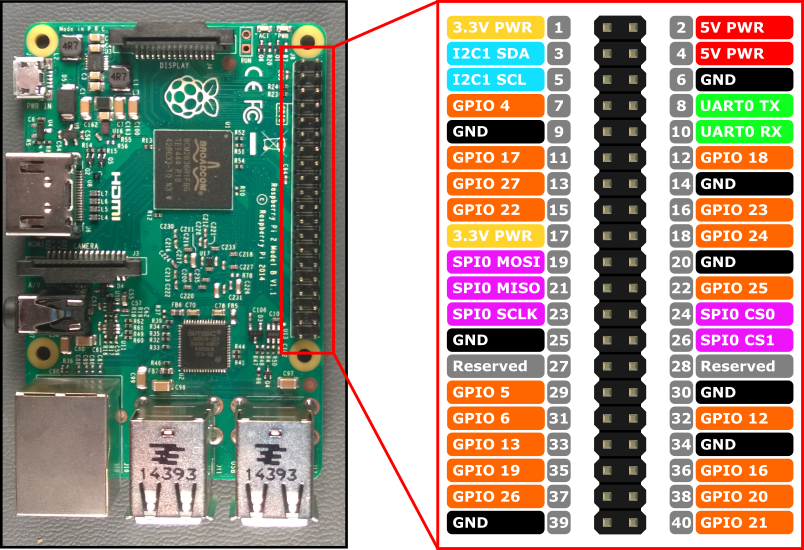


Ilustración - Raspbery Pi 2 y sus GPIOs

Con los cuales, al igual que en la plataforma Arduino, se pueden conectar y manipular un número considerable de actuadores y sensores. Cabe aclarar que todos estos pines son del tipo digital, por lo que, este computador no cuenta con pines del tipo analógico. Para la lectura de sensores que devuelvan valores analógicos se debe utilizar un convertidor externo o un Arduino.

Para manipular estos pines, existe una librería nombrada como RPi.GPIO que nos permite, mediante Python, configurarlos rápidamente.

Como se puede apreciar en la ilustración anterior podemos distinguir entre dos tipos de identificación de estos pines, según su orden físico en la placa o según su posición correspondiente a su conexión física a su CPU (en Raspberry Pi, Broadcom), estas formas de identificación se llaman BCM y BOARD respectivamente.

El número que identifica al pin en el modo BCM es el que esta después de la palabra GPIO (ver diagrama anterior), el índice para identificar los pines en modo BOARD son los que están dentro del círculo. Es importante mencionar que la identificación de los pines en modo BCM cambio entre la revisión 1 y la revisión 2 del hardware.

# Capítulo 9 - Análisis y selección de tecnologías para desarrollo del SAR

## **9.1 Primer análisis**

A partir del análisis de las distintas tecnologías hardware en microcontroladores, microprocesadores actuales y teniendo en cuenta las ventajas y desventajas, tanto en factores como rendimientos, tiempos de respuesta, consumo energético, portabilidad; examinada con la experiencia de la utilización de la familia de Arduino (Arduino Uno, Arduino Mega y Arduino Nano), PC (Ubuntu Mate), Raspberry Pi 3 Modelo B, como así también los distintos módulos arduino-compatibles como la cámara OV7670, GPS, ESP8266(Wifi), Bluetooth, entre otros; y articulando con el software dentro del desarrollo de aplicaciones móviles, permitió analizar Android con sus respectivos entornos de desarrollo como: Android Studio, Intel XDK, Ionic, App Inventor, Cordova, etc; y dada la necesidad de que el SAR debiera ser portable para cualquier dispositivo que posea una conexión Wifi, es que se decide realizar una aplicación web, seleccionando el stack MEAN.

Estas investigaciones nos permiten concluir en la siguiente selección final de tecnologías hardware y software para la creación del SAR.

## **9.2 Selección tecnologías hardware**

### 9.2.1 ¿Por qué Arduino?

Como se abordó en el capítulo 5 (Arduino), siendo una arquitectura hardware pensada para hobbistas, diseñadores y personas no relacionadas con la electrónica ni la programación a bajo nivel, Arduino permite una curva de aprendizaje más corta y la facilidad de conexión de los distintos componentes la hace muy atractiva, para encarar distintos proyectos con diversos niveles de complejidad.

Las placas Arduino, fueron pensadas mayormente para un uso del control de sensores y actuadores utilizando un microcontrolador, esto genera una mejor transición desde la electrónica discreta a la electrónica programable.

Las placas utilizadas fueron la Arduino UNO, donde se elaboraron distintos prototipos con protoboard simples, desde la manipulación de actuadores con motores hasta la toma de datos de distintos sensores como temperatura, humedad, obstáculos, entre otros. El problema encontrado en la placa Arduino UNO era la poca disponibilidad de pines E/S para la cantidad de sensores/actuadores y módulos que se requerían conectar, es por ello que se prefirió ampliar la cantidad de pines optando por la placa Arduino Mega. Esta última, otorgaba mayor cantidad de pines, pero no expandía la cantidad de memoria y procesamiento, generando dificultad a la hora de programar, sin contar con tantas interrupciones hardware y generando un pool constante en su bucle principal (loop).  Estos problemas surgieron a la hora de conectar el módulo de la cámara OV7670 y el módulo wifi ESP8266, los cuales requerían una alta cantidad de pines y nivel de cómputo.

Dado lo limitado en cuanto a poder de procesamiento y memoria, es por ello que se necesitaba otra plataforma que haga uso de las placas Arduino, resultando ser la Raspberry Pi.

### ¿Por qué Raspberry?

En el capítulo 4 se mencionó y analizó el computador de placa reducida (SBC) Raspberry Pi, en el cual se pudo apreciar las características fundamentales del mismo. Esta plataforma se diseñó, primordialmente, con fines didácticos para la educación secundaria, es por esto, que su costo es relativamente bajo. Al contar, la misma, con todas las capacidades básicas de una computadora portátil de hoy en día con su respectivo microprocesador (bastante potente para nuestras necesidades), memorias y puertos físicos (como el USB, HDMI, microSD, entre otros); y la posibilidad de instalar un sistema operativo totalmente funcional y con interfaz gráfica (en este caso Raspbian), es que se seleccionó como centro de administración y control del SAR.

Además, cuenta con pines GPIO para las conexión y manipulación de distintos módulos (como actuadores y sensores), aunque como se justifica en el apartado siguiente, es Arduino Mega y Arduino Nano quienes se encargan de estas funcionalidades, exceptuando la conexión y procesamiento de imágenes dadas por la cámara v2 exclusiva de Raspberry Pi y las comunicaciones inalámbricas proporcionadas por los módulos wifi y bluetooth integrados a este computador.

## **Comparativa entre Arduino Mega, Arduino Nano y Raspberry Pi 3 Model b**

|  |  |  |  |
| --- | --- | --- | --- |
| Factor | Arduino Mega | Arduino Nano | Raspberry Pi3 Model B |
| Microcontrolador/  Microprocesador | ATmega 1280 - 16Mhz 8bits | ATmega328 – 16Mhz 8bits | Quad Core 1.2GHz Broadcom BCM2837 64bit |
| Tensión | 5v | 5v | 5v |
| Memoria | 128 KB (Bootloader 4KB) | 32 KB (Bootloader 2KB) | 1 GB |
| Digital I/O | 54, 15 PWM | 22, 6 PWM | 40 GPIO |
| Analog I/O | 16 | 8 |
| Interfaces | USB x 1(energía) | USB x 1 (energia) | USB x 4, HDMI, CSI, DSI, MicroSD, WLAN y BLE, microUSB (Energía) |

Dada la comparativa entre las tecnologías, se decide utilizarlas como articulación entre la computación física (Arduinos) con manejo de sensores y actuadores; y por otro lado el procesamiento y comunicación proporcionado por la Raspberry.

¿Cuáles son los beneficios de esta tupla? se podría utilizar únicamente Raspberry para la elaboración del SAR, pero existen numerosos beneficios que proporciona las placas Arduino frente a la Raspberry y son:

* Menor costo de adquisición del producto en caso de fallos energéticos.
* Mayor flexibilidad y facilidad en la conexión con distintos componentes electrónicos.
* Mayor compatibilidad, con los módulos arduino-compatible y la familia adafruit
* Buen tiempo de respuesta de I/O.
* Alta confiabilidad en la lectura de sensores y en los valores de manipulación de actuadores.

Varios de estos beneficios se deben a que Arduino no posee un sistema operativo, sino un único programa que se ejecuta indefinidamente (LOOP) sin necesidad de correr algún software auxiliar que lo dispare o ejecutando como servicio; logrando concentrar su poder de procesamiento en el único programa definido.  La ejecución de servicio genera retrasos propios de sistemas operativos.

## **Cámara V2 de Raspberry Pi**

La cámara V2 de Raspberry, es una cámara exclusiva de esta plataforma la cual se conecta al puerto CSI de cualquier modelo de este computador (desde la Raspberry Pi 1 hasta el modelo actual, ósea, la Raspberry Pi 3), lo cual permite obviar la conexión pin a pin y abstraernos de la comunicación y procesamiento de la cámara. Como se comentó en el capítulo 4, es una cámara de alta definición de 8 megapíxeles, suficiente para el objetivo que se pretende con el desarrollo del SAR y saltando las problemáticas que se nos presentaron a la hora de probar la cámara OV7670 con Arduino; como el poder de procesamiento de imágenes y transmisión de las mismas (inalámbricamente) hacia otro dispositivo tal como una PC o un dispositivo móvil (en nuestro caso smartphones).

## **Módulos de Arduino**

Dentro de los módulos, sensores y actuadores de Arduino que se probaron y/o se utilizan, se encuentran:

*Utilizados en el sar:*

* El módulo GPS, será utilizado para determinar la ubicación geográfica del SAR (Geolocalización)
* Sensor de temperatura KY-001(-55° a +125°)
* Sensor ultrasonido HC-SR04 para determinar presencia de objetos a determinadas distancia y tratar de evitar el impacto con los mismos
* Motores CC para la movilidad del SAR dentro del ambiente

*Probados:*

* El módulo wifi esp8266 y el módulo Bluetooth HC-05, no se utilizarán debido a que la Raspberry Pi3 Model B, cuenta con cada uno de ellos.
* El módulo Acelerómetro MMA7361.
* Servomotor sg90.
* Sensor de evasión de obstáculos KY032.
* Sensor de golpe KY-031.
* Sensor de llamas KY-026.

# Capítulo 10 – Ensamblado del robot móvil

Para el desarrollo de esta tesina se procedió a armar un prototipo del SAR mediante un robot móvil, el cual cuenta con una variedad de actuadores y sensores que le permiten interactuar con el entorno que lo rodea.

## **Componentes**

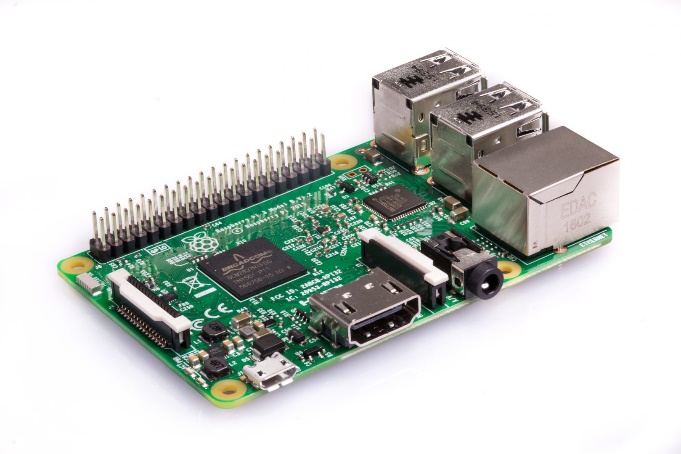
**Una Raspberry Pi 3 model B**: Componente principal del SAR, es el servidor del mismo, encargado de almacenar la aplicación web y recibir las peticiones de los clientes para luego mandar las ordenes a las placas Arduino. Cuenta con una tarje microSD donde almacena el sistema operativo Raspbian el cual se ejecuta al encenderla y permite correr la aplicación desarrollada.

Ilustración - Raspberry Pi 3

**Un Arduino Mega**: Es el principal controlador del SAR, en él se conectan todos los sensores y actuadores (a excepción del sensor de temperatura). Funciona como intermediario entre la Raspberry y el resto de los componentes, dado que recibe todas las ordenes de ejecución de la misma. En su memoria, se encuentra almacenada una versión del protocolo Firmata nombrada como StandarFirmata (dado por la librería Firmata de Arduino, ***Anexo X***) necesaria para establecer la comunicación con los comandos enviados desde Javascript por la aplicación web.

Ilustración - Arduino Mega

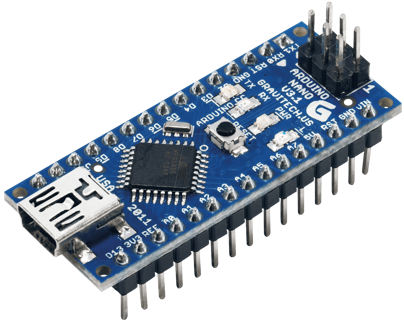
**Un Arduino Nano**: Esta versión de Arduino es la que se encarga de capturar la temperatura obtenida por el sensor DS18B20. Se debió optar por el uso de otro Arduino, dado que para la captura de temperaturas y el envío de los datos a la Raspberry mediante javascript se necesita una versión particular del protocolo Firmata, nombrada como ConfigurableFirmata (***Anexo X1***).

Ilustración - Arduino Nano

**Cuatro motores DC (corriente continua de 3v a 6v) con caja reductora:** Estos motores, en conjunto con cuatro ruedas de plástico cubiertas con una goma cada una, son los que permiten darle la movilidad al SAR.

Ilustración - Motores CC

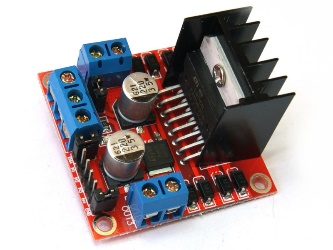
**Tres sensores ultrasónicos HC-SR04**: Los sensores ultrasónicos, se utilizan para determinar la presencia de algún objeto a una distancia menor a 20 centímetros, tanto al frente del SAR como en sus laterales. Al identificar un objeto a una distancia menor a la mencionada, se bloquea el avance del robot en la dirección en donde se encuentre dicho objeto.

Ilustración - Sensor de ultrasonido



Ilustración - Portapilas

**Dos portas pilas AA x4 con sus respectivas pilas recargables:** Utilizados para alimentar de corriente eléctrica a los 4 motores.



**Dos puentes H L298N:** Son los intermediarios entre el Arduino Mega y los motores, cada uno de ellos se encarga de la manipulación de dos motores.



Ilustración - Módulo Puente H

**Una mini Protoboard:** Utilizada como extensión de pines, más que nada para los pines GND y 5v de la placa Arduino Mega.

Ilustración - Mini-protoboard

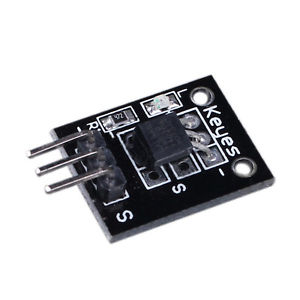
**Un sensor de temperatura DS18B20 montado sobre una placa KY-001:** Este módulo es el encargado de sensar la temperatura, se encuentra conectado al Arduino Nano.

Ilustración - Sensor de Temperatura



Ilustración - MQ7 CO

**Un sensor de monóxido de carbono MQ-7:** El sensor de monóxido, conectado al Arduino Mega, detecta la ausencia o presencia de dicho gas.

**Un GPS GY-GPS6MV2:** Con este módulo de GPS obtenemos toda la información necesaria con respecto a la Geolocalización del SAR (latitud, longitud, punto cardinal, velocidad, orientación, fecha y hora).

Ilustración - GPS

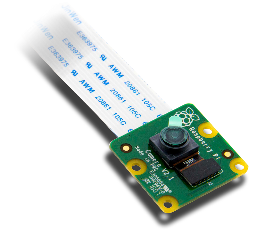
**

Ilustración - Cámara V2

**Cámara de Raspberry Pi V2:** Esta cámara, exclusiva de Raspberry, es la utilizada para captar con señal de video en tiempo real (mediante el software motion) el entorno que rodea al SAR.



**PowerBank Malibu de 20Ah con panel solar:** Funciona como batería del SAR, provee de corriente eléctrica a la Raspberry y por ende a los arduinos conectados a ella.

## **Estructura**

Ilustración - Panel Solar Power Bank

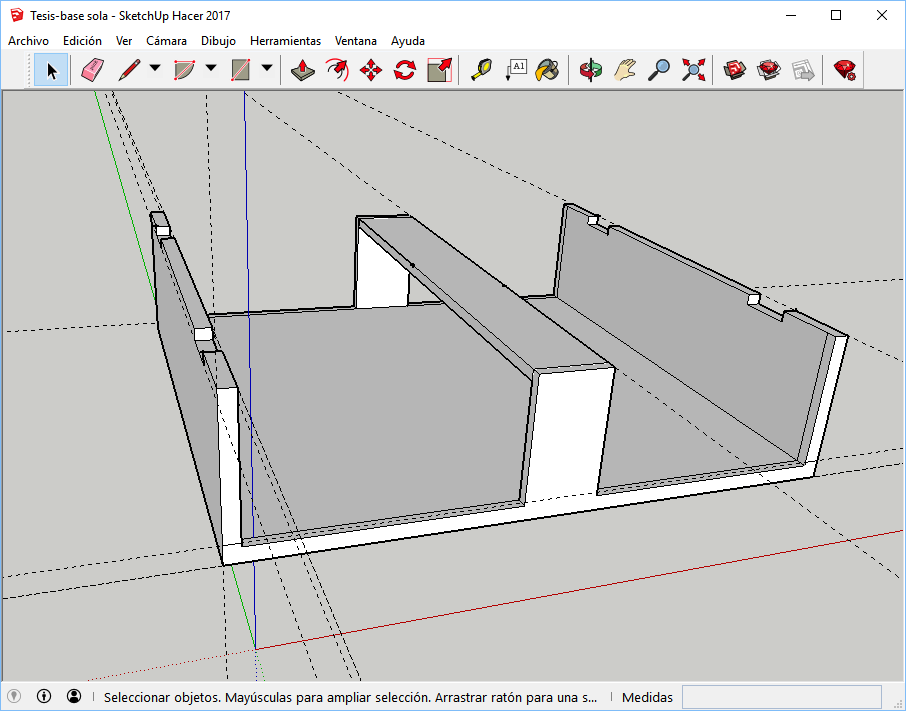
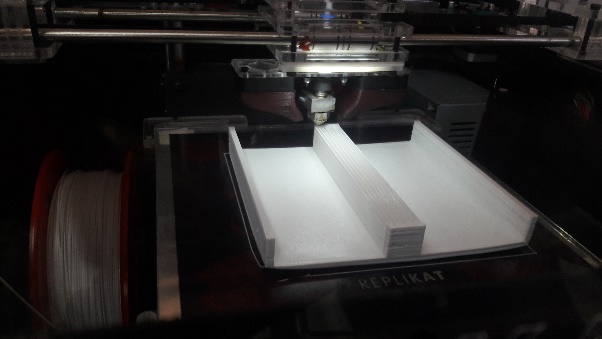


Ilustración - Diseño estructura SketchUp

Para el armado de la estructura se procedió a diseñar en 3D las distintas piezas por medio del entorno de diseño gráfico SketchUp 2017. Se tomaron medidas de los distintos componentes y en base a ellas decidimos dividir el gabinete del SAR en cuatro niveles.

Una vez armados los modelos de los distintos niveles, fuimos imprimiendo los mismos mediante una impresora 3D.

Ilustración - Impresión 3D

**Nivel 1**: El primer nivel es en donde se instalaron los motores, con distintas piezas estructurales metálicas diseñadas exclusivamente para dicha función, además se encuentran los dos puentes H L298N conectados a cada par de motores respectivamente. Cada motor cuenta con su rueda de plástico.

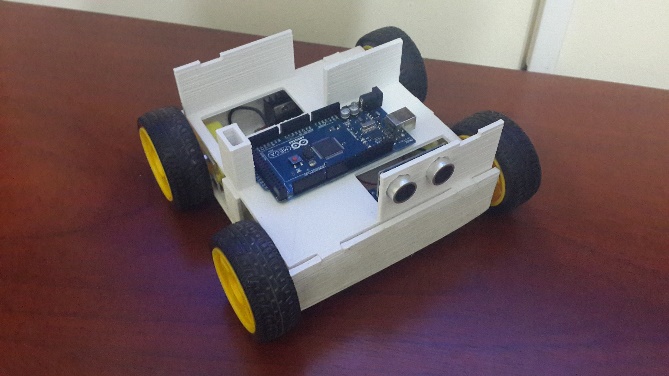
**Nivel 2**: En este nivel se adhirió con tornillos el Arduino Mega y la mini protoboard cada uno con sus respectivas conexiones, además de los porta pilas utilizados para la alimentación de los motores. En su frente se colocó uno de los sensores ultrasónicos HC-SR04 que verifica la presencia de objetos en la parte delantera del SAR.

Ilustración - Nivel 2 descubierto

**Nivel 3**: En el tercer nivel se encuentra la Raspberry Pi y el Arduino nano, en conjunto con una gran variedad de sensores, tales como, 2 sensores HC-SR04, uno en cada uno de los laterales para verificar objetos en dichos lugres. El sensor de monóxido de carbono MQ-7. La cámara de Raspberry en el frente del SAR.

Ilustración - RM Vista Lateral

**Nivel 4**: El nivel superior es el que se equipa con la batería portátil solar y que se conecta directamente a la Raspberry, además se encuentra a la vista el GPS y el sensor de temperatura DS18B20. Este último conectado al Arduino Nano.

**Falta agregar esquemas fritzing**

# Capítulo 11 – Desarrollo del SAR

El desarrollo del SAR se descompone en varios niveles de capas. Por un lado, existen dos esquemas muy diferenciados el lógico y el físico.

El esquema físico se compone de los dispositivos electrónicos que controlan los actuadores, efectores y sensores. Además de los microcontroladores (Arduino Mega, Arduino Nano) y la microcomputadora Raspberry Pi 3.

El esquema lógico se compone del sistema operativo Raspbian y una aplicación web desarrollada bajo la arquitectura cliente/servidor respetando el conjunto de herramientas MEAN. Además contiene un administrador de servicios para Node denominado PM2, y un controlador de cámaras de video conectadas al Sistema Operativo llamado Motion.

Este esquema a su vez se encuentra organizado en dos unidades funcionales llamadas Back-end y Front-end.

## Esquema general del SAR (lógico /físico)

Ilustración - Esquema general del SAR

## Desarrollo de la aplicación (front-end)

El front-end se encuentra desarrollado en Angular 4+, contando con los siguientes esquemas:

(Pegar esquemas del Compodoc)

Al conectarnos al servidor, desde un cliente (Browser) nos descarga la aplicación embebida, y la comunicación con el servidor sucede a través de Json.

## Desarrollo del servidor (back-end)

El back-end, construido en Node y Express, almacena los estáticos del front-end y los suministra al conectarse un cliente. Los directorios del proyecto son: COLOCAR DIRECTORIOS

Express filtra con las rutas AGREGAR RUTAS resolviendo y respondiendo las distintas consultas realizadas por la App en Angular.

Node es administrado por PM2 (administrador de procesos para JavaScript) el cual inicia el servidor, automáticamente al arrancar Raspbian, controlando y monitoreándolo.

Ilustración - Logo PM2

MOSTRAR PM2 EN FUNCIONAMIENTO

## Esquema de la arquitectura lógica

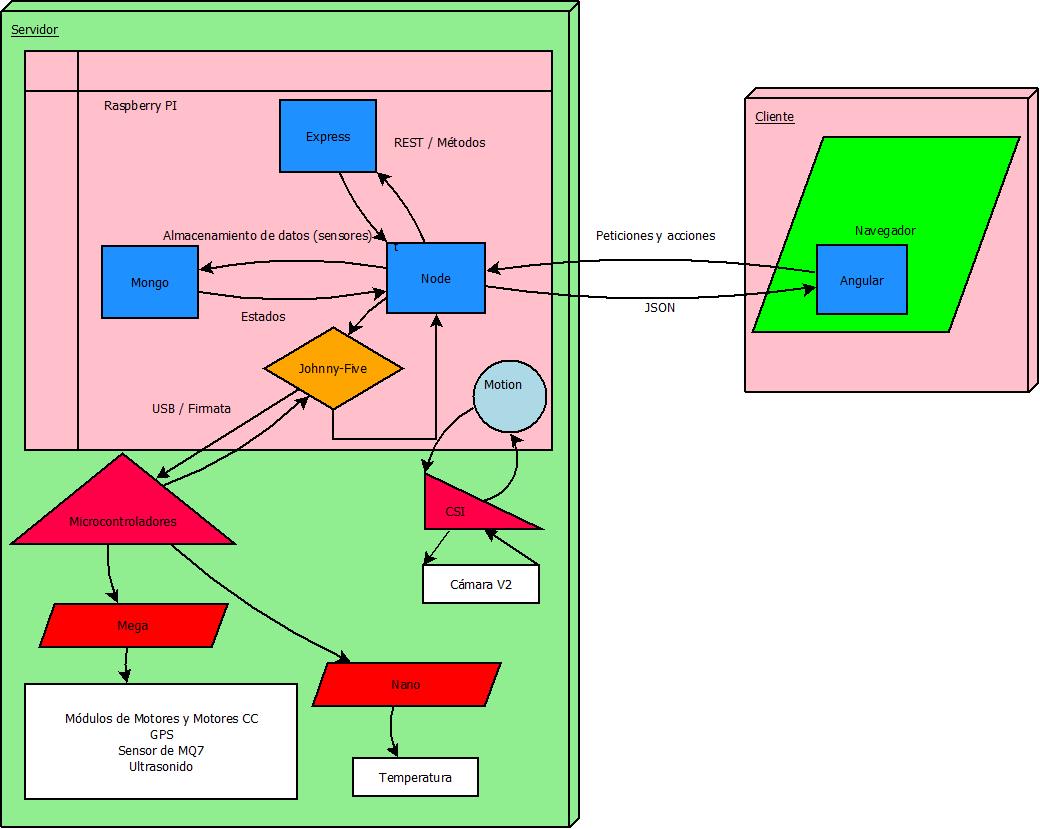


Ilustración - Arquitectura lógica del SAR

## Capturas de pantalla de la aplicación

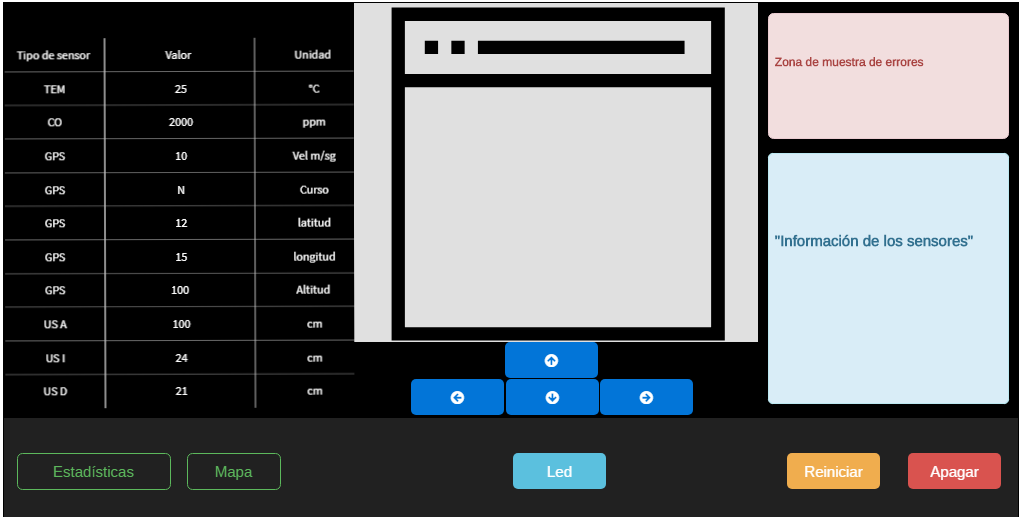


Ilustración - Aplicación Web

## Funcionamiento de la App

FALTA FINALIZARLA

## Como se construyen las estadísticas

## FALTA DETERMINAR LAS MUESTRAS

## Parámetros de configuración

Dentro de esta sección agregar MOTION, PM2, Ad-hoc

## **FALTA FINALIZARLA**

Referencias bibliográficas

1. Agente basado en objetivos: “Almacena información del estado del mundo, así como del conjunto de objetivos que intenta alcanzar, y que es capaz de seleccionar la acción que eventualmente lo guiará hacia la consecución de sus objetivos” [Inteligencia Artificial un enfoque moderno. Person. Stuart Russell, Peter Norving 2da Ed. Pág. 57] [↑](#footnote-ref-1)
2. “Arduino nace como una solución para los diseñadores…”” Donde más se está potenciando es en la educación…” Matías Scovotti, director pedagógico y co-fundador de Educabot. <http://www.telam.com.ar/notas/201704/184406-robotica-arduino-day.html> [↑](#footnote-ref-2)
3. <http://www.educaciontrespuntocero.com/noticias/raspberry-pi-educacion/34377.html> [↑](#footnote-ref-3)
4. Inteligencia Artificial: “Es la inteligencia exhibida por máquinas… una máquina ‘inteligente’ ideal es un agente racional flexible que percibe su entorno y lleva a cabo acciones que maximicen sus posibilidades de éxito en algún objetivo o tarea.” <https://es.wikipedia.org/wiki/Inteligencia_artificial> [↑](#footnote-ref-4)
5. <https://www.upc.edu/latevaupc/usos-y-beneficios-robotica-las-aulas/> [↑](#endnote-ref-1)
6. http://comoprogramarpic.blogspot.com.ar/2012/06/programando-un-atmel-mi-primer-programa.html [↑](#endnote-ref-2)
7. <https://www.arduino.cc/en/Main/Products> [↑](#endnote-ref-3)
8. Internet de las cosas: “Concepto que se refiere a la interconexión de objetos cotidianos con Internet. “ <https://es.wikipedia.org/wiki/Internet_de_las_cosas> [↑](#footnote-ref-5)
9. <https://www.arduino.cc/en/aug/> [↑](#footnote-ref-6)
10. https://www.arduino.cc/en/Reference/PortManipulation [↑](#footnote-ref-7)
11. El uso de algunos de estos módulos queda en forma tentativa, dado que existen también en la Raspberry y su uso puede ser complementario. [↑](#footnote-ref-8)
12. <https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/> [↑](#endnote-ref-4)
13. RISC OS: “Sistema operativo de kernel propio… actualmente mantenido por RISC OS Ltd con una licencia Open Source. Se elaboro para computadoras de escritorio basadas en los chips ARM”. <https://es.wikipedia.org/wiki/RISC_OS> [↑](#footnote-ref-9)
14. Protoboard: O placa de pruebas en castellano, se le llama así a un tablero con orificios que se encuentran conectados eléctricamente entre si siguiendo un determinado patrón. Es utilizado para la conexión de componentes electrónicos. [↑](#footnote-ref-10)
15. En el sitio oficial se encuentra disponible una sección en donde la comunidad puede compartir distintas experiencias y novedades sobre esta plataforma (<https://www.raspberrypi.org/community/>). Por otro lado, cuenta con un área exclusiva donde se puede obtener distinto material didáctico, con proyectos para realizar por ejemplo con alumnos (<https://projects.raspberrypi.org/en/projects>). [↑](#footnote-ref-11)
16. https://source.android.com/security/overview/kernel-security [↑](#endnote-ref-5)
17. http://premios.eset-la.com/universitario/pdf/analisis\_y\_explotacion\_de\_vulnerabilidades\_en\_android.pdf [↑](#endnote-ref-6)