

Diplomado en Software Embebido

Reporte Final Proyecto Integrador

**“Control de Velocidad de un Motor de Corriente Directa por PWM”**

Equipo C

**González Mateo Fátima Daniela**

**Arriaga Martínez Ladislao**

MABE

Asesor:

**Marcos Samuel Peña Flores**

MABE

Querétaro, Qro. **09 de Agosto** de 2019.

Tabla de contenido

[Revisores y aprobaciones 4](#_Toc16261084)

[Historial de cambios 4](#_Toc16261085)

[Definición del proyecto 5](#_Toc16261086)

[Alcance del proyecto 5](#_Toc16261087)

[Identificación del proyecto 5](#_Toc16261088)

[Descripción general del proyecto 5](#_Toc16261089)

[Entregables 5](#_Toc16261090)

[Entregables con el cliente 5](#_Toc16261091)

[Compromisos de entrega de los módulos del software 6](#_Toc16261092)

[Arquitectura 6](#_Toc16261093)

[Módulos Propuestos 6](#_Toc16261094)

[Metodologías de Desarrollo 6](#_Toc16261095)

[Metodología de desarrollo de software seleccionada 6](#_Toc16261096)

[Plan de entregas de software al cliente 6](#_Toc16261097)

[Procedimientos de control 7](#_Toc16261098)

[Estimación 8](#_Toc16261099)

[Recursos críticos (RAM, ROM, Throughput) 8](#_Toc16261100)

[**Recursos Criticos figura 1:** % de uso RAM y ROM al inicio. 9](#_Toc16261101)

[Infraestructura y herramientas 9](#_Toc16261102)

[Recursos humanos 10](#_Toc16261103)

[Planeación 10](#_Toc16261104)

[Suposiciones 10](#_Toc16261105)

[Restricciones 11](#_Toc16261106)

[Riesgos 11](#_Toc16261107)

[Resolución de problemas 12](#_Toc16261108)

[Roles y responsabilidades 12](#_Toc16261109)

[Trazabilidad de entregables 12](#_Toc16261110)

[Administración de riesgos 13](#_Toc16261111)

[Análisis de requisitos de software 14](#_Toc16261112)

[Diseño de arquitectura de software 15](#_Toc16261113)

[Diagrama de bloques de SW 15](#_Toc16261114)

[Diagrama de control 15](#_Toc16261115)

[Entradas y Salidas 16](#_Toc16261116)

[Diagramas 16](#_Toc16261117)

[Diagramas de tiempos / recursos 16](#_Toc16261118)

[Diagrama de flujo 16](#_Toc16261119)

[Construcción del software 17](#_Toc16261120)

[Aplicación de estándares de codificación y convenciones 17](#_Toc16261121)

[Nomenclatura 17](#_Toc16261122)

[Nomenclatura de funciones 18](#_Toc16261123)

[Nomenclatura de clases 18](#_Toc16261124)

[Nomenclatura de variables 19](#_Toc16261125)

[Tipo de datos 19](#_Toc16261126)

[Estructura del proyecto 20](#_Toc16261127)

[Metodología de revisión de software 21](#_Toc16261128)

[Pruebas de software 22](#_Toc16261129)

[Pruebas de integración 22](#_Toc16261130)

[White Box Test 23](#_Toc16261131)

[Final: % Uso de ROM, % Uso de RAM , STACK, Mapa de Memoria 23](#_Toc16261132)

[**Final, RAM y ROM Figura 1:** % de uso de RAM y ROM. 24](#_Toc16261133)

[**Figura 2:** Final Stack y Mapa de memoria. 24](#_Toc16261134)

[Throughput 24](#_Toc16261135)

[Complejidad de Codigo 24](#_Toc16261136)

[**Analisis de Complejidad, Figura 1:** control\_thread\_entry.c 24](#_Toc16261137)

[**Analisis de Complejidad, Figura 2:** sensor\_thread\_entry.c 25](#_Toc16261138)

[**Analisis de Complejidad, Figura 3:** main\_thread\_entry.c 25](#_Toc16261139)

[Gray Box Test 25](#_Toc16261140)

[Black Box Test 26](#_Toc16261141)

[**Black box figura 1:** Resumen de pruebas. 27](#_Toc16261142)

[**Black box figura 2:** Prueba # 1. 27](#_Toc16261143)

[**Black box figura 3:** Prueba # 2. 27](#_Toc16261144)

[**Black box figura 5:** Prueba # 4. 28](#_Toc16261145)

[ABA Test 28](#_Toc16261146)

[Verificación y aseguramiento de la calidad 28](#_Toc16261147)

[Liberación de versiones de software 28](#_Toc16261148)

[Prueba funcional y validación en campo 29](#_Toc16261149)

[Calendario de revisión: documental y código 30](#_Toc16261150)

# Revisores y aprobaciones

Revisión: 1.0

Fecha de última revisión: 9 de agosto de 2019

Realizado por:

González Mateo Fátima Daniela

Arriaga Martínez Ladislao

Revisores:

Marcos Samuel Peña Flores

Aprobadores:

|  |  |  |  |
| --- | --- | --- | --- |
| Marcos Samuel Peña Flores  **Product Owner - Asesor de proyecto**  **Nombre / Puesto** | 09/08/2019  Fecha | **Nombre / Puesto** | **Fecha** |
| Ladislao Arriaga Martinez  **Scrum master - Ing. de desarrollo de SW**  **Nombre / Puesto** | 09/08/2019  Fecha | **Nombre / Puesto** | **Fecha** |
| Fátima Daniela Gonzalez Mateo  **Ing. de desarrollo de SW**  **Nombre / Puesto** | 09/08/2019  Fecha | **Nombre / Puesto** | **Fecha** |

# Historial de cambios

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Revisiones** | | | | | | | | |
| 0 | 09/08/2019 | F. Gonzalez |  |  |  |  |  |  |
| L. Arriaga |  |  |
| 1 | 29/07/2019 | F. Gonzalez |  |  |  |  |  |  |
| L. Arriaga |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |

URL del repositorio:

<https://github.com/fdgm/diplomadosoftwareembebido#workspaces/proyectointegrador-5c9677efb350086a7c331fea/board?repos=177324101>

<https://github.com/laoarriaga77/DSE_Documentacion>

# Definición del proyecto

## Alcance del proyecto

La unión de varias empresas de distintos giros en sinergia con la UTEQ identificó la necesidad de preparar personal sin experiencia en Software Embebido a nivel profesional, con el objetivo de que ambas partes fueran beneficiadas; derivado de lo anterior, nació el Diplomado en Software Embebido.

Como parte del diplomado, se debe desarrollar un proyecto integrador donde se aplicará la metodología de desarrollo vista durante el diplomado. Análisis de Requisitos, Definición de Módulos de Software, implementación de código embebido, verificación y validación del producto con su debida documentación.

Cabe resaltar que la documentación es un factor fundamental a la hora de desarrollar un proyecto, ya que una buena documentación evita re trabajo y garantiza el modularidad en cada proyecto.

## Identificación del proyecto

El proyecto integrador consiste en controlar la velocidad de un motor de corriente directa alimentado con 12 volts mediante la aplicación de una señal cuadrada la cual varía su ancho de pulso (PWM), y cuya frecuencia de trabajo está definida en 1kHz.

## Descripción general del proyecto

Se utilizará un sensor de efecto Hall acoplado al rotor del motor para poder medir la velocidad del mismo. Dicho sensor provee 4 pulsos por revolución, es decir, si se requiere una mayor velocidad el número de pulsos leídos aumentará, y por el contrario, si la velocidad es menor, la lectura de pulsos disminuye. El set point de referencia de velocidad (RPM) estará determinado por un potenciómetro.

Adicional, se tendrá una interfaz gráfica en la cual se deberá desplegar la velocidad actual de motor, el valor de set point, y el porcentaje de memoria utilizado.

# Entregables

## Entregables con el cliente

Como parte del proyecto integrador del diplomado se entregará lo siguiente:

1. Reporte final con la descripción del proyecto.
2. Software implementado en el proyecto.

## Compromisos de entrega de los módulos del software

### Arquitectura

* Propuesta de los pines in / out en la tarjeta: Entradas y Salidas.
* Inicialización de temporizadores y ADC
* Requerimientos funcionales: Duty cycle PWM, Set Point y RPMS.

### Módulos Propuestos

* Módulo ADC: Resolución 8 BITS.
* Módulo de comunicación con LCD.
* Módulo PWM: Duty cycle.
* Etapa de potencia.
* Módulo de I/O.
* Modulo Sensor Hall

# Metodologías de Desarrollo

## Metodología de desarrollo de software seleccionada

Para gestionar el desarrollo del software del proyecto integrador se seleccionó la metodología SCRUM, la cual se caracteriza por el trabajo colaborativo, en equipo y con mejores resultados en el desarrollo de un proyecto. Como ventajas se tiene flexibilidad a los cambios de requerimientos, reducción de riesgos pues se van realizando entregas parciales y regulares del software final.

Como parte de la metodología SCRUM, se definió al equipo quedando de la siguiente manera: Producto Owner - Ing. de software Marcos Samuel Peña Flores, Scrum Master - Ing. de desarrollo Ladislao Arriaga Martinez, Development Team - Ing. de desarrollo Ladislao Arriaga Martinez y Fatima Daniela Gonzalez Mateo. De igual forma se eligió GitHub como plataforma de desarrollo colaborativo de software.

Se realizó el Product Backlog, que fue revisada en conjunto al Product Owner, en la cual se enlistaron todos los requerimientos necesarios para llevar a cabo el proyecto; para cada requerimiento del Product Backlog, se definieron las tareas (tickets) necesarios para cumplir con los requerimientos, cada tarea con un entregable que será incluido en la entrega final. Se realizó la planificación de los sprints, el cual va asociado a las tareas definidas.

## Plan de entregas de software al cliente

Se pretende hacer una sola entrega de software al cliente en las instalaciones de la UTEQ (edificio Intel) y en la fecha definida en la portada de este reporte.

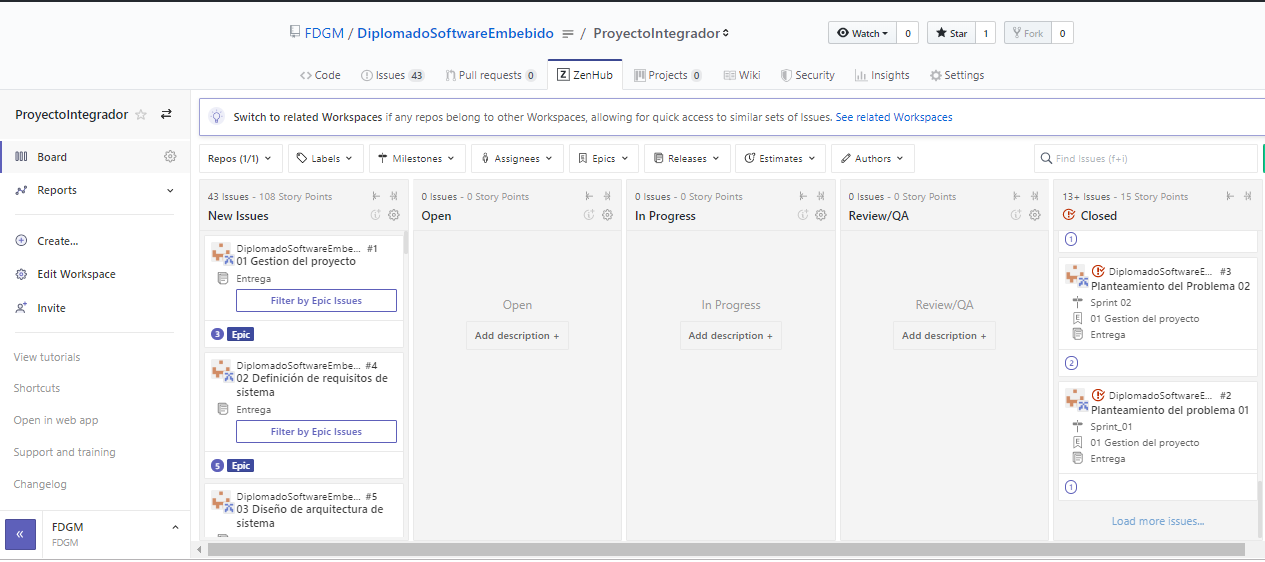
Previamente a esta entrega se deberán hacer una serie de revisiones con el asesor para corroborar que todo esté completo en cuanto a que los requisitos del cliente se están cumpliendo, han sido validados y sin bugs por corregir

## Procedimientos de control

Para llevar un buen control del proyecto, este se ha ido documentando con la ayuda de ZenHub de la plataforma GIT HUB®. En esta plataforma se llevó la administración del avance, así como el control de versiones de software del proyecto. Dentro de GIT HUB®, se creó un repositorio donde se vació toda la planeación realizada con ayuda de la metodología SCRUM, en el repositorio se crearon sprints y dentro de los sprint se crearon los tickets de cada tarea y ahí mismo se asignó su responsable para darle seguimiento y cerrar las tareas.

Se ha utilizado también Excel. Estaherramienta de hojas de cálculo utilizada para el desarrollo de diagramas de Gantt para la administración de tareas.

En la imagen siguiente, se puede ver la parte de ZenHub dentro de la plataforma GitHub, donde se cargaron las Épicas y también donde se van cerrando.



## Estimación

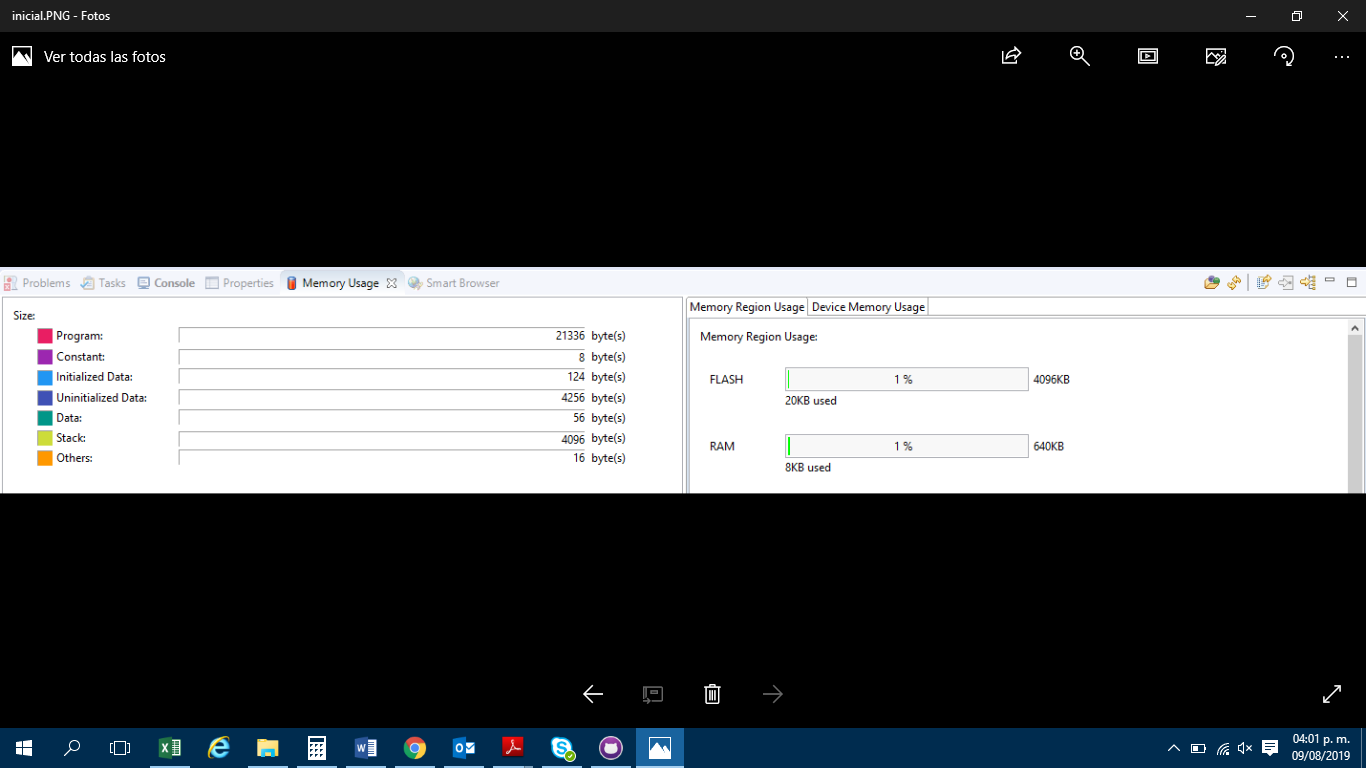
Como parte del seguimiento de las actividades del proyecto, así como sus fechas críticas, se realiza un Gantt de tiempos.





## Recursos críticos (RAM, ROM, Throughput)

Al inicio del proyecto se tienen los recursos de RAM y ROM de la siguiente manera:

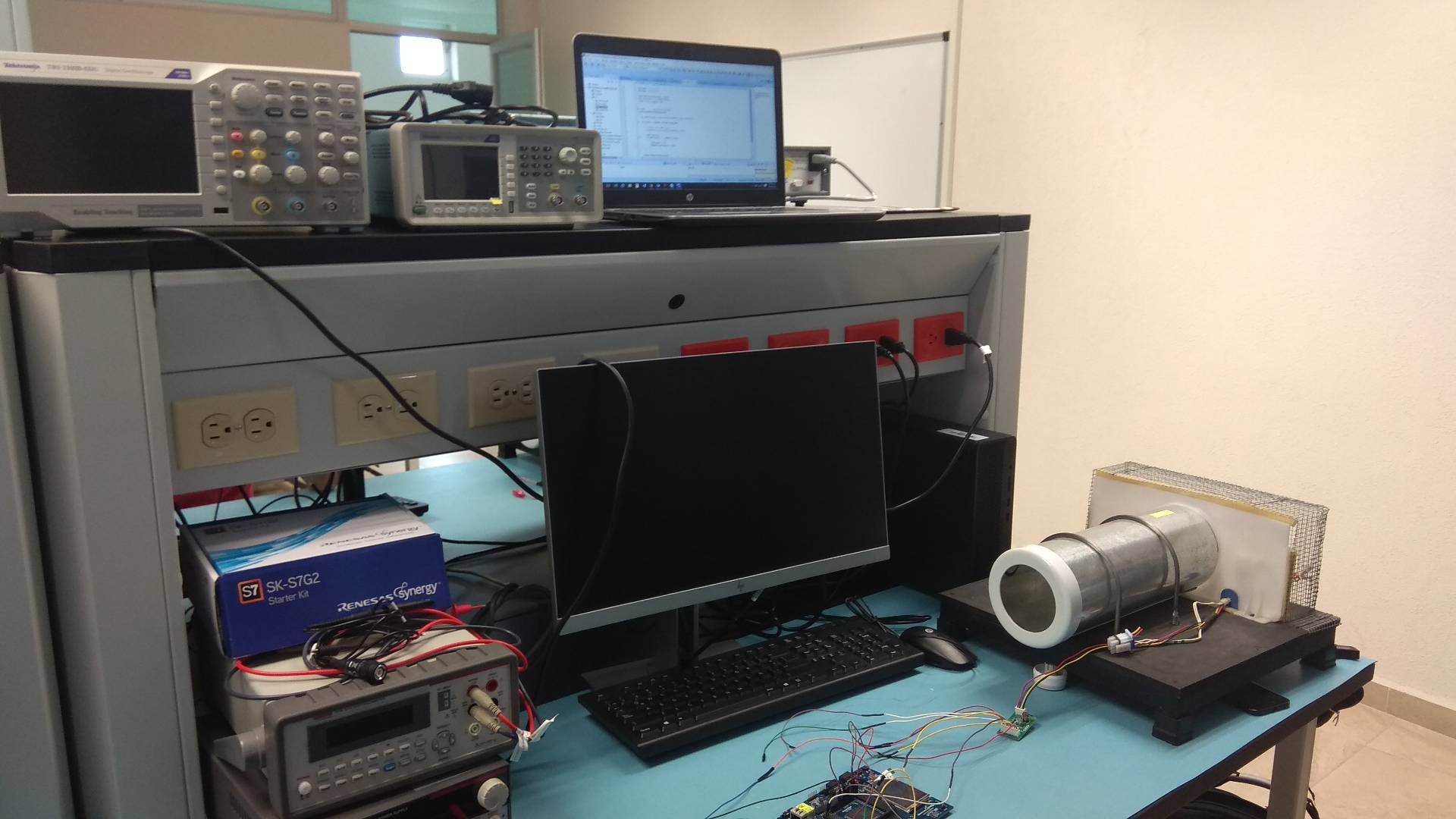


## **Recursos Criticos figura 1:** % de uso RAM y ROM al inicio.

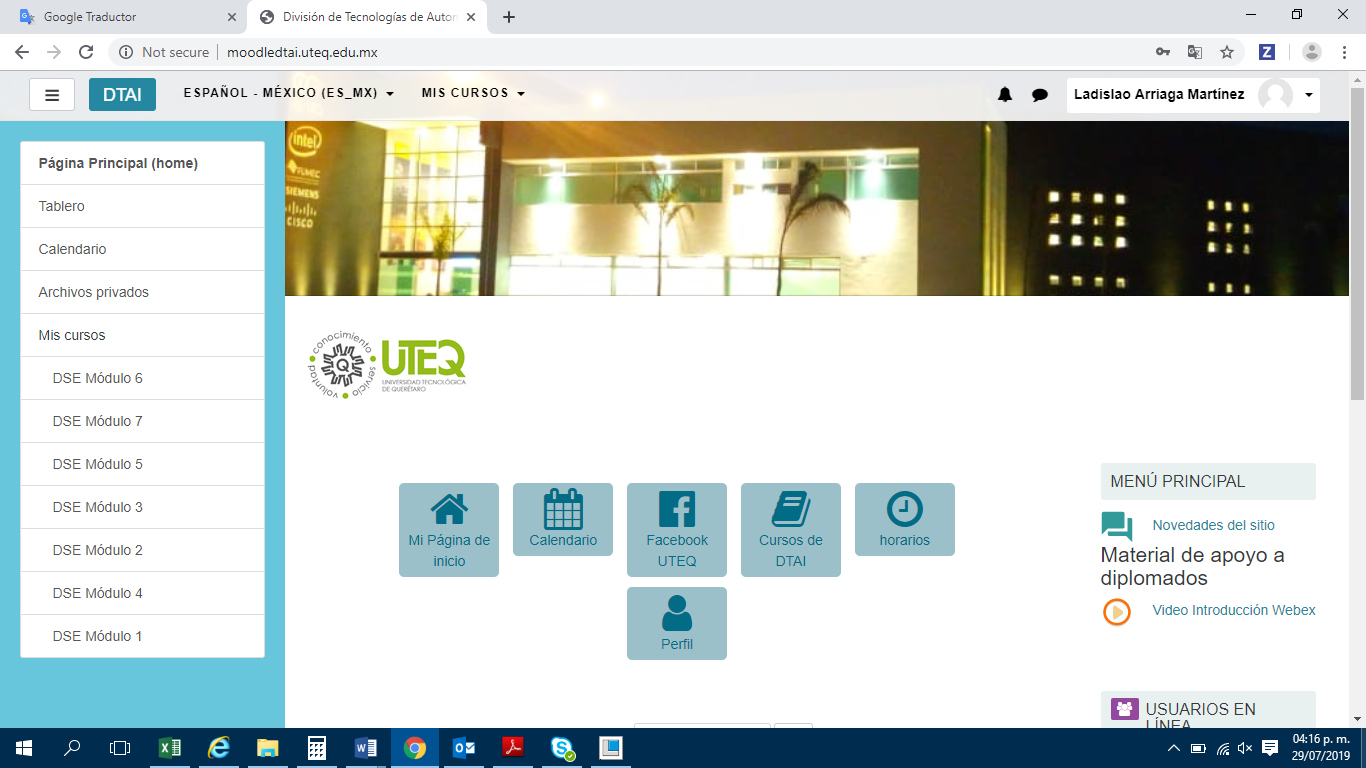
## Infraestructura y herramientas

Como parte del diplomado y para llevar a cabo el proyecto integrador se brindó un laboratorio con herramientas y materiales necesarios para el desarrollo, los cuales se enlistan a continuación:

1. Equipo de cómputo.
2. Tarjeta de programación y desarrollo Renesas S7-G2.
3. Tarjeta Driver con puente H.
4. Planta con el motor ventilador y sensor de efecto hall.
5. Arnés de comunicación entre tarjeta driver y tarjeta de programación.
6. Material para realizar soldadura.
7. Cable tomacorriente, cables banana, caimán y Dupont.
8. Generador de funciones.
9. Osciloscopio.
10. Fuente de alimentación.
11. Multímetro.



También la Universidad Tecnológica de Querétaro (UTEQ) brindó su plataforma en línea, en la cual se encuentra la información proporcionada en las clases y donde se guardaron las tareas realizadas y se aplicaron los exámenes.



## 

## Recursos humanos

Las empresas involucradas en el desarrollo del diplomado en sinergia con la UTEQ brindaron capital humano para impartir las clases del temario presentado al inicio de este, los cuales cuentan con amplia experiencia en el ramo y adicionalmente fueron encargados de dar soporte a cada equipo durante el desarrollo del proyecto. Cada equipo contó con un asesor.

El equipo C cuenta con 2 recursos humanos que dividen las tareas en los siguientes puntos:

* Software.
* Hardware.
* Evaluación.
* Documentación.

# Planeación

## Suposiciones

* Implementación del control PI al 100%.
* Interfaz de desarrollo (e2studio) amigable para la implementación.

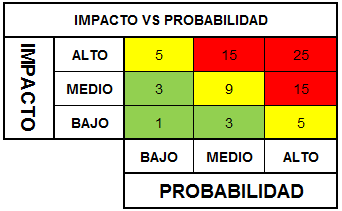
## Restricciones

* Limitación en horas prácticas para el desarrollo del proyecto.
* No contar con la posibilidad de usar la tarjeta de desarrollo y planta fuera del horario del diplomado.
* Carga de trabajo diario no permite dedicarle tiempo necesario al desarrollo del proyecto.

## Riesgos

Se realiza un análisis de riesgos con el equipo involucrado para identificar los posibles riesgos que pudieran impactar la implementación del proyecto.





## Resolución de problemas

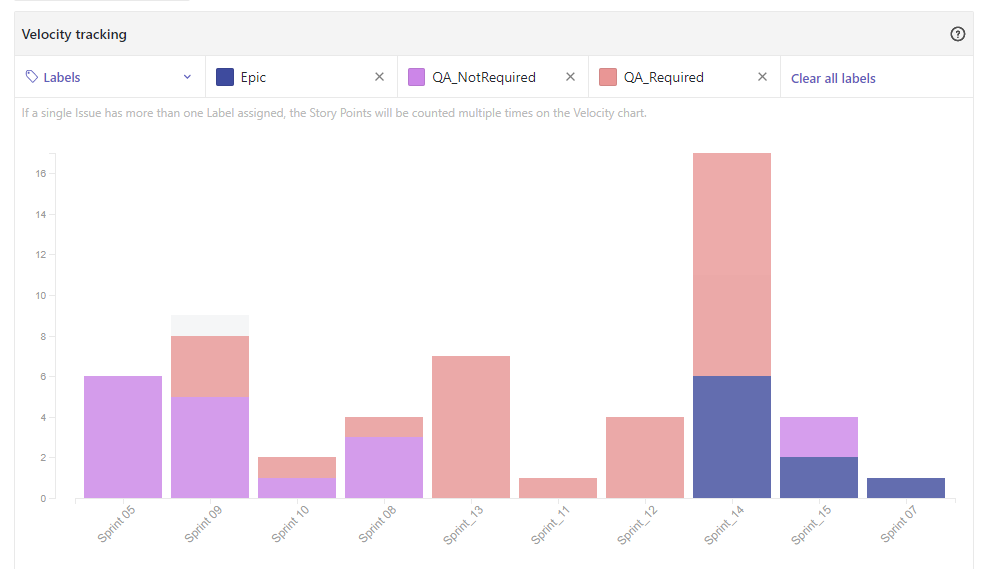
* Acudir con asesor y profesores para la solución de problemas durante el desarrollo.
* Leer notas de aplicación de Renesas.

## Roles y responsabilidades

|  |  |  |
| --- | --- | --- |
| Participante | Rol | Responsabilidad |
| Marcos Samuel Peña Flores | Product Owner - Asesor de proyecto | * Dueño del proyecto * Aprobar la entrega final del proyecto * Asesor de actividades desarrolladas durante el proyecto. |
| Ladislao Arriaga Martinez. | Scrum Master - Development Team | * Asignar tareas definidas. * Participar en la realización de tareas y entregables. * Participar en la mejora del proyecto. * Actualizar estado de las tareas en GitHub. * Documentar proyecto en el reporte final. |
| Fatima Daniela Gonzalez Mateo. | Development Team | * Participar en la realización de tareas y entregables. * Participar en la mejora del proyecto. * Actualizar estado de las tareas en GitHub. * Documentar proyecto en el reporte final. |

## Trazabilidad de entregables

Como parte del desarrollo se utilizó GitHub para la administración del proyecto, en el cual se generaron tickets asignados a los sprints correspondientes, quedando documentado los entregables en las fechas asignadas, a continuación se muestra el grafico velocity tracking:



## Administración de riesgos

Como acciones para reducir o eliminar el impacto de los riesgos encontrados durante el análisis se incluye los planes de acción o abatimiento para la solución de los mismos y los responsables de su implementación.



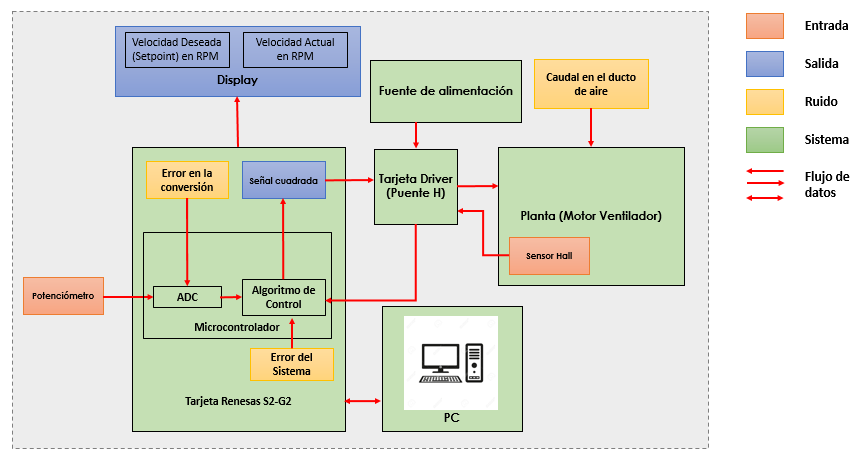
## Análisis de requisitos de software

El análisis de requisitos del software se realizó de acuerdo a los requerimientos del cliente, los cuales, a su vez, fueron traducidos a requisitos de sistema, indicando su debida canalización a las diferentes disciplinas que componen un sistema embebido.

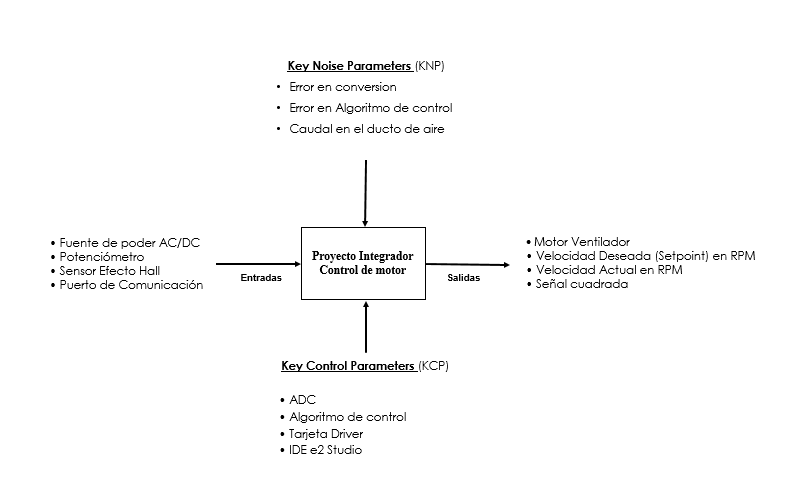
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Requerimiento** | | | | | | | | | | | | | | |
| **Item** | **Descripción** | **Necesario** | **Completo** | **Conciso** | **Consistente** | **Correcto** | **Verificable** | **No ambiguo** | **Diseño Libre** | | **Rastreable** | **Mantenimiento** | **Atómico** | **Factible** |
| Req 1 | Control mediante una señal cuadrada de PWM |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 2 | Frecuencia de trabajo de la señal cuadrada f=1KHz |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 3 | Medición de velocidad a través de sensor de efecto hall |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 4 | Voltaje de alimentación del sistema debe ser de 12 VCD |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 5 | Valor de "Set Point" de la velocidad será dada a través de un potenciómetro |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 6 | Mostrar en LCD velocidad del motor en RPM |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 7 | Mostrar en LCD set point en RPM |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 8 | Mostrar en LCD % de trabajo de la señal cuadrada |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 9 | Revisión de la Complejidad a través de la CCCC, la cual deberá ser <19. |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 10 | Utilizar herramienta GitHub para administración de proyecto. |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 11 | El microcontrolador estará acoplado al puente H DRV8848-2A |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 12 | El sistema deberá contar con un diodo de giro libre o reversa tipo 1N4001 |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 13 | El sistema deberá contar con un control PI |  |  |  |  |  |  |  |  |  | |  |  |  |
| Req 14 | Los textos en la pantalla LCD deben utilizar la Tipografía indicada en el documento del proyecto integrador. |  |  |  |  |  |  |  |  |  | |  |  |  |

# Diseño de arquitectura de software

## Diagrama de bloques de SW



## Diagrama de control



### Entradas y Salidas

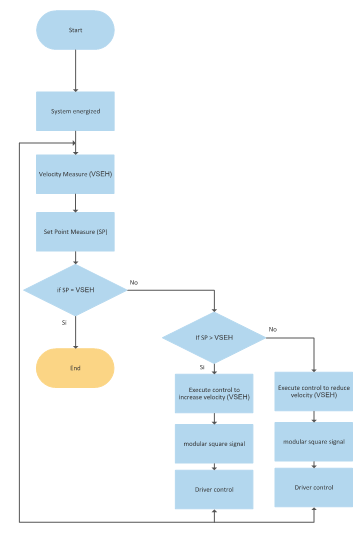
|  |  |  |  |
| --- | --- | --- | --- |
| **No. de parte esquemático** | **Descripción** | **Puerto de conexión** | **Función** |
| J26 | Señal de entrada del potenciómetro (ADC) | AN0 | Señal de Set Point |
| J21 | Señal de salida PWM | Port 4 Pin 6 | Control de velocidad de motor (PWM) |
| J22 | Señal de entrada de sensor Hall | Port 1 Pin 2 | Lectura de pulsos de sensor Hall |
| J25 | Voltaje de alimentación potenciómetro | 3V3 | 3.3V |
| J25 | Señal de tierra de potenciómetro | GND | Tierra |
| J20 | MOSI-SPI-LCD |  | SPI LCD Conectivity |
| J20 | MISO- SPILCD |  | SPI LCD Conectivity |
| J20 | SCK-SPI-LCD |  | SPI LCD Conectivity |
| J24 | SDA IIC2 |  | Touch Conectivity |
| J24 | SCL IIC2 |  | Touch Conectivity |
| NA | Reset para Touch Panel |  | Touch Conectivity |
| NA | LCD Reset |  | Touch Conectivity |
| NA | LCD CS |  | Touch Conectivity |

# Diagramas

## Diagramas de tiempos / recursos

### Diagrama de flujo

A continuación, se muestra el diagrama de flujo del sistema:



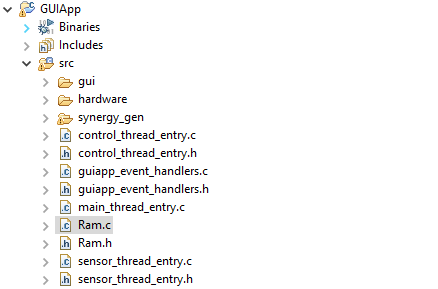
# Construcción del software

## Aplicación de estándares de codificación y convenciones

### Nomenclatura

A continuación se enlistan los archivos necesarios para la ejecución del programa.

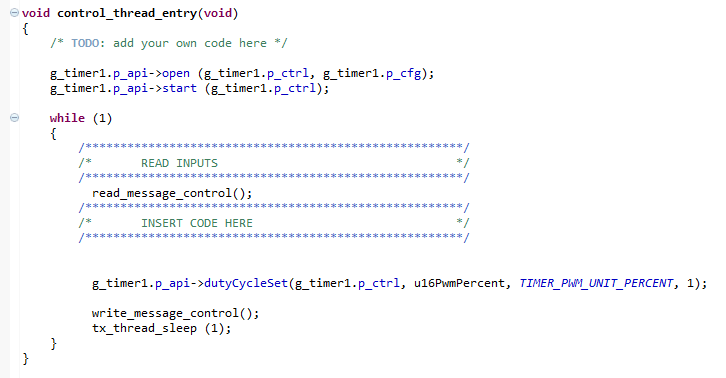
Los archivos que interactúan con el LCD, lo realizan mediante mensajes de eventos con headers que se crean con terminación \_api.h. Estos contienen variables sin signo de 16 bits.

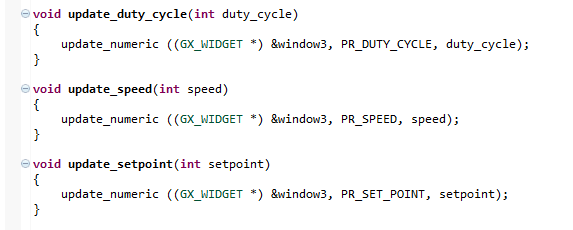


### Nomenclatura de funciones

El nombre de la función deberá ser lo más asertivo a la acción que va a desempeñar dicha función, deberá comenzar la primera letra con mayúscula de la palabra; y si se compone de más palabras deberán comenzar con mayúsculas.

Ejemplo:





### Nomenclatura de clases

Se crearon las siguientes clases para la lectura y actualización de variables en el LCD:

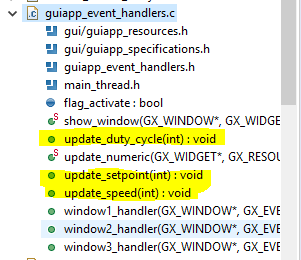
SF\_MESSAGE\_EVENT\_CLASS\_ADC, /\* ADC \*/

SF\_MESSAGE\_EVENT\_CLASS\_PWM, /\* PWM \*/

SF\_MESSAGE\_EVENT\_ADC, /\* ADC \*/

SF\_MESSAGE\_EVENT\_PWM, /\* PWM \*/

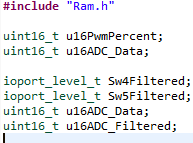
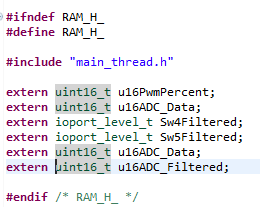
La actualización del LCD se realizó con event class los cuales son llamados al main\_thread\_entry por medio de las clases ADC y PWM y los cuales tienen asignados un GXEVENT\_MSG creados a partir de la ventana de eventos, los cuales interactúan con GUIX para la ejecución de acciones en la LCD.



### Nomenclatura de variables

Globales

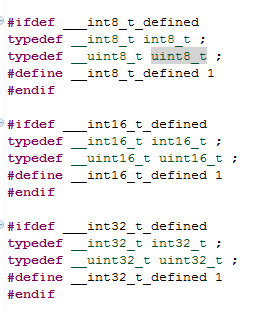
Las variables globales deberán iniciar con el tipo de variable (unsigned #bits o insigned #bits) Seguido del nombre más asertivo a la acción a desempeñar. Deberán ser declaradas en la Ram.h y Ram.c

### Tipo de datos

#### Estándares

Para los datos se aplica el ISO C standart, ver System Interfaces volume of IEEE Std 1003.1-2001, Section 2.2, The Compilation Environment.

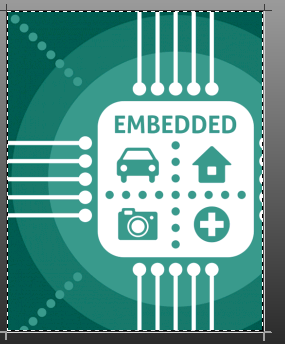


### Estructura del proyecto

Para la visualización de datos en la pantalla se eligió la siguiente distribución:

Pantalla 1 Fondo de pantalla inicial

En esta pantalla únicamente se muestra una imagen de fondo que hace referencia al diplomado, al dar click en la pantalla, se enlaza a la siguiente.



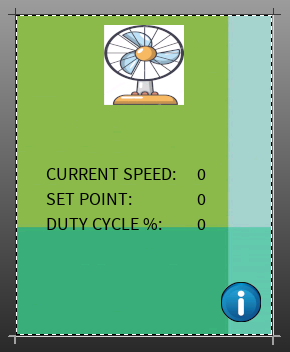
Pantalla 2 Visualización de datos

En la pantalla siguiente se muestran los valores que se están leyendo del sistema, los cuales continuamente se van actualizando.

Se muestra el set point, es decir la lectura de del potenciómetro en un valor de ADC entre 0 a 255.

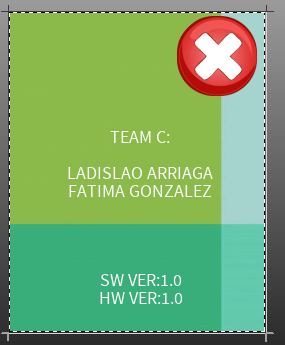
El ciclo de trabajo al que está funcionando el motor, este valor va de 0 a 100% equivalente al porcentaje de PWM.

Y finalmente, contiene un icono de información que al presionarlo enlaza a la siguiente pantalla.



Pantalla 3 Información adicional

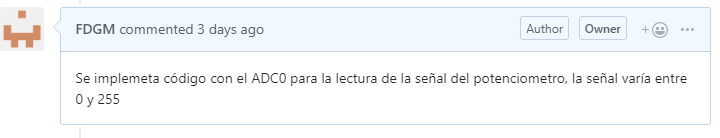
En esta pantalla se muestra al equipo de trabajo y las versiones se software. Se puede regresar a la pantalla anterior presionando el icono X (cancelar).



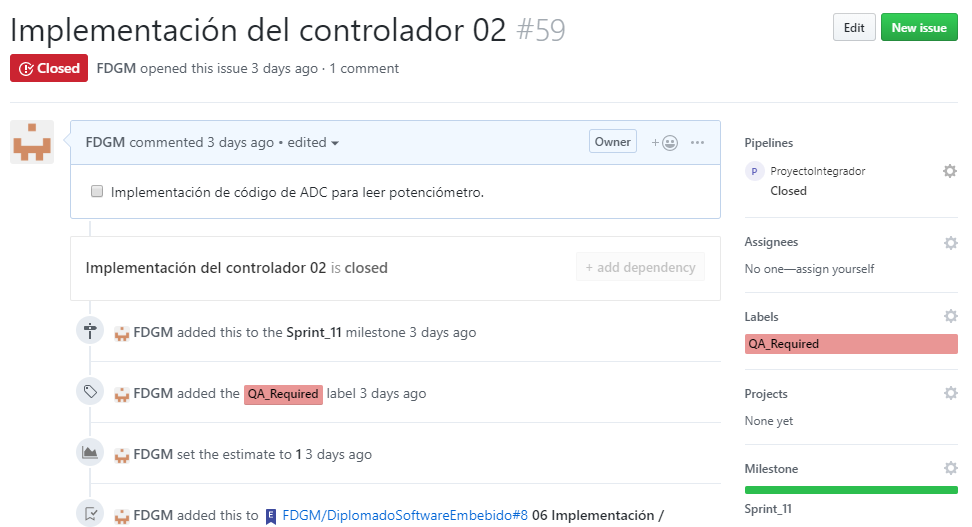
## Metodología de revisión de software

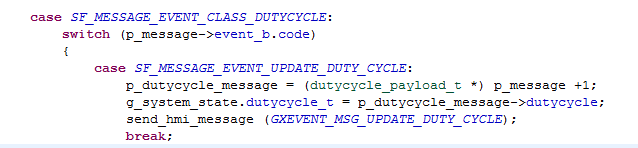
Como parte de la revisón se implementaron los siguientes 2 métodos:

1. Agregar comentarios dentro de cada ticket, como se muestra a continuación:



1. Se creó la etiqueta QA Required que se colocaba en los tickets que era necesario se realizara una revisión.

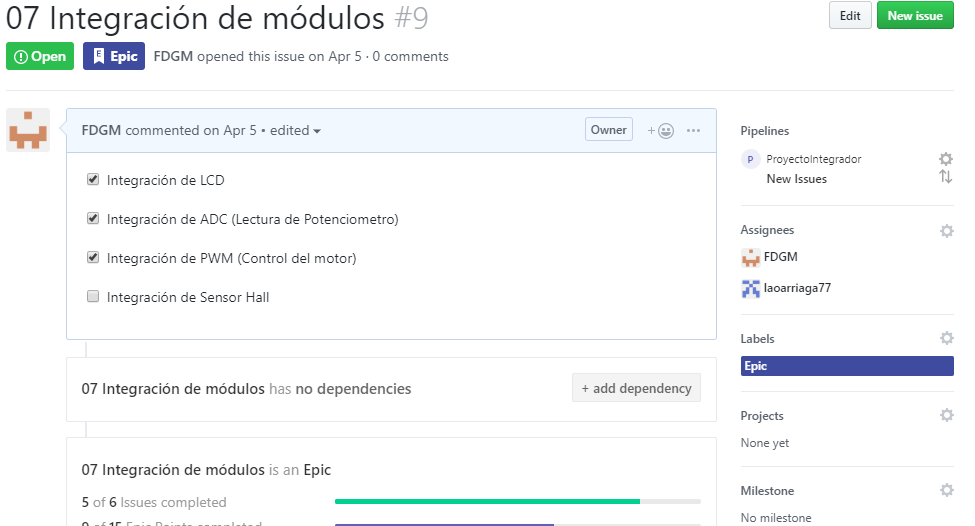


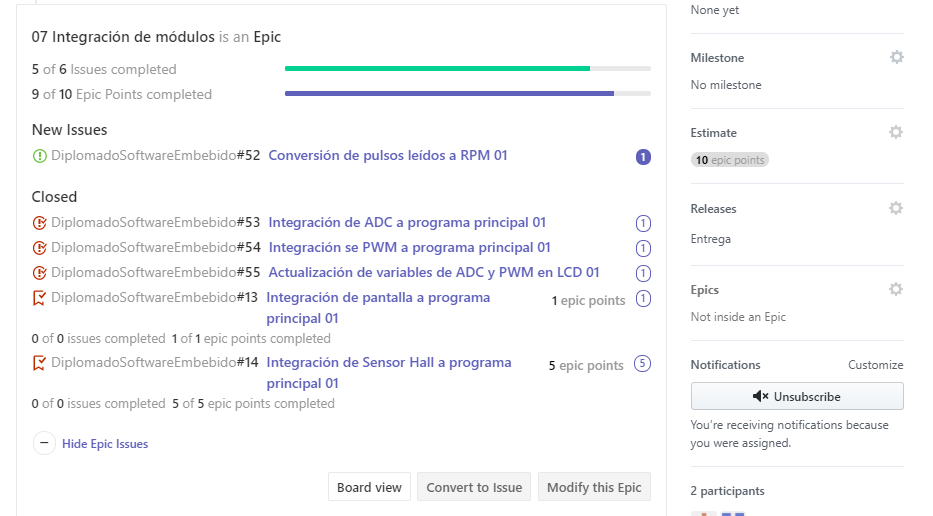


# Pruebas de software

## Pruebas de integración

Para tener documentada la integración de código al proyecto principal, se creó la epica 07 Integración de módulos, a la cua se ligaron los tickets de pruebas de integración.



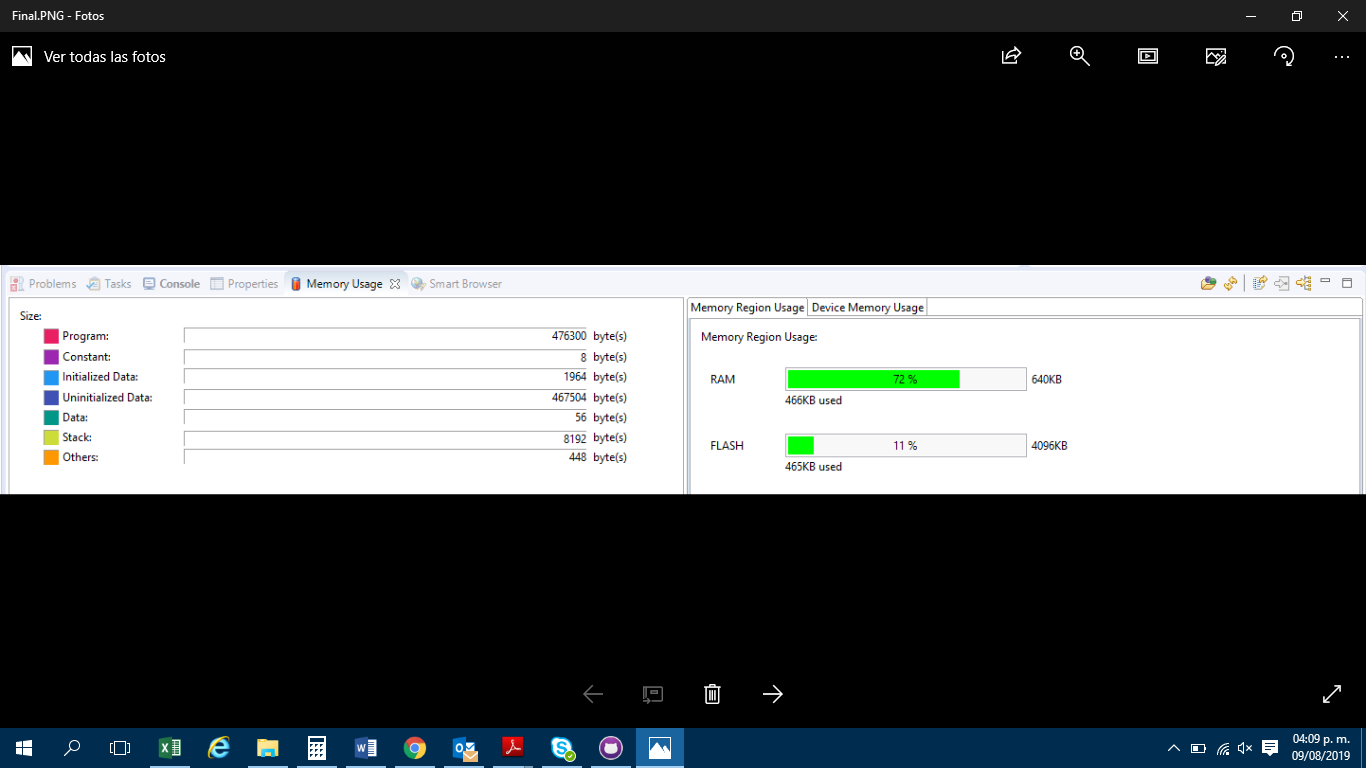


Esta integración también se ve reflejada en los branches creados en la liberación de versiones se software.

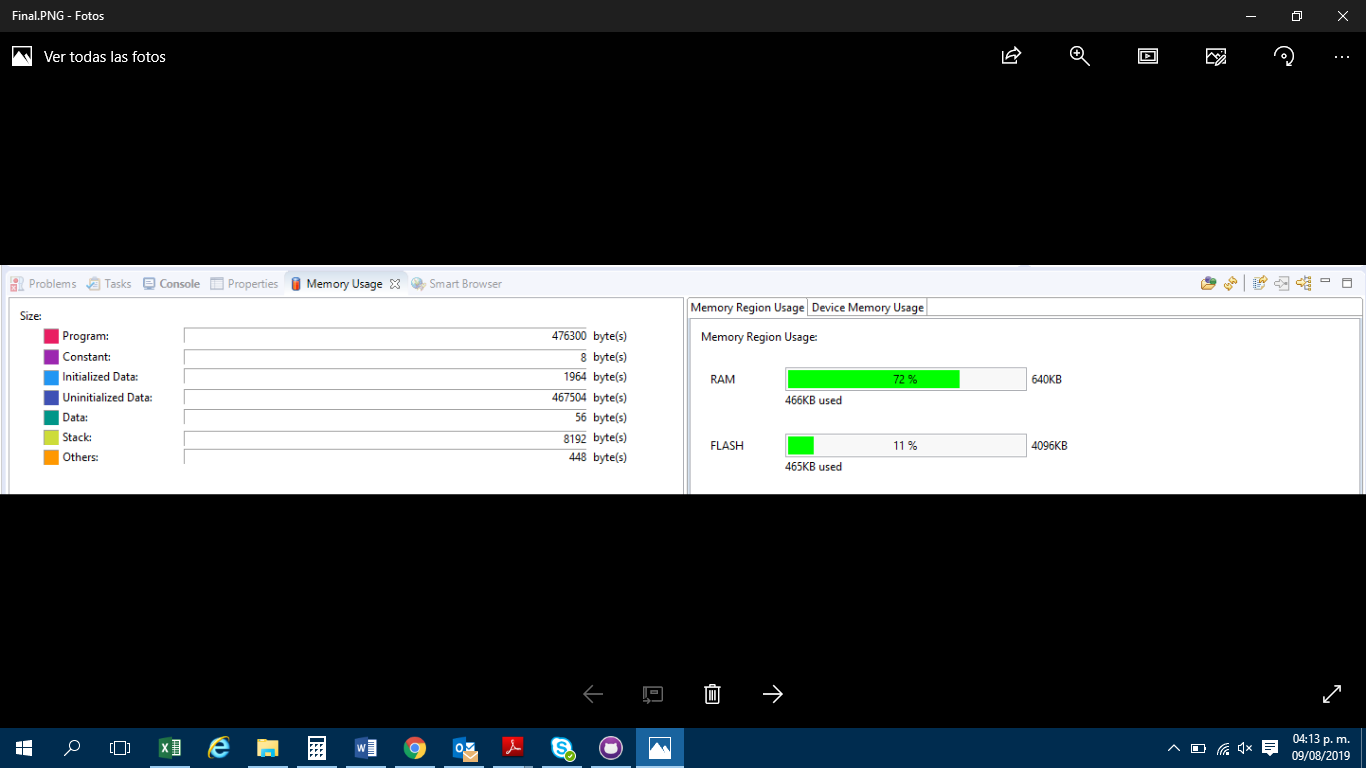
## White Box Test

### Final: % Uso de ROM, % Uso de RAM , STACK, Mapa de Memoria

Al final del proyecto el porcentaje de uso de RAM y ROM, además del Stack y Mapa de memoria quedo de la siguiente manera:



## **Final, RAM y ROM Figura 1:** % de uso de RAM y ROM.

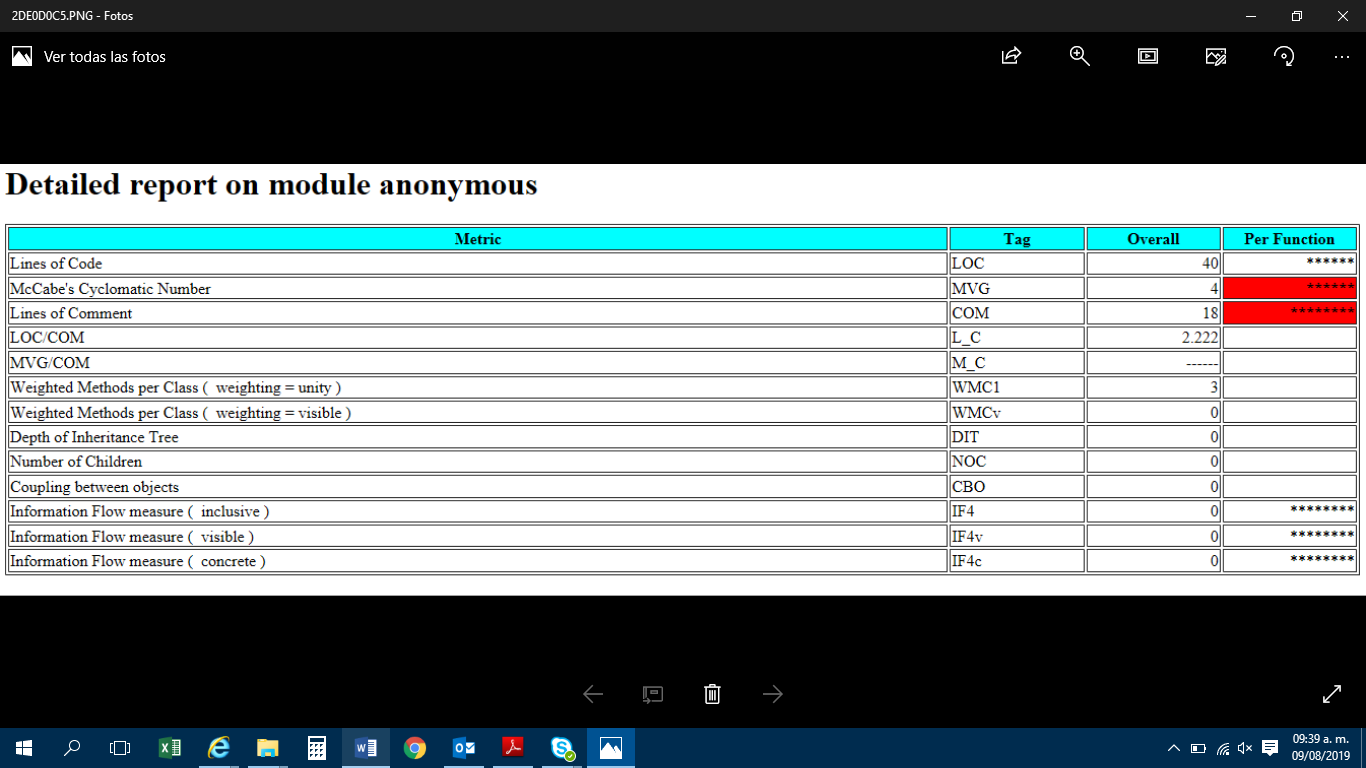


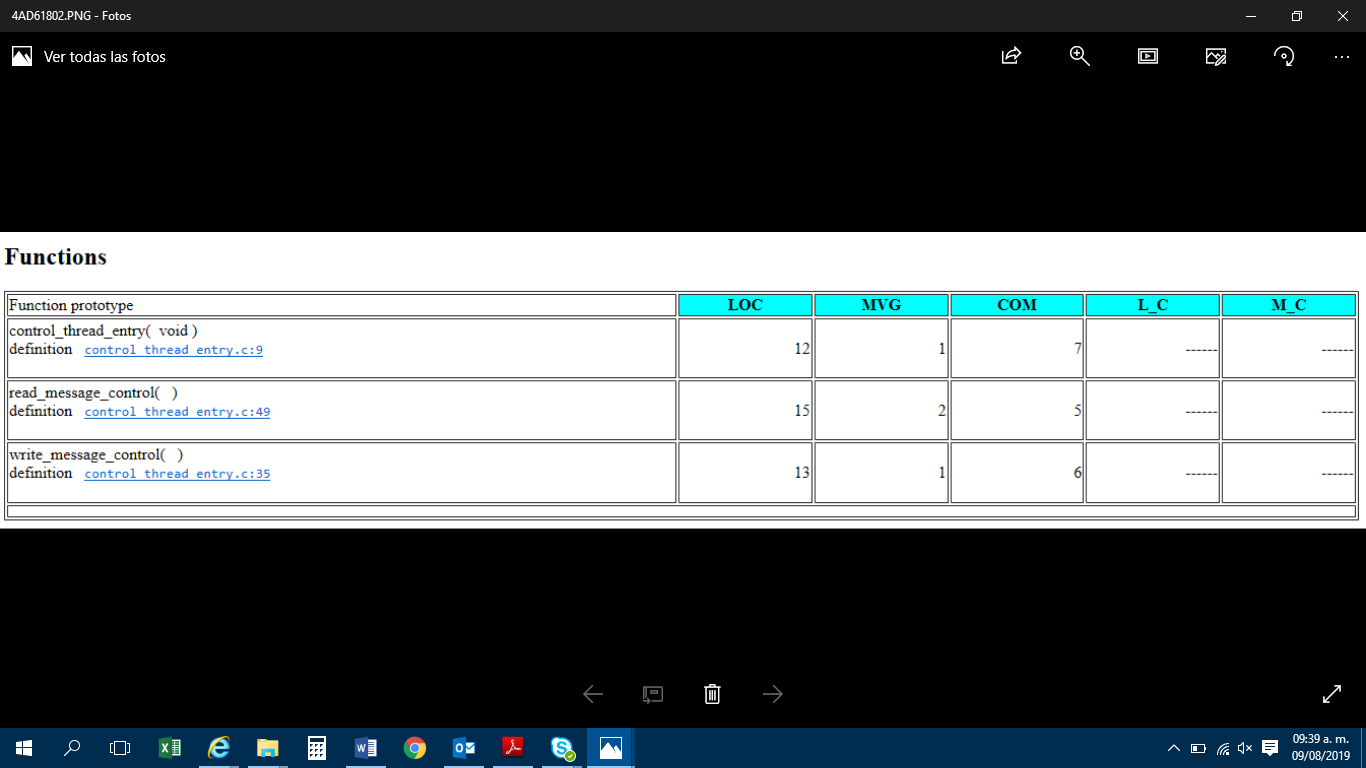
## **Figura 2:** Final Stack y Mapa de memoria.

### Throughput

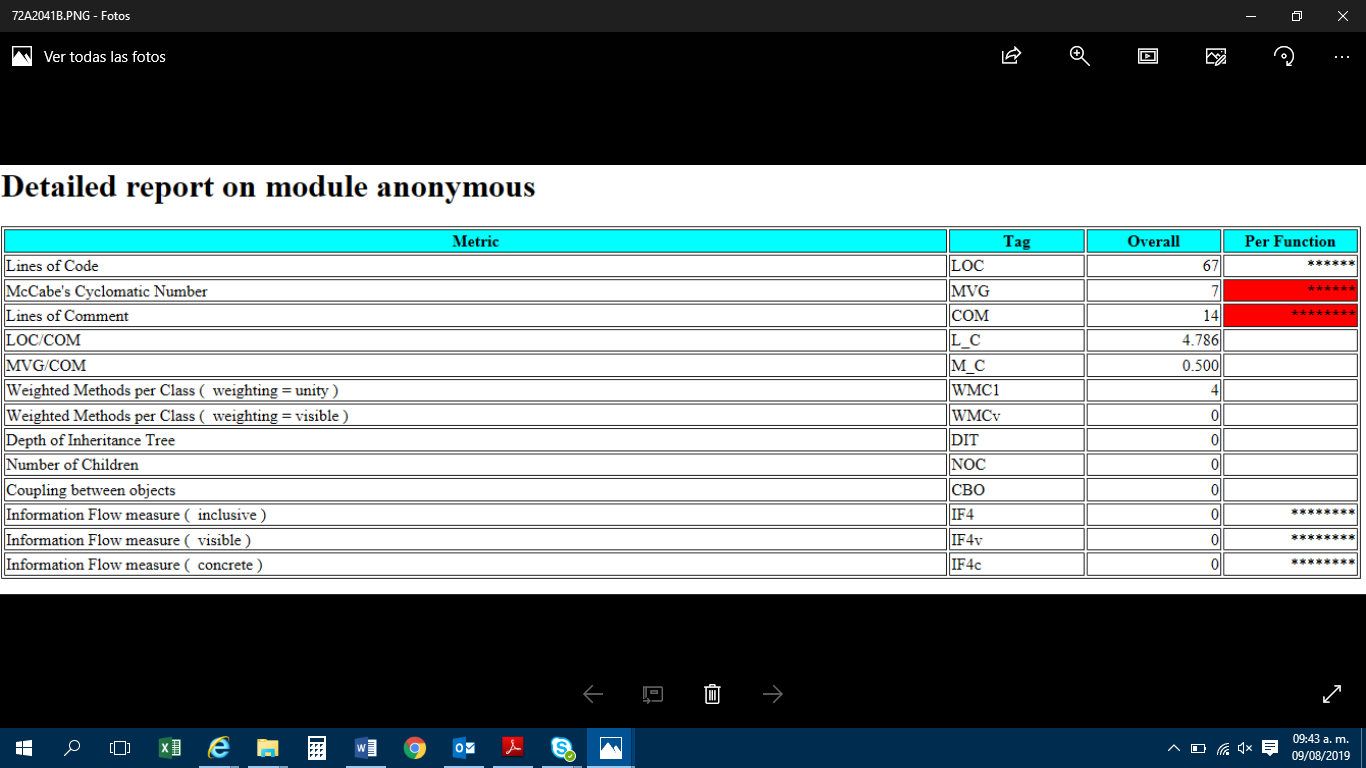
### Complejidad de Codigo

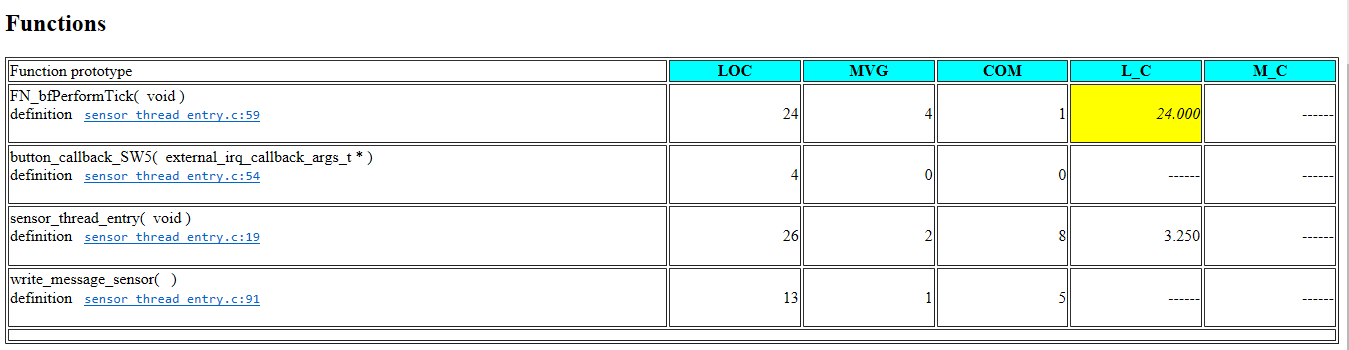
Se hace el análisis para ver la complejidad del código con el programa CCCC, para verificar si se cumple con la especificación de no mayor de 19.



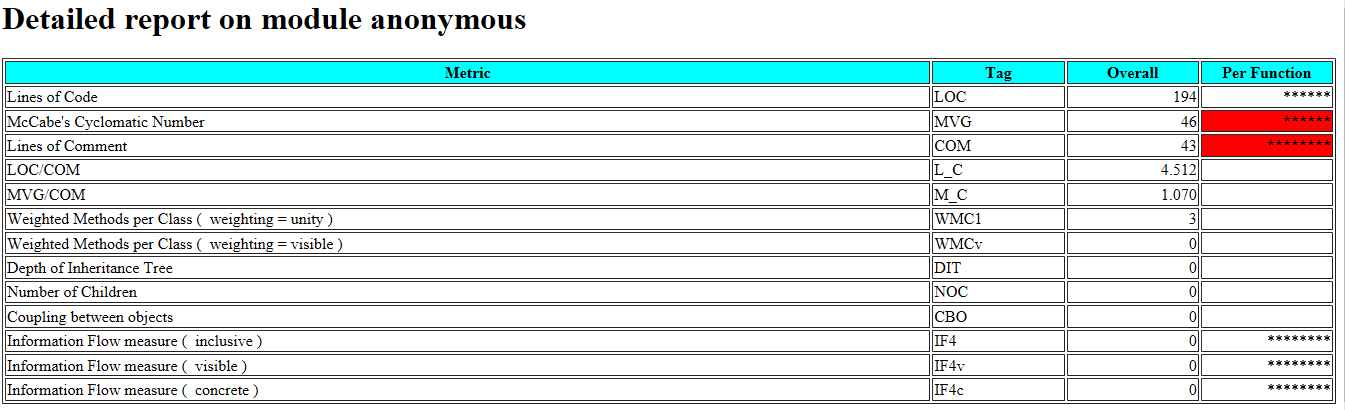


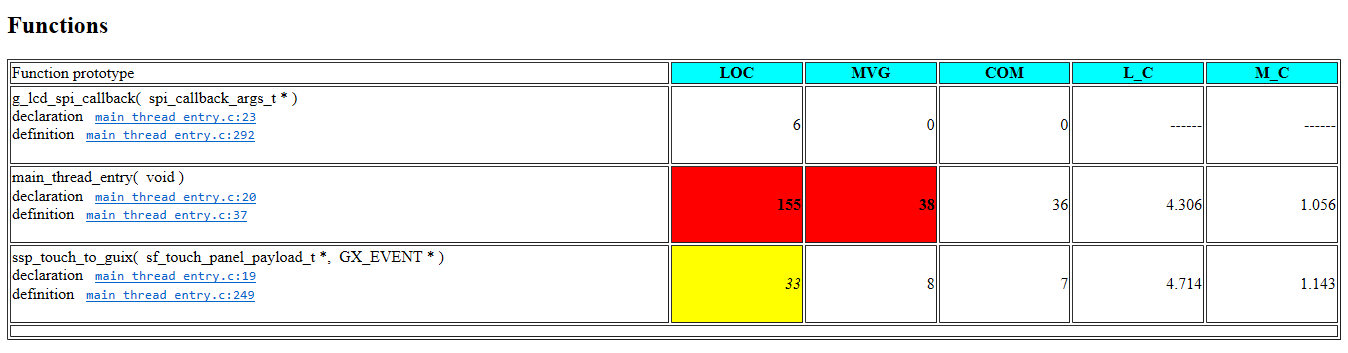
## **Analisis de Complejidad, Figura 1:** control\_thread\_entry.c





## **Analisis de Complejidad, Figura 2:** sensor\_thread\_entry.c





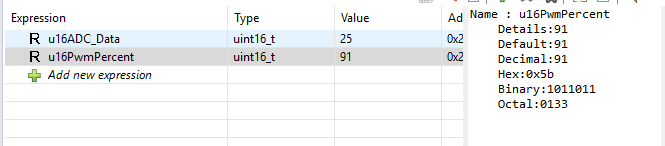
## **Analisis de Complejidad, Figura 3:** main\_thread\_entry.c

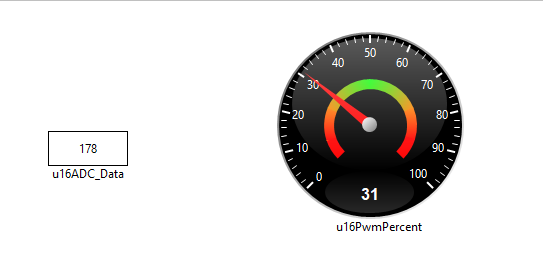
## Gray Box Test

Para las pruebas de Gray se utilizaron las herramientas del e2Studio

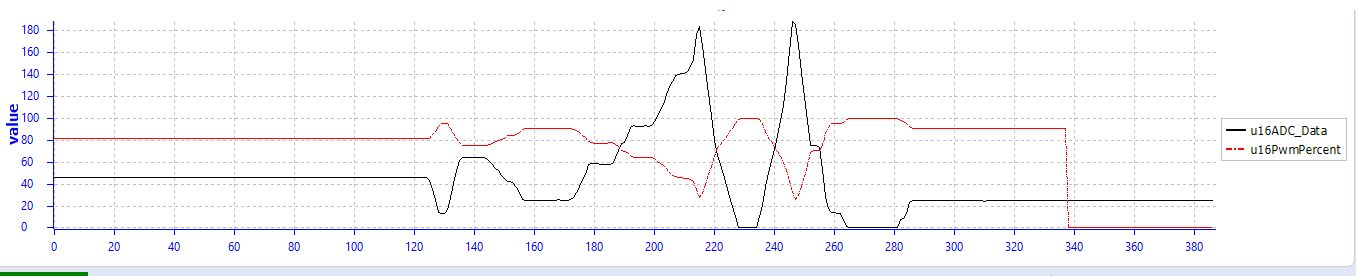
Expression y Visual Expression

Esta herramienta permite ver los valores de la variable, para ello se activó que Real Time Refresh el cual refresca los valores continuamente. Para nuestra aplicación agregamos las variables de u16ADC\_Data uint16\_t y u16PwmPercent uint16\_t.



Real Time Chart

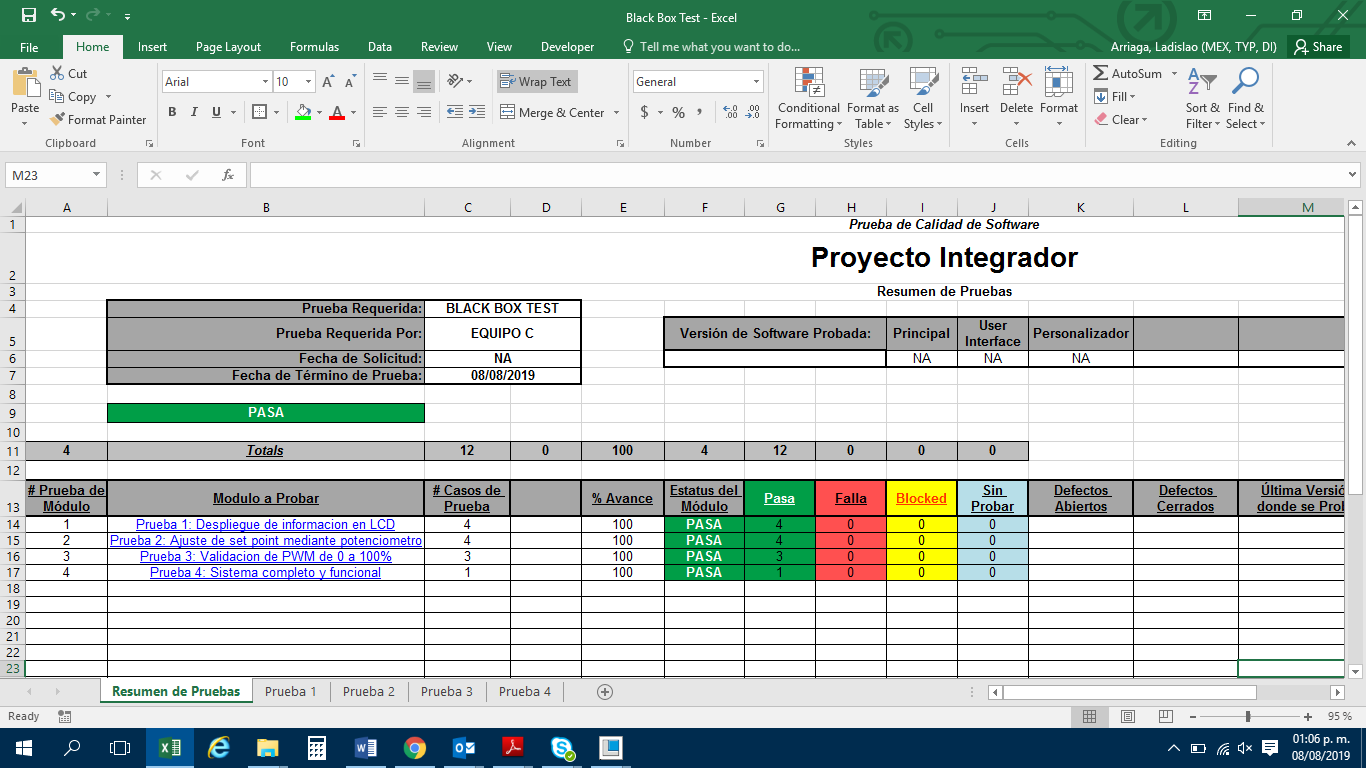
Esta herramienta permite ir graficando las variables.



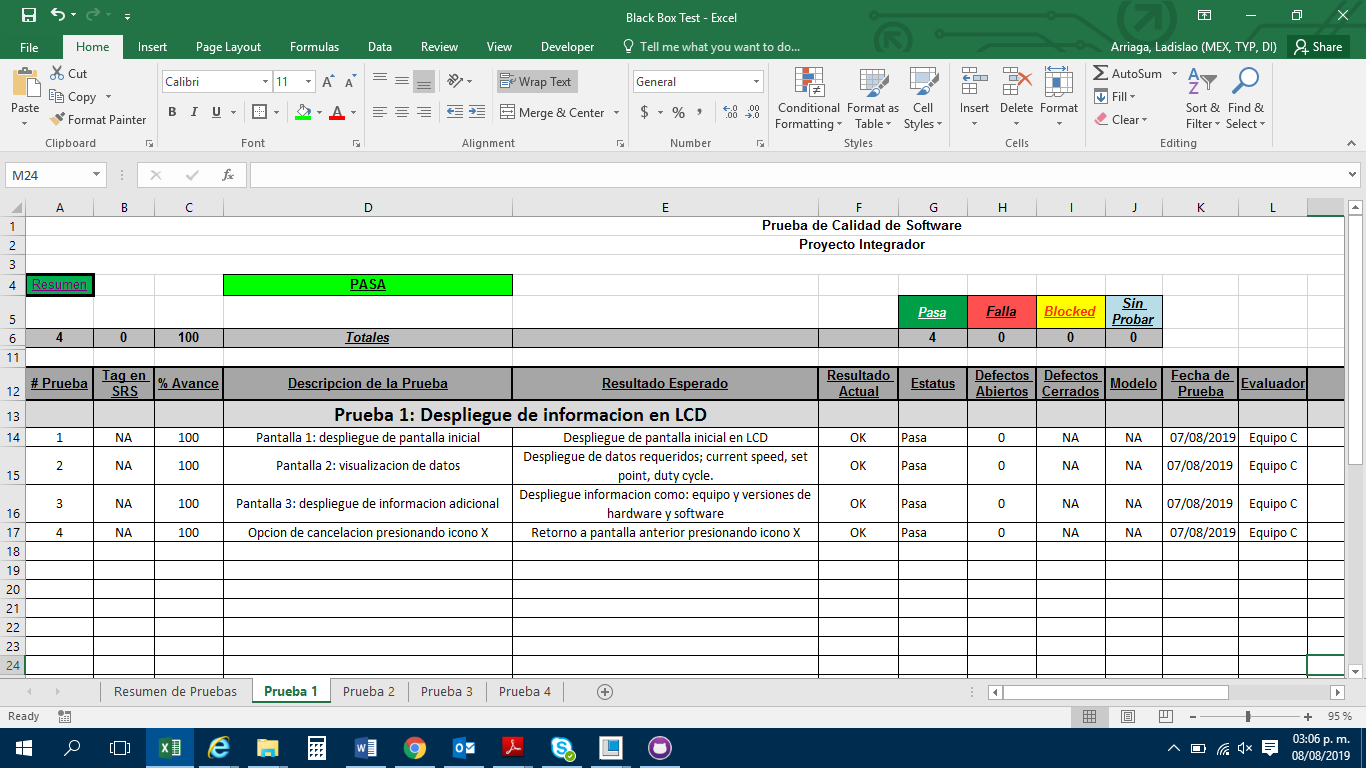
## Black Box Test

Se realizo una prueba de Black box para verificar la funcionalidad del software sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software. La verificación se realizo en partes especificas del software y en el sistema completo.

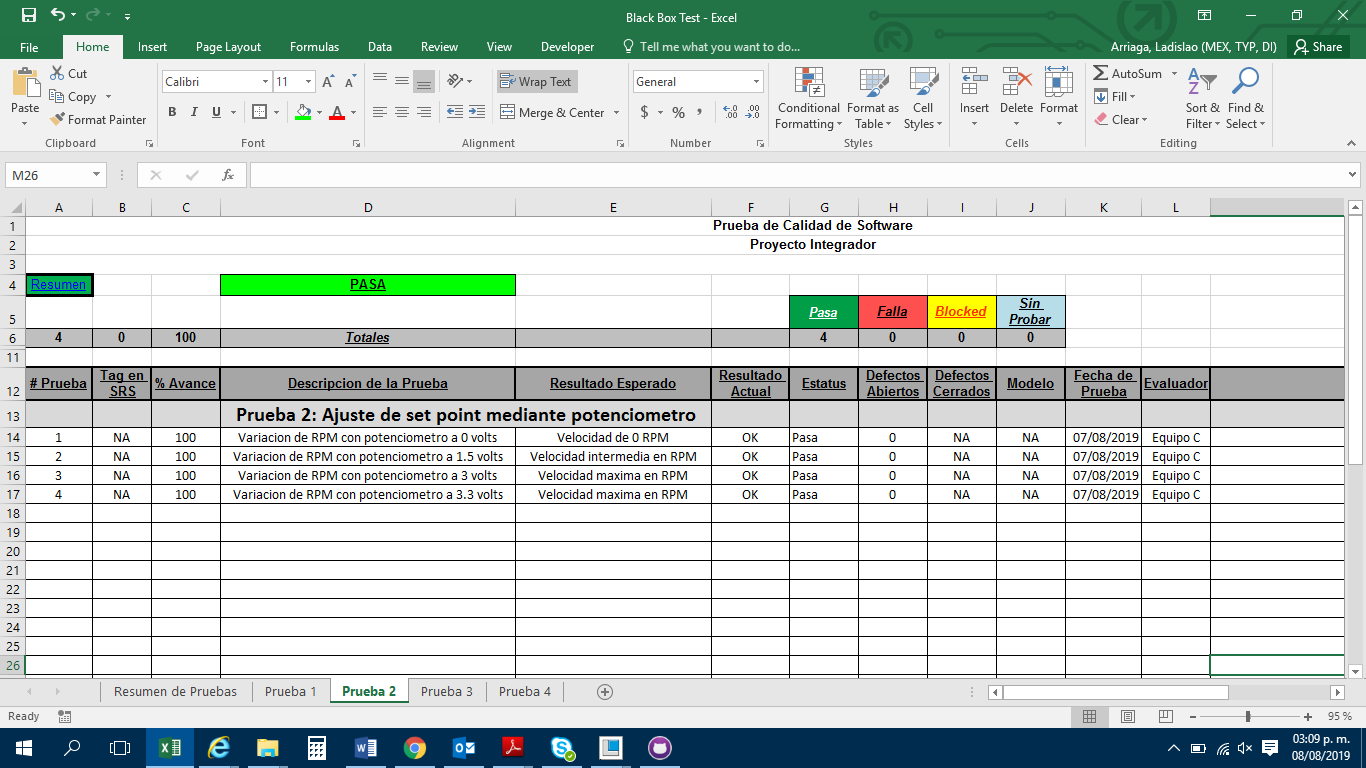
El objetivo principal de esta prueba es determinar si el software cumple con las expectativas del usuario o no.



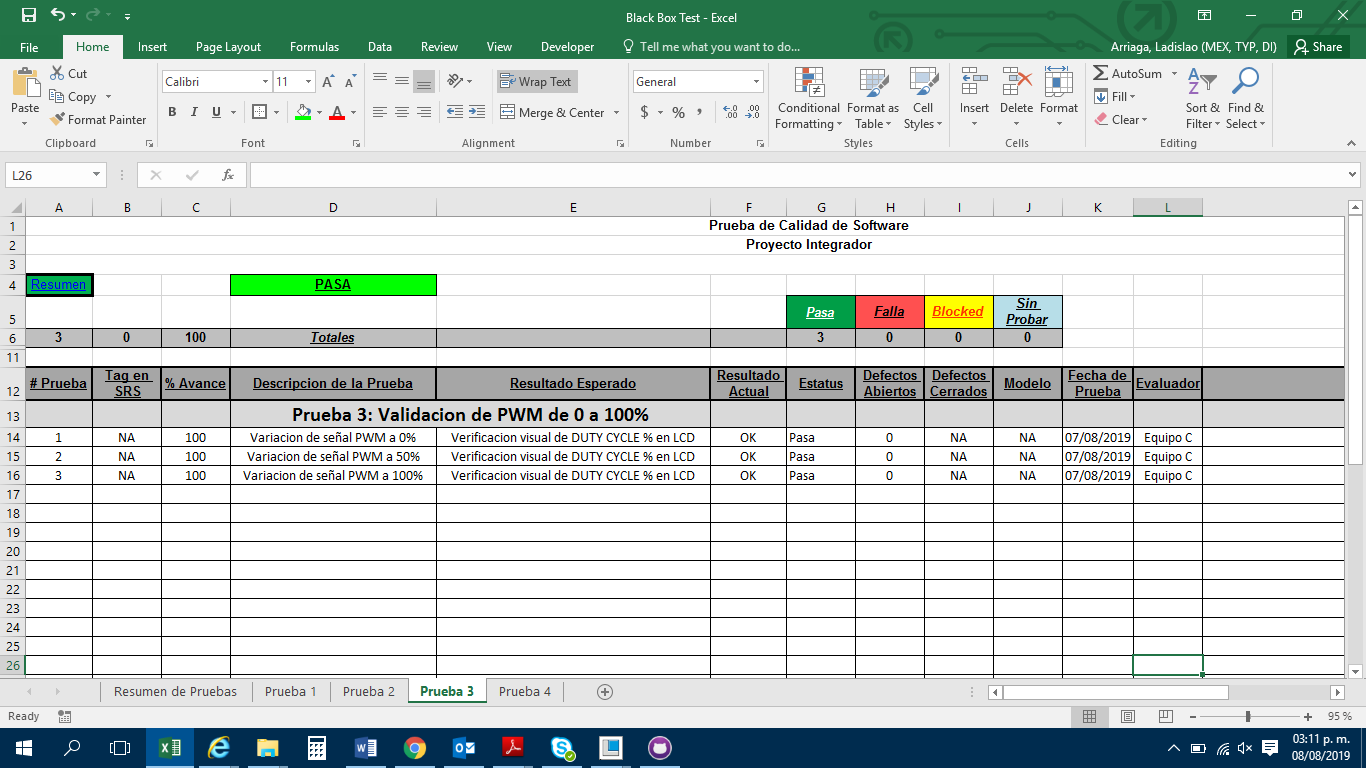
## **Black box figura 1:** Resumen de pruebas.



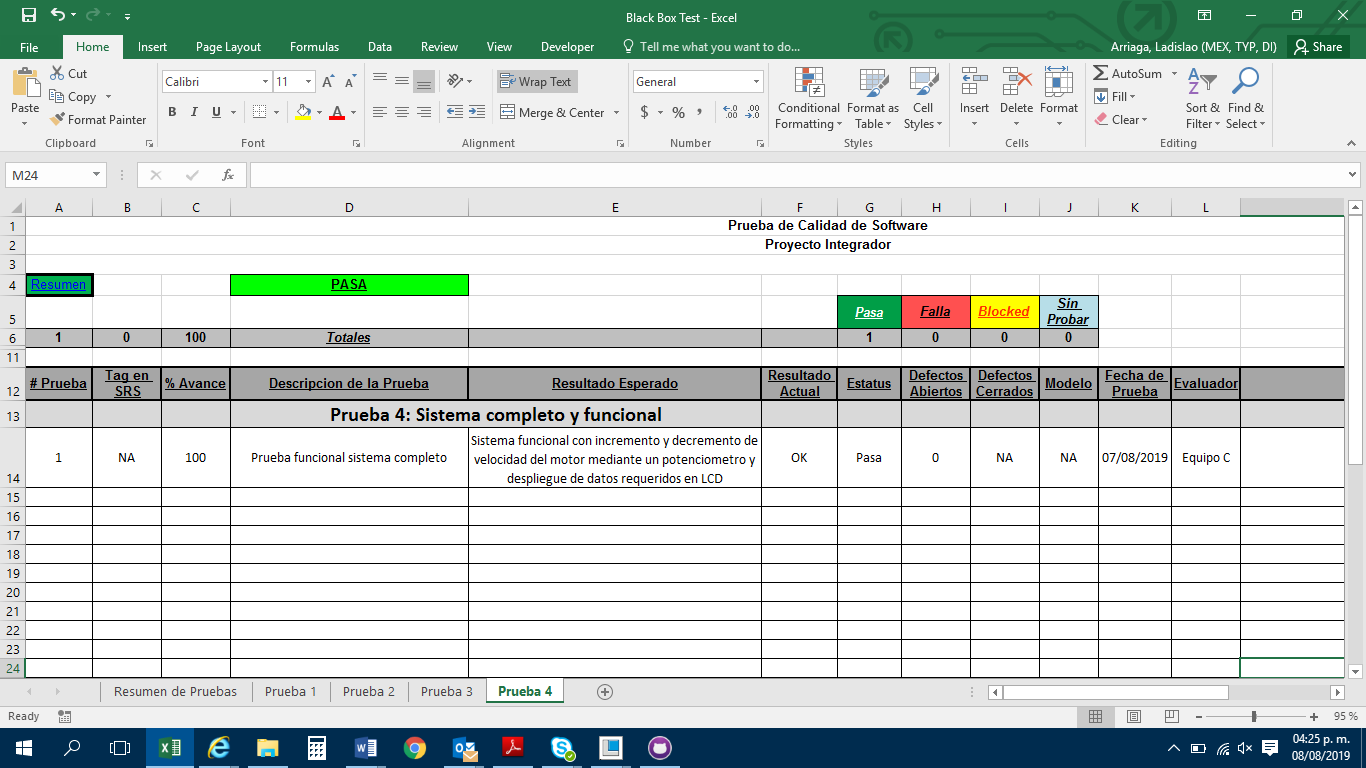
## **Black box figura 2:** Prueba # 1.



## **Black box figura 3:** Prueba # 2.



**Black box figura 4:** Prueba # 3.



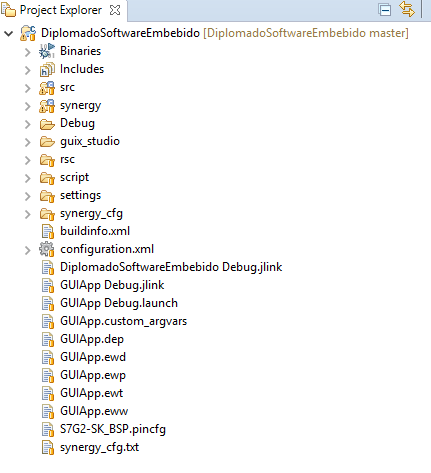
## **Black box figura 5:** Prueba # 4.

## ABA Test

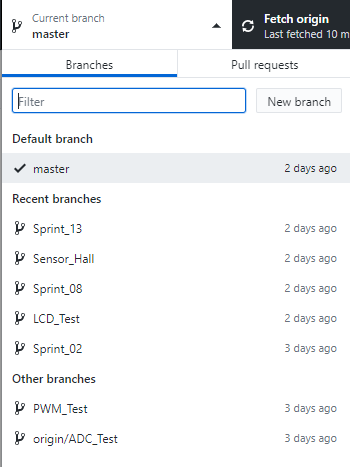
# Verificación y aseguramiento de la calidad

# Liberación de versiones de software

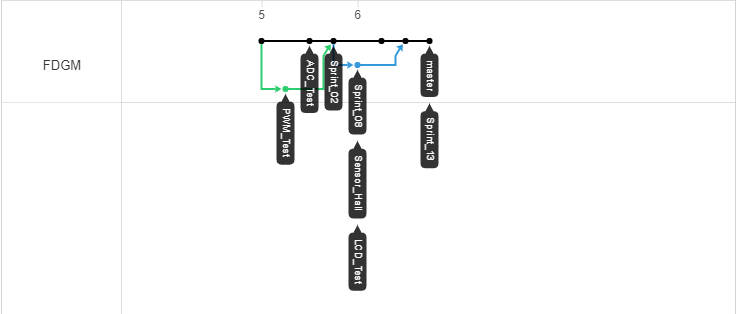
Utilizando GitHub se ligó el proyecto en e2Studio para ir actualizando las versiones de software.



En GitHub se crearon branches en los cuales se iban haciendo cambios de código y los cuales posteriormente eran actualizados a todos los ususarios, creando nuevamente un proeyecto master en cual ya contenia los cambios realizados.



A continuación, se muestra el gráfico de como fue avanzando proyecto de acuerdo a las actualizaciones.



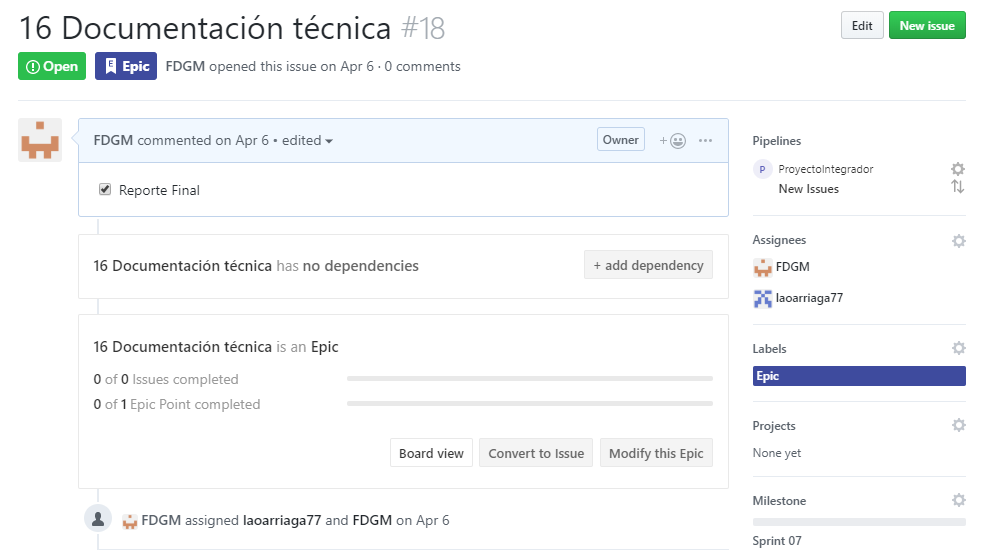
## Prueba funcional y validación en campo

Se valida el día 09 de agosto del 2019 con evaluadores la funcionalidad del proyecto. Se documenta en GitHub este punto con la epica Aceptación del usuario.



## Calendario de revisión: documental y código

Para el control de la documentación del reporte final se generó un ticket asociado al sprint 7.



Se creó el ticket Cierre de Proyecto asociado al sprint 15, que tiene fecha de finalización el día 09 de agosto de 2019, este tiene como objetivo documentar la revisión con los evaluadores.

