

**Implementación en un** **Robot un Sistema Embebido de Aprendizaje por Refuerzo**

Especificación de requerimientos de software

***Pablo Daniel Folino***

## ([pfolino@gmail.com](mailto:pfolino@gmail.com))

27/07/2020

Versión 1.0

Índice

**Historial de cambios** [2](#HistorialCambios)

**1. Introducción** [3](#Introducción)

1.1. Propósito [3](#Propósito)

1.2. Ámbito del Sistema [3](#_1.2_Ámbito_del)

1.3. Definiciones, Acrónimos y Abreviaturas [3](#Definiciones)

1.4. Referencias [3](#Referencias)

1.5. Visión General del Documento [3](#Visión)

**2. Descripción General** [4](#Descripción)

2.1. Perspectiva del Producto [4](#Perspectiva)

2.2. Funciones del Producto [4](#Funciones)

2.3. Características de los Usuarios [5](#Características) 2.4. Restricciones [5](#Restriccione)

2.5. Suposiciones y Dependencias [5](#Suposiciones)

2.6. Requisitos Futuros [5](#Requisitos)

**3. Requisitos Específicos** [6](#RequisitosEspecíficos)

3.1. Interfaces Externas [6](#InterfacesExternas)

3.2. Funciones [6](#Funciones3_2)

3.3. Requisitos de Rendimiento [7](#RequisitosRendimiento)

3.4. Restricciones de Diseño [7](#RestriccionesDiseño)

3.5. Atributos del Sistema [7](#AtributosSistema)

3.6. Otros Requisitos [7](#_3.6_Otros_Requisitos)

**4. Apéndices** [7](#_4._Apéndices)

**5. Anexos** [8](#Anexos)

Anexo 1.

Caso de uso 1 [8](#Anexos)

Caso de uso 2 [9](#_Caso_de_uso)

Caso de uso 3 [10](#_Caso_de_uso_1)

Caso de uso 4[11](#_Caso_de_uso_2)

**Historial de cambios**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fecha** | **Versión** | **Cambio** | **Autor** | **Revisores** |
| 27/07/2020 | 1.0 | Versión Original | Pablo Folino |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# **Introducción**

## **1.1** **Propósito**

1. Este documento representa una especificación de requerimiento de software para implementar en un Robot un Sistema Embebido de Aprendizaje por Refuerzo.

2. Está dirigido a desarrolladores que se ocupen del análisis, diseño e implementación, así como también a quienes desarrollen el testing de software.

## **1.2** **Ámbito del Sistema**

1. Este software formará parte del trabajo final de la Carrera de Especialización en Sistemas Embebidos.

2. Estará integrado al Robot y brindará al mismo toda la lógica necesaria para controlar sus movimientos, y la comunicación con el mundo exterior.

3. Con este software se pretende evaluar la lógica de control de movimientos de un robot utilizando Aprendizaje por Refuerzo (***RL***) y además compararla con un sistema de control clásico como el ***PID***.

## **Definiciones, Acrónimos y Abreviaturas**

|  |  |  |
| --- | --- | --- |
| CIAA | Computadora Industrial Abierta Argentina | |
| GPS | Sistema de Posicionamiento Global | |
| LCD | Pantalla de cristal líquido | |
| LED | Light Emitting Diode | |
| N/A | No Aplica | |
| PID | Control Proporcional, Integral y Derivativo | |
| PWM | Modulación por ancho de pulso | |
| RL | Aprendizaje por refuerzo (reinforcement learning) | |
| TBD | A ser definido (to be defined) | |
| TBC | A ser confirmado (to be confirmed) | |
| TTL | Transistor-Transistor Logic | |
| WIFI | Sistema de Conexión Inalámbrica | |
| Checksum | Sistema de chequeo de información mediante sumas | |
| WatchDog  (**WDT**) | Técnica para evitar que un procesador quede “colgado” | |
|  | |  | |
|  | |  | |

## **Referencias**

Plan del Proyecto del Trabajo Final de la Carrera de Especialización de Sistemas Embebidos. [<https://github.com/PabloFolino/Plantilla-planificacion>]

## **1.5** **Visión General del Documento**

Este documento se realiza siguiendo el estándar IEEE Std. 830-1998.

# **Descripción General**

## **2.1** **Perspectiva del Producto**

1. El software aquí especificado es independiente de otros sistemas y no tiene relación con otros productos.

2. Se muestra en la siguiente figura en dónde se implementará el software.

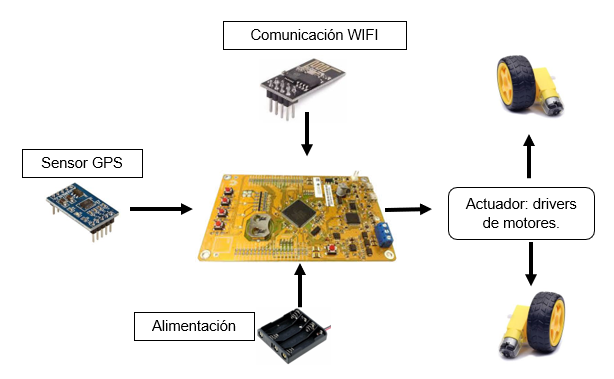


Figura 1. Diagrama conceptual del sistema

3. El software recibirá información de los distintos sensores (***GPS***, Acelerómetros, etc.), y controlará los motores por ***PWM***. Se recibirá comunicación por un módulo ***WIFI***, por el cual se enviarán las distintas acciones que realizará el robot y enviará a un dispositivo remoto información para evaluar los distintos algoritmos de control.

## **2.2** **Funciones del Producto**

1. El software aquí especificado brindará las siguientes funcionalidades:
   * Control de 2 motores por ***PWM*.**
   * Lectura de los módulos acelerómetro, y ***GPS***.
   * Transmisión y recepción inalámbrica de datos por ***WIFI***.
   * Implementación de un control PID clásico.
   * Implementación de controles de***RL***.
2. El software aquí especificado no brindará los servicios de:
   * Medición de consumo de energía eléctrica.
   * Medición de posicionamiento por odometría.
   * Muestra de información en un display tipo ***LCD***.

## **2.3** **Características de los Usuarios**

1. Los usuarios finales de este producto son ingenieros, técnicos, docentes o estudiantes avanzados en alguna carrera relacionada con la programación o electrónica que deseen probar en una plataforma robótica distintas técnicas de control relacionadas con ***RL***.
2. Adicionalmente serán usuarios de este producto personas que no requieren ningún tipo de formación previa.

## **2.4** **Restricciones**

1. El software debe mantenerse bajo control de versiones
2. El software debe ser escrito en lenguaje C

## **2.5** **Suposiciones y Dependencias**

Se asume que se dispondrá de la plataforma de desarrollo, y del hardware del robot durante el tiempo del desarrollo del software para poder hacer distintas comprobaciones.

## **2.6** **Requisitos Futuros**

1. En el futuro se podrán implementar distintos módulos de software que interactúen con distinto tipos de hardware como; módulos de sensor de sonido, módulo digitalizador de imágenes, de comunicación bluetooth, etc.

2. Para evaluar la autonomía se podrá implementar un software que lleve el control de la energía del sistema de baterías.

# **3.** **Requisitos Específicos**

## **3.1** **Interfaces Externas**

* El botón de “Reset” (niveles *TTL*) deberá ser atendido mediante una interrupción, el cual reiniciará todo el sistema. [ROBOT\_RL-ER-0001-REQ0001]​
* El software deberá indicar mediante un LED (niveles ***TTL***) que titile (entre 0,5 a 1,5 segundos) si está funcionando correctamente. [ROBOT\_RL-ER-0001-REQ0002]​
* El software deberá indicar mediante un LED (niveles ***TTL***) el encendido o apagado, y el tiempo de los mismos. [ROBOT\_RL-ER-0001-REQ0003]​
* El software deberá manejar 2 motores con ***PWM*** en forma independiente. [ROBOT\_RL-ER-0001-REQ0004]​
* El software deberá permitir deshabilitar el módulo ***PWM***. [ROBOT\_RL-ER-0001 -REQ0005]​
* La comunicación con ***WIFI***, deberá realizarse en forma de tramas, con un campo de verificación de tramas de 2 bytes mediante Checksum un byte de campo de que me indique el comienzo de la trama, un byte que indique cuántos bytes de datos se envían, la cantidad máxima de datos son de 8 bytes, y una velocidad de comunicación de 115200bps. [ROBOT\_RL-ER-0001-REQ0006]​
* El software deberá poder leer las tres dimensiones del acelerómetro (X, Y y Z), con una precisión ±***TBD*** *m/s2* [ROBOT\_RL-ER-0001-REQ0007]​
* El software deberá poder leer el ***GPS***, con una precisión ± ***TBD*** *metros*. [ROBOT\_RL-ER-0001-REQ0008]​
* El software deberá poder enviar por ***WIFI*** los datos del acelerómetro y del ***GPS*** [ROBOT\_RL-ER-0001-REQ0009]​
* El valor de velocidad en cada motor contemplará no enviar información en la zona muerta del mismo, para evitar recalentamiento y gasto de energía innecesaria. [ROBOT\_RL-ER-0001 -REQ0019]​

## **3.2** **Funciones**

1. Sistema
   * El software deberá esperar un tiempo de 5 segundos una vez iniciado el programa. [ROBOT\_RL-ER-0001 -REQ0010]​
   * El software deberá tener un sistema de WatchDog (de 1 segundo máximo) para asegurar la robustez del sistema. [ROBOT\_RL-ER-0001 -REQ0011]​
2. Control de movimientos
   * El software deberá poder elegir entre controlar la plataforma con un ***PID*** clásico, y un control mediante ***RL.*** [ROBOT\_RL-ER-0001 -REQ0012]​
   * Cuando se controla al movimiento de la plataforma en ***RL***, el software deberá poder seleccionar si está en formato de exploración o explotación. [ROBOT\_RL-ER-0001 -REQ0013]​
   * Cuando se controla al movimiento de la plataforma en ***RL***, el software deberá poder leer y escribir la matriz Estado/Acción. [ROBOT\_RL-ER-0001 -REQ0014]​
   * Cuando se controla al movimiento de la plataforma en ***RL***, el software deberá enviar mediante tramas la matriz Estado/Acción. [ROBOT\_RL-ER-0001 -REQ0015]​
   * El software podrá recibir las constantes del controlador ***PID****,* para poder sintonizar el sistema.[ROBOT\_RL-ER-0001 -REQ0016]
   * El software podrá recibir tramas de control de movimiento, hacia adelante, hacia atrás, giro hacia la derecha e izquierda y parada.[ROBOT\_RL-ER-0001 -REQ0017]

## 

## **3.3** **Requisitos de Rendimiento**

* + El ciclo de control de los motores no será mayor a 100mseg. [ROBOT\_RL-ER-0001 -REQ0018]​

## **3.4** **Restricciones de Diseño**

* Se utilizará el ESP8266 como dispositivo para la comunicación ***WIFI*** en modo servidor. [ROBOT\_RL-ER-0001 -REQ0019]​
* Se utilizará la CIAA-NXP como computadora principal. [ROBOT\_RL-ER-0001 -REQ0020]​
* Se utilizará un puente H para controlar los motores de corriente continúa con un integrado L298. [ROBOT\_RL-ER-0001 -REQ0021]​

]​

## **3.5** **Atributos del Sistema**

1. Mantenibilidad
   * El software deberá permitir su actualización y restablecimiento remoto. [ROBOT\_RL-ER-0001 -REQ0022]​
   * El software deberá permitir modificar sus parámetros de funcionamiento y almacenarlos en la memoria EEPROM. [ROBOT\_RL-ER-0001 -REQ0023]​
2. Seguridad
   * El software deberá permitir conectar en forma remota a un usuario por vez. [ROBOT\_RL-ER-0001 -REQ0024]
   * El software deberá solicitar un password, para poder solicitar la modificación los parámetros de funcionamiento. [ROBOT\_RL-ER-0001 -REQ0025]​

## **3.6 Otros Requisitos**

N/A

# **4.** **Apéndices**

N/A

# **5. Anexos**

## **Anexo 1.** **Casos de uso**

Caso de uso 1 [ROBOT\_RL-ER-0001-CU0001]​

|  |  |  |
| --- | --- | --- |
| **Título** | | **Descripción** |
| 1. Nombre | | Iniciar el funcionamiento del robot. |
|  | 1.1 Breve  Descripción | Este caso describe inicio del funcionamiento del robot suponiendo las condiciones estándar de funcionamiento. |
| 1.2 Actor Principal | Usuario final del producto. |
| 1.3 Disparadores | Pulsación del botón de encendido del robot. |
| 2. Flujo de eventos | |  |
|  | 2.1 Flujo básico | 1. El usuario presiona el botón de encendido. 2. El sistema espera 5 segundos. 3. El software carga las constantes del sistema, lanza el WatchDog, y el controlador de motores por defecto. 4. El software se conecta al módulo ***WIFI***, y envía un mensaje de conexión. 5. El software lee el ***GPS***, y los datos de nivel del acelerómetro. 6. El software en cada ciclo de control hace titilar un LED. 7. El software queda esperando un comando (una orden a ejecutar). |
| 2.2 Flujo alternativo | 4.A. El software no puede conectarse al módulo ***WIFI***, un lapso de 1 minuto.  4.A.1 El software deja de titilar el LED.  4.A.2 Se entra en modo de falla. |
| 3. Requerimientos especiales | | N/A |
| 4. Pre-Condiciones | | 1. Se supone que el pack de baterías está puesto y posee energía.  2. Se supone el Robot apagado.  3. Se supone que el Robot está puesto en el suelo, en una superficie plana. |
| 5. Post-Condiciones | | 1. Queda titilando el LED en condiciones normales. |

### Caso de uso 2 [ROBOT\_RL-ER-0001-CU0002]​

|  |  |  |
| --- | --- | --- |
| **Título** | | **Descripción** |
| 1. Nombre | | Configurar las constantes del ***PID*** clásico. |
|  | 1.1 Breve  Descripción | Este caso se muestra la secuencia de cómo se configuran las constantes del ***PID***. |
| 1.2 Actor Principal | Técnico especializado(usuario). |
| 1.3 Disparadores | Se envía una trama de configuración del ***PID*** clásico. |
| 2. Flujo de eventos | |  |
|  | 2.1 Flujo básico | 1. El usuario envía por el ***WIFI****,* una trama de configuración. 2. El software solicita al usuario por el canal de ***WIFI*** *un password.* 3. El usuario manda el password vía ***WIFI***. 4. El software acepta el password del usuario. 5. El software duplica la velocidad en que titila el LED para indicar que está en modo programación. 6. El usuario envía las tres constantes ***PID*** de configuración en una trama. 7. El software envía al usuario la aceptación de la configuración. 8. El software vuelve a la velocidad normal de encendido/apagado del LED. 9. El software sale del modo programación, y queda a la espera de una orden. |
| 2.2 Flujo alternativo | 4.A. El software no acepta el password del usuario en tres reintentos.  4.A.1 El software mediante una trama informa al usuario que sale del modo de programación.  4.A.2 Vuelve al paso 8 del Fujo Principal. |
| 3. Requerimientos especiales | | N/A |
| 4. Pre-Condiciones | | 1. Se supone que el Robot está encendido y preparado para recibir un comando. |
| 5. Post-Condiciones | | 1. Queda titilando el LED en condición normal, con la nueva programación de las constantes ***PID***. |

### Caso de uso 3 [ROBOT\_RL-ER-0001-CU0003]​

|  |  |  |
| --- | --- | --- |
| **Título** | | **Descripción** |
| 1. Nombre | | Iniciar un movimiento. |
|  | 1.1 Breve  Descripción | Se describe un envío de una trama de un movimiento. |
| 1.2 Actor Principal | Usuario final del producto. |
| 1.3 Disparadores | Se envía una trama de movimiento por el canal ***WIFI***. |
| 2. Flujo de eventos | |  |
|  | 2.1 Flujo básico | 1. El usuario envía por el ***WIFI****,* una trama de movimiento. 2. El software acepta la trama de movimiento. 3. El software deja encendido el LED durante 2 segundos, y luego retorna a la velocidad normal. 4. El software envía al usuario vía ***WIFI*** la aceptación de la trama. 5. El software queda a la espera de una orden. |
| 2.2 Flujo alternativo | 2.A. Si el software no acepta la trama.  2.A.1 El software apaga el LED por 2 segundos.  2.A.2 Vuelve al ítem 5 del Flujo Principal. |
| 3. Requerimientos especiales | | N/A |
| 4. Pre-Condiciones | | 1. Se supone que el Robot está encendido y preparado para recibir un comando. |
| 5. Post-Condiciones | | 1. Queda titilando el LED en conficiones normales. |

### Caso de uso 4 [ROBOT\_RL-ER-0001-CU0004]​

|  |  |  |
| --- | --- | --- |
| **Título** | | **Descripción** |
| 1. Nombre | | Carga de la matriz Estado/Acción. |
|  | 1.1 Breve  Descripción | Este caso describe el escenario en donde se envía la tabla Estado/Acción del controlador de ***RL***. |
| 1.2 Actor Principal | Usuario final del producto. |
| 1.3 Disparadores | Se envía una trama de carga de la tabla Estado/Acción desde el canal de **WIFI**. |
| 2. Flujo de eventos | |  |
|  | 2.1 Flujo básico | 1. El usuario envía por el ***WIFI****,* una trama de configuración. 2. El software solicita al usuario por el canal de ***WIFI*** *un password.* 3. El usuario manda el password vía ***WIFI***. 4. El software acepta el password del usuario. 5. El software duplica la velocidad en que titila el LED para indicar que está en modo programación. 6. El usuario envía tramas de datos de la matriz Estado//Acción. 7. El software verifica la integridad de la matriz recibida. 8. El software graba la matriz en la memoria EEPROM. 9. El software vuelve a la velocidad normal de encendido/apagado del LED. 10. El software sale del modo programación, y queda a la espera de una orden. |
| 2.2 Flujo alternativo | 7.A. El software no verifica que la integridad de la matriz no es la correcta.  7.A.1 El software apaga el LED por 2 segundos.  7.A.2 Vuelve al ítem 9 del Flujo Principal. |
| 3. Requerimientos especiales | | N/A |
| 4. Pre-Condiciones | | 1. Se supone que el pack de baterías está puesto y posee energía.  2. Se supone el Robot apagado.  2. Se supone que el Robot está puesto en el suelo, en una superficie plana. |
| 5. Post-Condiciones | | 1. Queda titilando el LED. |