



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Máster Universitario en Ingeniería Informática



**TFM del Máster Universitario en
Ingeniería Informática**

**Comunicación TCP/IP con
sistemas empotrados
Documentación Técnica**



Presentado por RPC
en Universidad de Burgos — 9 de febrero
de 2019
Tutor: AMG

Índice general

Índice general	I
Índice de figuras	III
Índice de tablas	v
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	2
A.3. Estudio de viabilidad	14
Apéndice B Especificación de Requisitos	21
B.1. Introducción	21
B.2. Objetivos generales	22
B.3. Catalogo de requisitos	23
B.4. Especificación de requisitos	25
Apéndice C Especificación de diseño	35
C.1. Introducción	35
C.2. Diseño de datos	36
C.3. Diseño arquitectónico	38
C.4. Diseño procedimental	46
Apéndice D Documentación técnica de programación	47
D.1. Introducción	47
D.2. Estructura de directorios	47
D.3. Manual del programador	49

D.4. Compilación, instalación y ejecución del proyecto	55
D.5. Pruebas del sistema	55
Apéndice E Documentación de usuario	57
E.1. Introducción	57
E.2. Requisitos de usuarios	57
E.3. Instalación	57
E.4. Manual del usuario	57
Bibliografía	59

Índice de figuras

A.1. Tareas planificadas para el Sprint 1	3
A.2. Tareas planificadas para el Sprint 2	4
A.3. Tareas planificadas para el Sprint 3	5
A.4. Tareas planificadas para el Sprint 4	6
A.5. Tareas planificadas para el Sprint 5	7
A.6. Tareas planificadas para el Sprint 6	8
A.7. Sprints planificados inicialmente	10
A.8. Sprints planificados finalmente	11
A.9. Diagrama de Gantt de la planificación inicial	12
A.10. Diagrama de Gantt de la planificación final	13
 B.1. Diagrama de Casos de uso	26
C.1. Conexión TCP [19]	36
C.2. Tareas en bucle infinito	39
C.3. Bucle principal del <i>software</i>	40
C.4. Tareas del <i>software</i>	41
C.5. Relación entre MVC y el usuario	42
C.6. Diagrama de componentes del sistema	43
C.7. Introducción de datos	43
C.8. Panel para cambiar el color de los LED RGB	44
C.9. Panel para cambiar enviar un texto al LCD	44
C.10. Panel para regular el brillo de los LED PWM	45
C.11. Diseño de la interfaz	46
 D.1. Descarga de MCUXpresso IDE	50
D.2. MCUXpresso recién instalado	51
D.3. Sitio de MCUXpresso SDK Builder [26]	51

D.4. Selección de la placa de desarrollo [26]	52
D.5. Resumen del SDK completo [26]	53
D.6. Importación del SDK	53
D.7. Detalles del SDK correctamente importado	54

Índice de tablas

A.1. Costes pertenecientes a la SS	15
A.2. Coste total de personal	15
A.3. Coste del <i>hardware</i>	16
A.4. Coste del <i>software</i>	17
A.5. Coste del SE	17
A.6. Coste total del proyecto	17
A.7. Licencia Apache 2.0	19
A.8. Licencia CC BY 4.0	20
B.1. CU-1 Seleccionar SE	27
B.2. CU-2 Cambiar color RGB	28
B.3. CU-3 Mostrar mensaje	29
B.4. CU-4 Regular PWM	30
B.5. CU-5 Recibir comando	31
B.6. CU-6 Actuar sobre LED RGB	32
B.7. CU-7 Actuar sobre LCD	33
B.8. CU-8 Actuar sobre LED PWM	34
C.1. Comando LED	37
C.2. Comando MSG	37
C.3. Comando MSG	38

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este capítulo se describe el resultado obtenido tras la planificación del proyecto *software*.

La planificación toma dos vertientes. Primero se analiza la planificación temporal del proyecto. En esta fase se determina de manera general la cantidad y el tipo de tareas a realizar. Una vez descompuestas las tareas en subtareas se puede estimar el tiempo requerido para la realización de cada subtarea. Y a su vez, tareas y el proyecto completo.

Cabe destacar que la planificación inicial puede sufrir desvíos que impliquen modificarla a lo largo del desarrollo. Al usar la metodología Scrum, tras terminar cada *sprint* se puede evaluar si el desarrollo se está ciñendo a lo esperado.

Por otro parte se realiza un análisis de la viabilidad del proyecto. En todo desarrollo se ven involucrados diferentes factores que pueden determinar la viabilidad o no del proyecto. Los circunstancias analizadas son la viabilidad económica y la viabilidad legal. Se busca conocer el coste económico estimado en la realización del proyecto. Y también se quiere analizar el tipo de licenciamiento *software* que afecta al proyecto. Así como las licencias que se van a asignar.

A.2. Planificación temporal

Como se han empleado la metodología Scrum, la planificación se ha realizado entorno a los *sprints*. En la planificación inicial se establecieron de forma general el número de *sprints* y el objetivo de cada uno de ellos.

La duración asignada a los *sprint* se extendía de una semana a un mes. Como se ha usado ZenHub para la gestión del proyecto, se pudo asignar una estimación a las tareas en forma de *Story points*. La puntuación abarca desde un punto, para las tareas más simples y breves; hasta los cuarenta puntos, de las tareas más complejas y que podían ocupar la mayor parte del tiempo del *sprint*.

La planificación inicial fue la siguiente:

- *Sprint 1* Estudio e investigación
- *Sprint 2* Comunicarse con la placa usando TCP/IP
- *Sprint 3* Implementar las funciones del *hardware*
- *Sprint 4* Desarrollar la aplicación web
- *Sprint 5* Finalizar la documentación de la memoria
- *Sprint 6* Finalizar la documentación de los apéndices

A continuación se muestra la planificación específica de cada *sprints* mostrando sus objetivos, las tareas que los componen, la estimación de cada tarea y el resultado obtenido tras su finalización.

Plan del *Sprint* 1

La estimación del *sprint* se estableció en una semana de duración. Con este *sprint* se realizaba la primera toma de contacto con el proyecto. Las tareas a realizar giraban en torno al estudio de la placa de desarrollo y sobre la elección del resto de herramientas, técnicas y metodologías.

Acciones a realizar en este *sprint*:

- Estudio de las herramientas de trabajo.
- Investigación sobre la placa de desarrollo FRDM-K64F.
- Estudio sobre FreeRTOS y lwIP.
- Elección de las herramientas y *software* a emplear.

Completed Issues and Pull Requests	Story points
② Búsqueda de bibliografía [enhancement] k64f-lwip #1 Closed ↑ Sprint 1	(3)
② Elección del entorno de desarrollo [enhancement] k64f-lwip #2 Closed ↑ Sprint 1	(2)
② Estudio del entorno de desarrollo [enhancement] k64f-lwip #3 Closed ↑ Sprint 1	(2)
② Elección del repositorio [enhancement] k64f-lwip #4 Closed ↑ Sprint 1	(2)
② Elección de las herramientas de documentación [enhancement] k64f-lwip #5 Closed ↑ Sprint 1	(2)
② Elección de las herramientas de comunicación [enhancement] k64f-lwip #6 Closed ↑ Sprint 1	(1)
② Elección de la herramienta de gestión de proyectos [enhancement] k64f-lwip #7 Closed ↑ Sprint 1	(2)
② Completar el Sprint 1 [Epic] [enhancement] k64f-lwip #8 Closed ↑ Sprint 1	Not estimated

Figura A.1: Tareas planificadas para el Sprint 1

Las tareas del *sprint* se realizaron según lo previsto al plan. El estudio sobre el IDE resultó la tarea más compleja según lo estimado al tener que usar las nuevas herramientas del IDE. Al terminar sin imprevistos, se pudo planear las tareas del siguiente *sprint*.

Plan del *Sprint* 2

La estimación del *sprint* se estableció en una semana de duración. Conociendo las herramientas con la que trabajar en este *sprint* se pretendía comenzar el desarrollo.

Acciones a realizar en este *sprint*:

- Creación del proyecto de trabajo.
- Realizar las primeras comunicaciones TCP/IP.
- Realizar las operaciones remotas con la placa.

Completed Issues and Pull Requests	Story points
② Crear nuevo proyecto enhancement k64f-lwip #9 III Closed ↑ Sprint 2	(2)
⑤ Configurar el entorno del proyecto enhancement k64f-lwip #11 III Closed ↑ Sprint 2	(5)
⑤ Añadir el código para usar lwIP enhancement k64f-lwip #12 III Closed ↑ Sprint 2	(5)
③ Comutar los colores de los ledes enhancement k64f-lwip #13 III Closed ↑ Sprint 2	(3)

Figura A.2: Tareas planificadas para el Sprint 2

El *sprint* se completó de acuerdo a lo previsto. Aunque se estimó que la configuración del proyecto y el uso de lwIP no serían muy complejos su ejecución resultó más compleja, sin embargo, se realizó a tiempo para planificar y empezar el siguiente *sprint*.

Plan del Sprint 3

De nuevo, la estimación del *sprint* se estableció en una semana de duración. Como era posible enviar datos a la placa, en este *sprint* se deseaba ampliar la funcionalidad de la placa K64F con el uso de sus periféricos.

Acciones a realizar en este *sprint*:

- Configurar los LED RBG y realizar la programación para su manejo remoto.
- Configurar el LCD y realizar la programación para su manejo remoto.
- Configurar los LED PWM y realizar la programación para su manejo remoto.

Completed Issues and Pull Requests	Story points
② Configuración del hardware para el uso de señales digitales enhancement k64f-lwip #15 Closed ↑ Sprint 3	(5)
② Configuración del hardware para el uso de señales analógicas enhancement k64f-lwip #16 Closed ↑ Sprint 3	(5)
② Recepción de cadenas de caracteres y retransmisión por un bus serie. enhancement k64f-lwip #17 Closed ↑ Sprint 3	(5)
② Recepción de órdenes y realizar cambios en el hardware mediante PWM. enhancement k64f-lwip #18 Closed ↑ Sprint 3	(5)
② Completar el Sprint 3 Epic k64f-lwip #19 Closed ↑ Sprint 3	Not estimated

Figura A.3: Tareas planificadas para el Sprint 3

Este fue el primer *sprint* en desviarse de lo previsto. La configuración del bus I²C y la adaptación de la librería del LCD requirió mucho más tiempo de lo estimado. Al aumentar la duración de este *sprint* se tuvo que retrasar el inicio del siguiente. Tras terminarlo, se pudieron planificar las tareas siguientes.

Plan del *Sprint 4*

Con el cambio al desarrollo de la aplicación web y teniendo en cuenta la posibilidad de nuevos retrasos como en el *sprint* anterior, en este se decidió aumentar la planificación temporal a dos semanas.

Acciones a realizar en este *sprint*:

- Analizar el intercambio de datos necesario entre app y placa.
- Diseñar las interfaces de la aplicación.
- Implementar las funciones de comunicación.
- Desarrollar la interfaz web.

Completed Issues and Pull Requests	Story points
⌚ Analizar y declarar las operaciones de la aplicación. enhancement k64f-lwip #20 Closed ↑ Sprint 4	(3)
⌚ Crear funciones que operan con la placa. enhancement k64f-lwip #21 Closed ↑ Sprint 4	(5)
⌚ Desarrollar el panel de operaciones enhancement k64f-lwip #22 Closed ↑ Sprint 4	(8)
⌚ Verificar la funcionalidad de la aplicación web. enhancement k64f-lwip #23 Closed ↑ Sprint 4	(2)
⌚ Completar el Sprint 4 Epic k64f-lwip #24 Closed ↑ Sprint 4	Not estimated

Figura A.4: Tareas planificadas para el Sprint 4

El desarrollo de la aplicación presentó varios desafíos, el aprendizaje y programación con CSS fueron algunos de ellos. Pese a todo la duración no se desvió demasiado de lo previsto.

Plan del *Sprint* 5

Terminados el desarrollo del *software* de la placa y el desarrollo de la aplicación web faltaba completar la documentación del proyecto. Inicialmente se planificó que el *sprint* tendría una duración de dos semanas.

Acciones a realizar en este *sprint*:

- Completar los distintos capítulos de la memoria.

Completed Issues and Pull Requests	Story points
② Documentación del resumen e introducción. enhancement k64f-lwip #25 Closed ↑ Sprint 5	5
② Documentación de los objetivos del proyecto. enhancement k64f-lwip #26 Closed ↑ Sprint 5	8
② Documentación de los conceptos teóricos. enhancement k64f-lwip #27 Closed ↑ Sprint 5	13
② Documentación de las técnicas y herramientas. enhancement k64f-lwip #28 Closed ↑ Sprint 5	8
② Documentación de los aspectos relevantes. enhancement k64f-lwip #29 Closed ↑ Sprint 5	8
② Documentación de los trabajos relacionados. enhancement k64f-lwip #30 Closed ↑ Sprint 5	8
② LaTeX enhancement k64f-lwip #33 Closed ↑ Sprint 5	3
② Documentación de las conclusiones. enhancement k64f-lwip #34 Closed ↑ Sprint 5	5

Figura A.5: Tareas planificadas para el Sprint 5

El tiempo dedicado a prepararse para usar L^AT_EX fue erróneamente subestimado. Al tener que dedicar varios días más de lo previsto en este punto, el último *sprint* tuvo que empezar jornadas un tiempo más tarde.

Plan del *Sprint 6*

Completada la memoria se completaría la información proporcionada con una serie de anexos. La duración se estimó en una semana debido a los plazos temporales restantes.

Acciones a realizar en este *sprint*:

- Completar los distintos apéndices.

Remaining Issues and Pull Requests	Story points
① Documentación del Plan de proyecto. enhancement k64f-lwip #35 In Progress ↑ Sprint 6	8
① Completar el Sprint 6 Epic enhancement k64f-lwip #36 Backlog ↑ Sprint 6	Not estimated
① Documentación de los Requisitos. enhancement k64f-lwip #37 Backlog ↑ Sprint 6	8
① Documentación del Diseño. enhancement k64f-lwip #38 Backlog ↑ Sprint 6	8
① Documentación del Manual del programador. enhancement k64f-lwip #39 Backlog ↑ Sprint 6	8
① Documentación del Manual de Usuario. enhancement k64f-lwip #40 Backlog ↑ Sprint 6	8

Figura A.6: Tareas planificadas para el Sprint 6

Completada la documentación se pudo proceder a la conclusión satisfactoria del proyecto.

Planificación temporal completa

En las siguientes páginas se pueden comparar la planificación inicial y la final.

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1	🏃	Sprint #1	7 días	lun 26/11/18	dom 02/12/18
2	💻	Estudio de las herramientas de trabajo	3 días	lun 26/11/18	mié 28/11/18
3	💻	Investigar sobre la placa K64F	2 días	jue 29/11/18	vie 30/11/18
4	💻	Estudio sobre FreeRTOS y lwIP	2 días	sáb 01/12/18	dom 02/12/18
5	💻	Elección de las herramientas y software a emplear	7 días	lun 26/11/18	dom 02/12/18
6	🏃	Sprint #2	7 días	lun 03/12/18	dom 09/12/18
7	💻	Creación del proyecto de trabajo	1 día	lun 03/12/18	lun 03/12/18
8	💻	Realizar las primeras comunicaciones TCP/IP	4 días	mar 04/12/18	vie 07/12/18
9	💻	Realizar operaciones remotas con la placa	2 días	sáb 08/12/18	dom 09/12/18
10	🏃	Sprint #3	7 días	lun 10/12/18	dom 16/12/18
11	💻	Configurar y programar LED RGB	2 días	lun 10/12/18	mar 11/12/18
12	💻	Configurar y programar LCD	3 días	mié 12/12/18	vie 14/12/18
13	💻	Configurar y programar LED PWM	2 días	sáb 15/12/18	dom 16/12/18
14	🏃	Sprint 4	14 días	lun 17/12/18	dom 30/12/18
15	💻	Análisis de la web app	2 días	lun 17/12/18	mar 18/12/18
16	💻	Diseño de las interfaces	2 días	mié 19/12/18	jue 20/12/18
17	💻	Implementación de las funciones	4 días	vie 21/12/18	lun 24/12/18
18	💻	Desarrollo gráfico de la web	6 días	mar 25/12/18	dom 30/12/18
19	🏃	Sprint 5	7 días	lun 31/12/18	dom 06/01/19
20	💻	Completar la memoria	7 días	lun 31/12/18	dom 06/01/19
21	🏃	Sprint 6	7 días	lun 07/01/19	dom 13/01/19
22	💻	Completar los apéndices	7 días	lun 07/01/19	dom 13/01/19

Figura A.7: Sprints planificados inicialmente

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		Sprint #1	7 días	lun 26/11/18	dom 02/12/18
2		Estudio de las herramientas de trabajo	3 días	lun 26/11/18	mié 28/11/18
3		Investigar sobre la placa K64F	2 días	jue 29/11/18	vie 30/11/18
4		Estudio sobre FreeRTOS y lwIP	2 días	sáb 01/12/18	dom 02/12/18
5		Elección de las herramientas y software a emplear	7 días	lun 26/11/18	dom 02/12/18
6		Sprint #2	7 días	lun 03/12/18	dom 09/12/18
7		Creación del proyecto de trabajo	1 día	lun 03/12/18	lun 03/12/18
8		Realizar las primeras comunicaciones TCP/IP	4 días	mar 04/12/18	vie 07/12/18
9		Realizar operaciones remotas con la placa	2 días	sáb 08/12/18	dom 09/12/18
10		Sprint #3	10 días	lun 10/12/18	mié 19/12/18
11		Configurar y programar LED RGB	2 días	lun 10/12/18	mar 11/12/18
12		Configurar y programar LCD	5 días	mié 12/12/18	dom 16/12/18
13		Configurar y programar LED PWM	3 días	lun 17/12/18	mié 19/12/18
14		Sprint 4	19 días	jue 20/12/18	lun 07/01/19
15		Análisis de la web app	2 días	jue 20/12/18	vie 21/12/18
16		Diseño de las interfaces	3 días	sáb 22/12/18	lun 24/12/18
17		Implementación de las funciones	5 días	mar 25/12/18	sáb 29/12/18
18		Desarrollo gráfico de la web	9 días	dom 30/12/18	lun 07/01/19
19		Sprint 5	29 días	mar 08/01/19	mar 05/02/19
20		Completar la memoria	29 días	mar 08/01/19	mar 05/02/19
21		Sprint 6	6 días	mié 06/02/19	lun 11/02/19
22		Completar los apéndices	6 días	mié 06/02/19	lun 11/02/19

Figura A.8: Sprints planificados finalmente

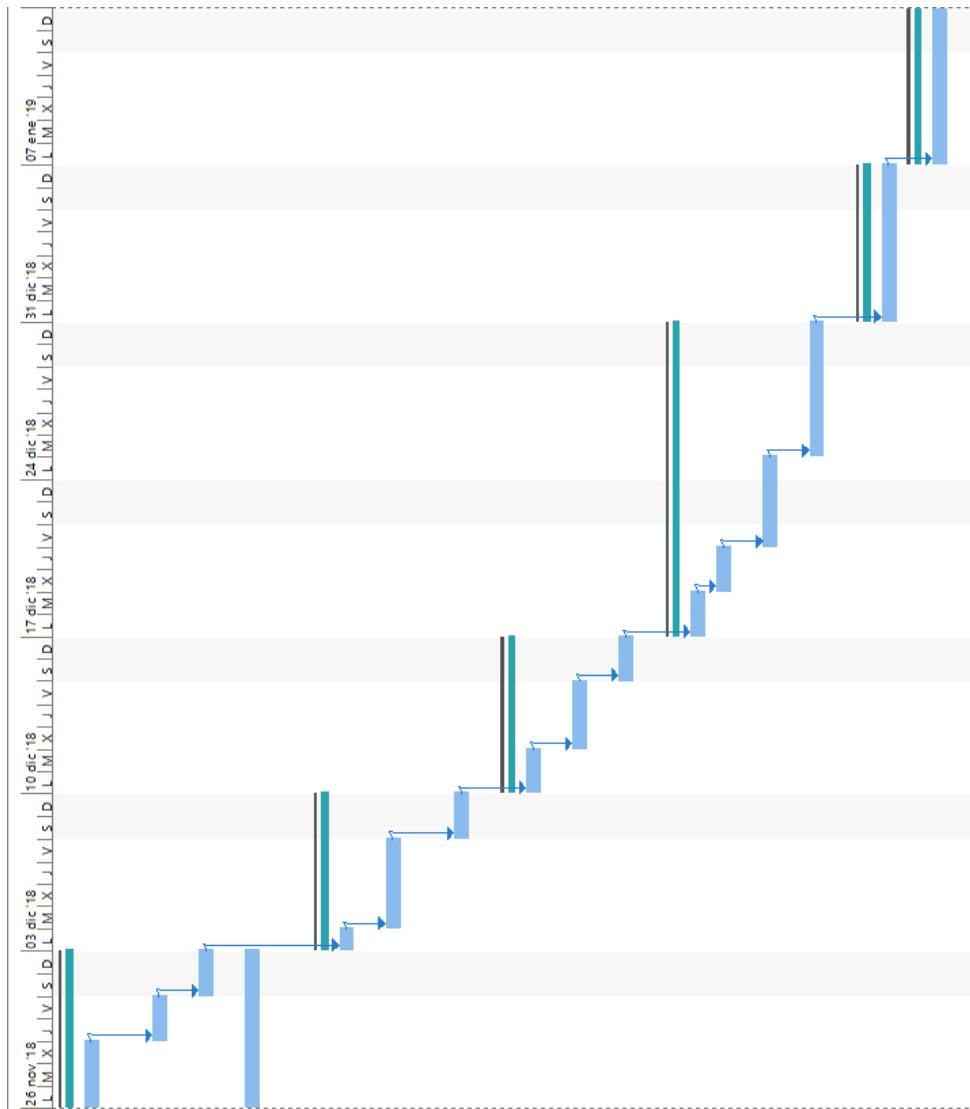
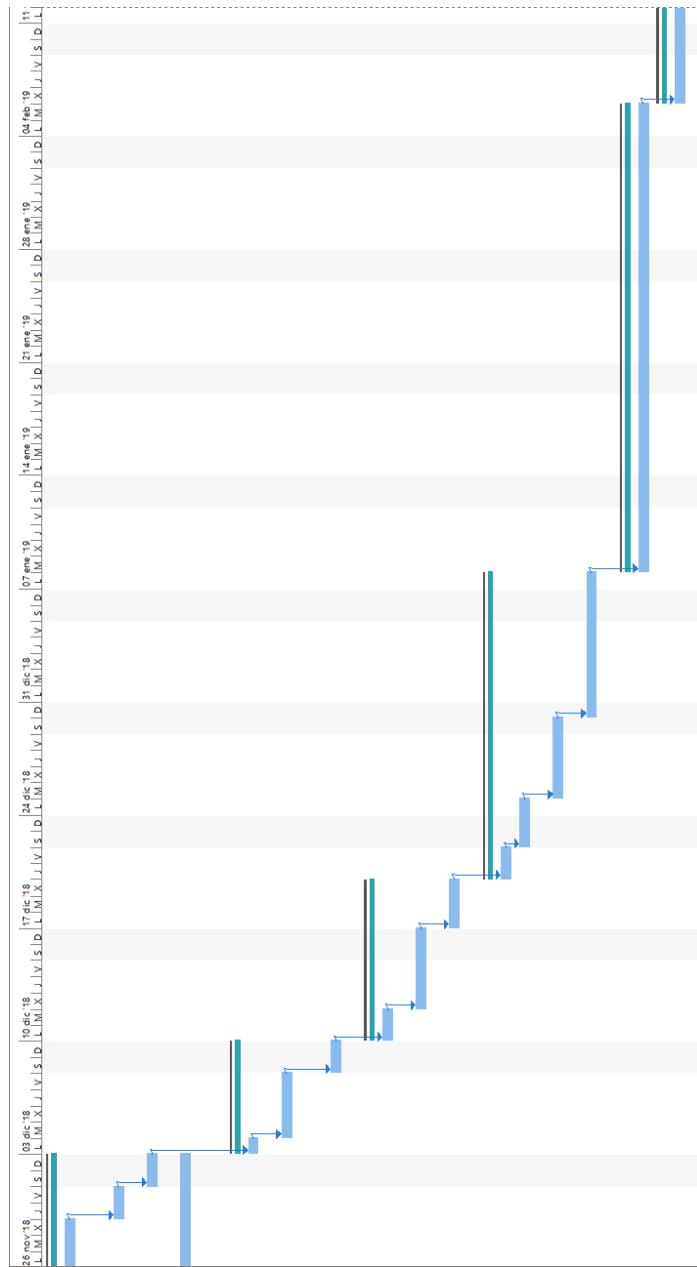


Figura A.9: Diagrama de Gantt de la planificación inicial



A.3. Estudio de viabilidad

Viabilidad económica

Para determinar la viabilidad económica del proyecto, en este apartado se computan los gastos previstos durante el desarrollo y los posibles beneficios si los hubiera. De haberse realizado en una empresa, los costes se calcularían como se muestra a continuación.

Coste de personal

El desarrollo se ha realizado por una sola persona en un tiempo aproximado de 90 días, así que, todos los costes de personal son relativos a esa única persona. El salario mensual bruto asignado ha sido de 2200€.

De manera simplificada, el salario neto se puede calcular de la siguiente forma:

$$\text{salario bruto} - \text{IRFP} - \text{SS} = \text{salario neto} \quad (\text{A.1})$$

Por tanto, para un mes dado el salario neto a percibir sería el siguiente:

$$2200\text{€} - 14\%^{\textcolor{red}{1}} - 6,35\% = 1752,30\text{€} \quad (\text{A.2})$$

Costes pertenecientes a la SS

Sobre el salarios se producen ciertas retenciones y pagos a la Seguridad Social, algunos conceptos corren a cargo de la empresa contratante y otros por parte del trabajador. A continuación de calculan sus importes.

Para el cálculo de las retenciones se toma como referencia la tablas de bases de cotización ofrecida por la propia Seguridad Social [2]. En concreto datos del primer grupo de cotización: “Ingenieros y Licenciados. Personal de alta dirección no incluido en el artículo 1.3.c) del Estatuto de los Trabajadores”. Este grupo tiene unas bases mínimas de 1466,40€/mes. Y unas máximas de 4070,10€/mes.

¹Usado el tramo más alto de la escala autonómica de Castilla y León aplicable a la base liquidable general del IRPF [1]

Concepto	Empresa	Trabajador
Contingencias comunes	23,60 %	4,70 %
Desempleo	5,50 %	1,55 %
FOGASA	0,20 %	0,00 %
Formación	0,60 %	0,10 %
Total	29,9 %	6,35 %

Tabla A.1: Costes pertenecientes a la SS

Por consiguiente, la empresa tendría que añadir a los costes el 23,9 % del salario del trabajador y el trabajador sufriría una retención del 6,35 %.

Coste total de personal

El coste total se calcula con la siguiente fórmula:

$$(salario mensual + retenciones ss) \times n^o \text{ meses} = \text{coste total} \quad (\text{A.3})$$

De modo que el coste total se obtiene así:

$$(2200\text{€} + 657,8\text{€}) \times 3 = 8573,4\text{€} \quad (\text{A.4})$$

La tabla siguiente recoge diferentes costes asociados al personal y su coste total.

Concepto	Coste
Salarios	2200€
Seguridad Social	657,8€
Meses	3 meses
Total	8573,4 €

Tabla A.2: Coste total de personal

Coste del hardware

Para realizar el desarrollo se precisa contar con varios dispositivos. Aunque en la Ley del Impuesto sobre Sociedades [3] se permite un máximo de 8

años para amortizar equipos para procesos de información, se considera la que renovación de *hardware* se produce en un periodo más corto de 4 años. Así pues, la amortización se calcula en función de esos 4 años.

El coste mensual de un dispositivo se calcula así:

$$\text{coste del dispositivo} / \text{periodo de amortización (en meses)} \quad (\text{A.5})$$

Por lo tanto, el coste de un dispositivo es el siguiente:

$$\text{coste mensual amortizado} \times n^{\circ}\text{meses} \quad (\text{A.6})$$

En el desarrollo solo se va a necesitar una estación de trabajo, así pues, el coste total del *hardware* es el mostrado en la tabla a continuación.

<i>Hardware</i>	Coste	Coste amortizado
Estación de trabajo	1250 €	104,16 €
Total	1250 €	104,16 €

Tabla A.3: Coste del *hardware*

Coste del *software*

El desarrollo necesita de varios programas y aplicaciones. De igual manera que el *hardware*, la Ley del Impuesto sobre Sociedades [3] indica el máximo de años para realizar la amortización. En sistemas y programas informáticos es de 6 años. Pero coincidiendo con el periodo de renovación del *hardware*, la amortización se calcula en un periodo menor de 4 años.

Las fórmulas para calcular las amortizaciones son equivalentes a las usadas en el *hardware* A.5 y A.6.

Otra parte de *software* se emplea bajo suscripción, en los cálculos solo se computa el coste de los meses que se usó dicha suscripción.

Hay que tener en cuenta, que gran parte del *software* utilizado se publica bajo licencias que permiten su uso sin coste y que gracias a esto se reducen significativamente los costes del *software*. En consecuencia solo se calculan los costes del *software* que no es gratuito.

<i>Software</i>	Coste	Coste amortizado
Windows 10 Pro[4]	259€	21,58€
Office 365[5]	126€	31,5€
Adobe CC[6]	359,88€	89,97€
Total	744,88€	143,05 €

Tabla A.4: Coste del *software*

Coste del sistema empotrado

El sistema empotrado se ha realizado con la ayuda de diversos componentes que también tienen unos costes de adquisición asociados.

Componente	Coste
Placa FRDM-K64F[7]	32,52€
Placa Arduino Basic I/O[8]	19,00€
Pantalla LCD[9]	8,91€
Placas de pruebas[10]	12,24€
Cables puente[11]	4,98€
Cable de par trenzado[12]	4,88€
Total	82,53 €

Tabla A.5: Coste del SE

Coste total del proyecto

Sumando todos los costes anteriores se puede calcular el coste total de todo el proyecto.

Tipo de coste	Coste
Personal	8573,4€
<i>Hardware</i>	104,16€
<i>Software</i>	143,05€
Componentes del SE	82,53€
Total	8903,14 €

Tabla A.6: Coste total del proyecto

Beneficios del proyecto

El código fuente del sistema empotrado (SE) y de la aplicación web se encuentran disponibles abiertamente. Además, el SE tampoco tiene definida una función claramente comercial. Por ello, no se considera que el proyecto pueda tener un beneficio económico directo.

Esto tampoco quiere decir que los gastos se hagan a fondo perdido. Más bien, se pueden considerar los gastos como una inversión en investigación y formación. Competencias que permitirán en el futuro crear nuevos SE que rentabilicen todos los costes.

Viabilidad legal

Al desarrollar un *software* hay que tener en consideración las implicaciones legales que se presentan al usar *software* de terceros.

Las licencias sirven como instrumento para establecer los términos y condiciones en los que se pueden utilizar el *software* licenciado. Para poder definir qué licencias usar en el proyecto es necesario conocer las licencias utilizadas y la restricciones que pueden imponer.

Licencias utilizadas en el desarrollo del SE

Para desarrollar el *software* de la placa se ha utilizado de código fuente de terceros que se describe a continuación.

- El código generado por MCUXpresso se proporciona bajo la licencia “The 3-Clause BSD License” (BSD-3), también conocida como la “New BSD License” o “Modified BSD License”.
- Otra parte del código generado por MCUXpress se licencia con “The Clear BSD License”. La licencia es similar a BSD-3, pero indica expresamente que no se conceden derechos sobre patentes.
- El código perteneciente a FreeRTOS es propiedad de Amazon.com, Inc. Se permite el uso, copia, modificación, publicación, distribución, volver a licenciar y comercializar, manteniendo siempre el aviso de derechos de autor.
- El código de lwIP se entrega con la licencia BSD-3.
- Parte del código relacionado con el procesador de la placa pertenece a ARM Limited. El código se proporciona con la licencia Apache 2.0.

Así pues, se encuentran licencias que permiten el uso, copia, modificación y redistribución del código. Como BSD-3 o Apache 2.0

Licencia para el SE

Teniendo en cuenta que el código fuente del *software* del SE se pretende que sea abierto, se va a utilizar una licencia en línea con las mostradas anteriormente.

En concreto se va a emplear la licencia Apache 2.0. Sus características principales se pueden ver resumidas en la siguiente tabla.

Permisos	Condiciones	Limitaciones
Uso comercial	Aviso de licencia	Uso de marcas registradas
Modificación	y derechos de autor	Responsabilidad
Distribución	Declaración de los cambios	Garantía
Uso en patentes		
Uso privado		

Tabla A.7: Licencia Apache 2.0

Licencia para la aplicación web

En el caso de la aplicación web no se ha requerido de bibliotecas o librerías de terceros como en el caso del SE. De igual forma que el SE, la aplicación web se licencia con Apache 2.0

Aviso de Apache

Para aplicar la licencia Apache [13] es necesario adjuntar un archivo de texto con la licencia con el nombre “LICENSE” en el directorio raíz del proyecto. El archivo de texto se puede copiar desde su propia web [14].

En caso de ser necesario, se incluirá un fichero llamado “NOTICE” en el mismo directorio que la licencia que contenga información adicional.

Por último, el texto a continuación será incluido en todos los ficheros fuente. Se tiene que añadir como comentario en el comienzo del fichero, sustituyendo el texto entre corchetes por el nombre del autor y por la fecha que correspondan.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Licencia para la documentación

Para el licenciamiento de la documentación se opta por otro tipo de licencia más adecuada al tipo de obra. La licencia escogida es Creative Commons Atribución 4.0 Internacional (CC BY 4.0) [15].

Esta licencia permite que la obra sea compartida libremente. Puede ser copiada y redistribuida en cualquier medio o formato. También se puede reeditar, transformar y crear obras derivadas a partir de la obra original, incluso comercialmente. Solo se requiere la atribución de la autoría original.

Permisos	Condiciones	Limitaciones
Distribución	Crédito al autor	Responsabilidad
Modificación	Aviso de licencia	Uso de patentes
Uso privado	y derechos de autor	Uso de marcas registradas
Uso comercial	Declaración de los cambios	Garantía

Tabla A.8: Licencia CC BY 4.0

El uso de la licencia se indica con la siguiente imagen y texto:



Este obra está bajo una [licencia de Creative Commons Reconocimiento 4.0 Internacional](#).

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apéndice de especificación de requisitos del *software* se proporciona una descripción completa del propósito y funcionalidad del *software*. Se detallan las funciones que debe realizar el *software* y se muestran casos de uso de como los usuarios utilizarán el *software*.

También sirve para definir como tiene que interactuar el *software* con el *hardware* y con otros *softwares*. Además, sirve para definir otros requisitos de carácter no funcional.

Para realizar la especificación de requisitos del *software* The Institute of Electrical and Electronics Engineers (IEEE) ha publicado varios estándares al respecto, IEEE 830-1998 [16] y IEEE 29148-2011 [17]¹. La especificación de requisitos del *software* se realizará siguiendo algunas de las indicaciones de estos estándares.

Identificación

Este apéndice sirve para especificar los requisitos del *software* realizado en el proyecto. Es decir:

- El *software* utilizado por un sistema empotrado para realizar transmisiones TCP/IP.
- El *software* utilizado por la aplicación web para realizar transmisiones TCP/IP con destino al sistema.

¹Actualizado recientemente a IEEE 29148-2018 [18].

Está previsto que ambos *softwares* se publiquen con la versión 1.0 y en este documento se describe su funcionalidad.

Audiencia destinataria

Como audiencia destinataria de este documento se incluye cualquier persona interesada en el proyecto. Tanto tutor, evaluadores, usuarios, como futuros desarrolladores interesados en conocer la funcionalidad del *software* del sistema empotrado y del *software* de la aplicación web.

Abreviaciones

Sistema empotrado: SE

Software del sistema empotrado: SW del SE

Aplicación web: AW

Software de la aplicación web: SW de la AW

Requisito de interfaces externas: RE

Requisito funcional: RF

Requisito no funcional: RNF

Caso de uso: CU

B.2. Objetivos generales

El proyecto cuenta con los siguientes objetivos en relación al funcionamiento del SE y de la AW.

- Configurar un SE que sea capaz de conectarse en red usando TCP/IP.
- El SW del SE puede realizar varias acciones sobre el *hardware*.
- El SW de la AW puede comunicarse con el SE enviando comandos via TCP/IP.
- El SW de la AW puede ordenar la realización de una acción sobre el *hardware*.

Clases de usuario y características

Cualquier persona interesada en SE puede ser usuaria del SE y de la AW. Además, no se requieren conocimientos avanzados ni experiencia técnica para su uso. Bastará con conectar el SE y acceder a la AW.

Entorno operativo

Por un lado, el SE requiere de un servidor DHCP para obtener una dirección IP que le permita comunicarse con el resto de los dispositivos de la red.

Por el otro, para acceder a la AW solo se requiere de un navegador sin importar si es un equipo de sobremesa o en un dispositivo móvil. Sin embargo, si el usuario desea ejecutar la AW por si mismo, se requiere de un sistema que cuente con un servidor de aplicaciones como GlassFish.

Documentación del usuario

Junto con el *software* se proporciona la memoria que describe el desarrollo y sus anexos con información complementaria. A destacar, el apéndice con el manual de usuario que cuenta con indicaciones para el uso del *software*.

B.3. Catalogo de requisitos

Requisitos de interfaces externas

Interfaces de usuario

- **RE-1 Acceso web** El usuario debe poder interactuar con el SE mediante una interfaz web.

Interfaces de *hardware*

- **RE-2 Acceso a la red** El SE tiene que ser capaz de usar Ethernet. RE de alta prioridad.
- **RE-3 Transmisiones en red** El SE tiene que ser capaz de usar TCP/IP. RE de alta prioridad.
- **RE-4 Uso de LED** El SE tiene que ser capaz de usar LED. RE de alta prioridad.

- **RE-5 Uso de I²C** El SE tiene que ser capaz de usar I²C. RE de alta prioridad.
- **RE-3 Uso de PWM** El SE tiene que ser capaz de usar PWM. RE de alta prioridad.

Requisitos funcionales del SW del SE

- **RF-1 Recepción de comandos** El SE tiene que recibir comandos transmitidos mediante paquetes TCP con destino a su correspondiente dirección IP y puerto TCP abierto. RF de alta prioridad.
- **RF-2 Identificación de comandos** El SE tiene que ser capaz de identificar los comandos recibidos para poder ejecutar las acciones correctas. RF de alta prioridad.
- **RF-3 Acción sobre los LED RGB** El SE tiene que poder variar los colores producidos por los LED RGB. RF de alta prioridad.
- **RF-4 Acción sobre el LCD** El SE tiene que poder mostrar cadenas de caracteres en el LCD. RF de alta prioridad.
- **RF-5 Acción sobre los LED PWM** El SE tiene que poder regular la intensidad del brillo de los LED mediante PWM. RF de alta prioridad.

Requisitos funcionales del SW de la AW

- **RF-6 Selección del SE** El usuario debe poder seleccionar el SE con el que desea interactuar. La selección se realiza introduciendo la dirección IP y el puerto del SE escogido. RF de alta prioridad.
- **RF-7 Envío de comandos** La aplicación tiene que ser capaz de construir comandos inteligibles por el SE y que realicen las operaciones indicadas por el usuario. RF de alta prioridad.
- **RF-8 Selección del color** El usuario debe poder escoger un color a ser exhibido por los LED RGB. RF de alta prioridad.
- **RF-9 Introducción de mensajes** El usuario debe poder introducir las cadenas de caracteres a mostrar por el LCD. RF de alta prioridad.
- **RF-10 Regulación del brillo** El usuario debe poder regular la intensidad del brillo de los LED PWM. RF de alta prioridad.

Requisitos no funcionales del SE y de la AW

- **RNF-1 hardware del SE** El SE tiene que ser desarrollado con una placa de desarrollo FRDM-K64F. RNF de alta prioridad.
- **RNF-2 Rendimiento del SE** El SE tiene que ser capaz de realizar las acciones indicadas por el usuario con prontitud. RNF de media prioridad.
- **RNF-3 Seguridad del SE** El SE tiene que asegurar que sus componentes no presentan riesgos eléctricos al usuario. RNF de alta prioridad.
- **RNF-4 Calidad del SW** El SW tiene que garantizar cierto nivel de calidad, p. ej., incluyendo comentarios que faciliten su comprensión para un mantenimiento o portabilidad posteriores. RNF de media prioridad.
- **RNF-5 Experiencia de usuario** La AW debe poder adaptarse a los diferentes dispositivos desde los que se pueda acceder. RNF de media prioridad.

B.4. Especificación de requisitos

Diagrama de Casos de uso

Con el actor se representa a cualquier usuario de la AW. La aplicación se encargará de transmitir los comandos a través de una red usando TCP/IP. Los comandos los procesará el SE para realizar la acción oportuna.

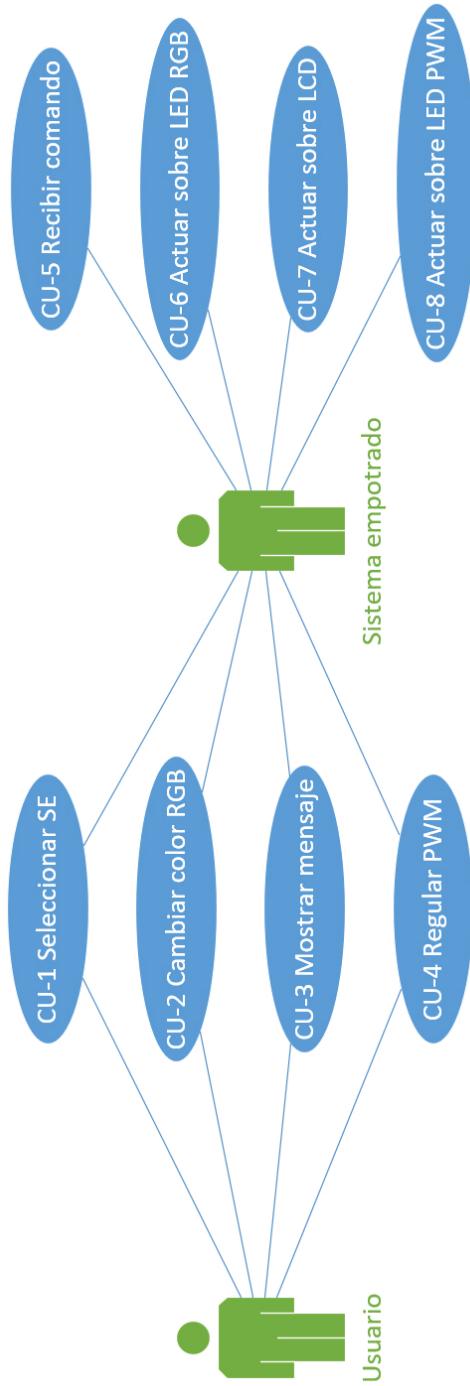


Figura B.1: Diagrama de Casos de uso

Casos de uso

CU-1	Seleccionar SE
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-6
Descripción	El usuario introduce en la AW la dirección IP y el puerto TCP a la escucha del SE.
Precondición	El usuario debe consultar datos en la pantalla LCD del SE.
Acciones	<ol style="list-style-type: none"> 1. El usuario introduce la dirección IP en un cuadro de texto. 2. El usuario introduce el puerto TCP en un cuadro de texto. 3. El usuario pulsa un botón para confirmar los parámetros.
Postcondición	La AW queda configurada para enviar comandos al SE. <ul style="list-style-type: none"> ■ Si la dirección IP es errónea, la configuración no se realiza.
Excepciones	<ul style="list-style-type: none"> ■ Si el puerto TCP es erróneo, la configuración no se realiza.
Importancia	Alta
Comentarios	Ninguno

Tabla B.1: CU-1 Seleccionar SE

Cu-2	Cambiar color RGB
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-7 RF-8
Descripción	El usuario selecciona un botón con el mismo color que desea iluminar con los LED RGB.
Precondición	El usuario ha seleccionado el SE.
Acciones	<ol style="list-style-type: none"> 1. El usuario pulsa uno de los botones de colores disponibles. 2. La AW envía un comando al SE indicando la operación.
Postcondición	El SE ilumina el color indicado.
Excepciones	<ul style="list-style-type: none"> ■ Si no se ha establecido el SE, no se realiza ninguna acción.
Importancia	Alta
Comentarios	Ninguno

Tabla B.2: CU-2 Cambiar color RGB

Cu-3	Mostrar mensaje
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-7 RF-9
Descripción	El usuario muestra un mensaje en una de las líneas del LCD.
Precondición	El usuario ha seleccionado el SE.
Acciones	<ol style="list-style-type: none"> 1. El usuario escoge un cuadro de texto correspondiente a la línea del LCD deseada. 2. El usuario escribe en ese cuadro la cadena de caracteres a mostrar. 3. El usuario pulsa un botón para confirmar el envío de la cadena. 4. La AW envía un comando al SE indicando la operación.
Postcondición	El SE muestra la cadena enviada.
Excepciones	<ul style="list-style-type: none"> ■ Si no se ha establecido el SE, no se realiza ninguna acción.
Importancia	Alta
Comentarios	Ninguno

Tabla B.3: CU-3 Mostrar mensaje

Cu-4	Regular PWM
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-7 RF-10
Descripción	El usuario usa un control deslizante para indicar la intensidad del brillo.
Precondición	El usuario ha seleccionado el SE.
Acciones	<ol style="list-style-type: none"> 1. El usuario utiliza el control deslizante del mismo color el LED a regular. 2. La AW envía un comando al SE indicando la operación.
Postcondición	El SE regula la intensidad LED indicado.
Excepciones	<ul style="list-style-type: none"> ■ Si no se ha establecido el SE, no se realiza ninguna acción.
Importancia	Alta
Comentarios	Ninguno

Tabla B.4: CU-4 Regular PWM

Cu-5	Recibir comando
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-1 RF-2
Descripción	El SE recibe un comando e interpreta su contenido.
Precondición	La AW ha enviado un comando al SE.
Acciones	<ol style="list-style-type: none"> 1. El SE analiza el contenido de un paquete TCP y analiza el comando. 2. El SE ordena realizar la acción solicitada.
Postcondición	El SE debe quedar a la espera de más comandos.
Excepciones	<ul style="list-style-type: none"> ■ Si el comando no es válido, no se realiza ninguna acción.
Importancia	Alta
Comentarios	Ninguno

Tabla B.5: CU-5 Recibir comando

Cu-6	Actuar sobre LED RGB
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-1 RF-2 RF-3
Descripción	El SE modifica los LED RGB.
Precondición	El SE ha recibido un comando que ordena encender o apagar un color determinado.
Acciones	<ol style="list-style-type: none"> 1. El SE identifica el color que se desea encender. 2. El SE enciende o apaga los LED RGB necesarios para mostrar el color.
Postcondición	El SE debe quedar a la espera de más comandos.
Excepciones	<ul style="list-style-type: none"> ■ Si el color no es válido, no se realiza ninguna acción.
Importancia	Alta
Comentarios	Ninguno

Tabla B.6: CU-6 Actuar sobre LED RGB

Cu-7	Actuar sobre LCD
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-1 RF-2 RF-4
Descripción	El SE modifica el texto mostrado por el LCD.
Precondición	El SE ha recibido un comando que ordena mostrar una cadena de caracteres en el LCD.
Acciones	<ol style="list-style-type: none"> 1. El SE identifica la línea del LCD donde escribir el mensaje. 2. El SE escribe la cadena de caracteres en dicha línea del LCD.
Postcondición	El SE debe quedar a la espera de más comandos.
Excepciones	<ul style="list-style-type: none"> ■ Si la línea no es válida, no se realiza ninguna acción.
Importancia	Alta
Comentarios	Ninguno

Tabla B.7: CU-7 Actuar sobre LCD

Cu-8	Actuar sobre LED PWM
Versión	1.0
Fecha	2018-11
Requisitos asociados	RF-1 RF-2 RF-5
Descripción	El SE regula la intensidad del brillo de uno de los LED PWM.
Precondición	El SE ha recibido un comando que ordena regular la intensidad de los LED PWM.
Acciones	<ol style="list-style-type: none"> 1. El SE identifica el LED PWM que hay que regular. 2. El SE regula mediante PWM la intensidad del brillo del LED indicado.
Postcondición	El SE debe quedar a la espera de más comandos.
Excepciones	<ul style="list-style-type: none"> ■ Si el valor de intensidad no es válido, no se realiza ninguna acción.
Importancia	Alta
Comentarios	La intensidad está expresada en forma de porcentaje con valores desde 0 % hasta 100 %.

Tabla B.8: CU-8 Actuar sobre LED PWM

Apéndice C

Especificación de diseño

C.1. Introducción

Tras recoger las especificaciones de los requisitos del *software* se puede comenzar con la especificación del diseño. Este apéndice detalla las soluciones tomadas respecto al diseño del *software*.

Con el diseño del *software* se trata de dar respuesta a cuestiones relativas a los datos, y su representación; los módulos, y su funcionamiento interno; y como estructurar el *software* arquitectónicamente.

Ámbito del *software*

El *software* a desarrollar debe permitir a un usuario operar con un sistema empotrado de forma remota a través de una aplicación web. Al ser entidades tan diferenciadas, tanto el sistema empotrado como la aplicación web tienen que contar con *software* propio y específico a cada plataforma.

El *software* del sistema empotrado tiene que residir en el propio sistema. Con su ejecución se pretende actuar sobre el *hardware* habilitado a tal fin. En cuanto al *software* de la aplicación web, ejecutado en un servidor de aplicaciones, va a servir como interfaz del sistema al usuario y le va a permitir transmitir la acciones a realizar por el sistema empotrado de forma remota.

C.2. Diseño de datos

Para que la aplicación web pueda ordenar al sistema empotrado actuar de una determinada manera se precisa de la transmisión de unos comandos concretos. En consecuencia, se tiene que definir la estructura de los comandos para que ambos *software* sean capaces de entenderse.

Transferencia de los comandos

Los comandos van a ser transferidos usando el protocolo Transmission Control Protocol (TCP). Como el protocolo está orientado a conexión se encarga de establecerla la comunicación mediante unos mensajes de sincronización (SYN) y reconocimiento (ACK), como se puede ver en C.1.

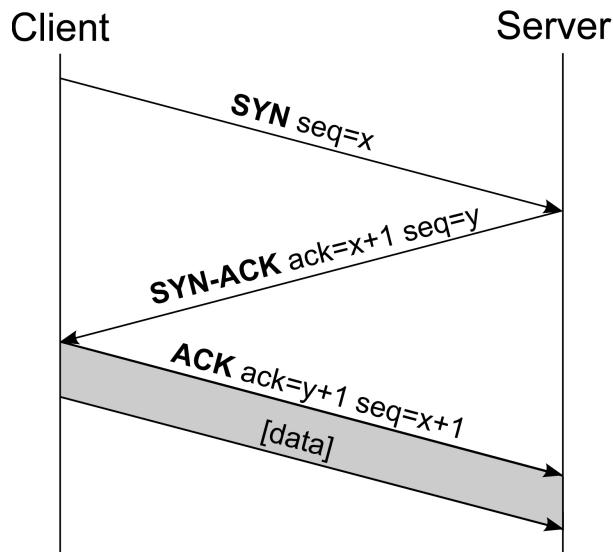


Figura C.1: Conexión TCP [19]

Aprovechando que el protocolo TCP se encarga establecer una conexión sea fiable, se delega en él el establecimiento de la conexión y los comandos serán enviados después como segmentos dentro de un paquete TCP.

Comando LED

Con este comando se indica al sistema empotrado muestre una luz con el color indicado. Individualmente, los LED RGB puede iluminarse cada LED con uno de los tres colores básicos rojo, verde y azul. Como se pueden encender varios LED simultáneamente, combinando los colores básicos se

pueden generar otros nuevos. Los nuevos colores son el amarillo, cyan, magenta, y blanco. Además, debe ser posible que los LED se apaguen y dejen de iluminar.

Para conseguir lo anterior, el nombre del comando refleja la alusión a los LED RGB. El comando se acompaña de un argumento que especifica el color concreto a exhibir. Para ayudar a diferenciar el comando del argumento se ubica un carácter separador.

Color	Comando	Separador	Argumento	Comando resultante
Rojo	led	:	r	led:r
Verde	led	:	g	led:g
Azul	led	:	b	led:b
Amarillo	led	:	y	led:y
Cyan	led	:	c	led:c
Magenta	led	:	m	led:m
Blanco	led	:	w	led:w
Apagados	led	:	o	led:o

Tabla C.1: Comando LED

Comando MSG

Este comando indica al sistema empotrado que muestre una cadena de caracteres en la pantalla LCD. Como la pantalla tiene dos líneas se necesita indicar la línea donde mostrar el texto.

El nombre del comando refleja que trata con los mensajes de texto. El comando se acompaña de un argumento que especifica la línea donde escribir la cadena de caracteres. Después se añade otro argumento con la cadena de caracteres a mostrar. Como en el resto de comandos, entre el comando y cada uno de los argumentos se ubica un carácter separador.

Línea	Comando	S.	Arg. 1	S.	Arg. 2	Comando resultante
1 ^a línea	msg	:	0	:	txt	msg:0:txt
2 ^a línea	msg	:	1	:	txt	msg:1:txt

Tabla C.2: Comando MSG

Comando PWM

Este comando indica al sistema empotrado que regule la intensidad del brillo de los LED compatibles con PWM. El sistema empotrado cuenta con 4 LED regulables por lo que es necesario identificar cual de ellos a regular.

El nombre del comando refleja que van a utilizar los LED PWM. Como hay 4 LED hace falta un argumento que permita seleccionar el LED a regular. Para indicar la intensidad se añade otro argumento con el valor nuevo, entre 0 y 100. De nuevo, entre el comando y cada uno de los argumentos se ubica un carácter separador.

LED	Comando	S.	Arg. 1	S.	Arg. 2	Comando resultante
Blanco	pwm	:	w	:	<0-100>	pwm:w:<0-100>
Verde	pwm	:	g	:	<0-100>	pwm:g:<0-100>
Amarillo	pwm	:	y	:	<0-100>	pwm:y:<0-100>
Rojo	pwm	:	r	:	<0-100>	pwm:r:<0-100>

Tabla C.3: Comando MSG

C.3. Diseño arquitectónico

La arquitectura de un *software* se describe en el estándar IEEE 42010-2011 [20] como los “conceptos fundamentales o propiedades de un sistema en su entorno encarnados por sus elementos, relaciones, y en los principios de su diseño y evolución.”

Así pues, como los entornos del *software* del sistema empotrado y de la aplicación web están tan diferenciados, se opta por usar estilos arquitectónicos diferentes y adaptados a cada *software*.

Diseño arquitectónico del SE

Los sistemas empotrados se relacionan con el entorno en el que se encuentran mediante actuadores y sensores, y dependiendo de su finalidad, con restricciones de tiempo real. La organización del *software* debe ajustarse a estas realidades.

En sistemas sistemas simples sin restricciones de tiempo se puede organizar el *software* de forma simple. En cambio, cuando las tareas de un sistema

empotrado tienen que responder rápidamente a eventos con restricciones de tiempo se hace necesario organizar el *software* de forma más compleja.

En un sistema empotrado sin restricciones se podría utilizar la arquitectura del *software* más simple conocida como Round Robin. Las tareas del sistema empotrado se ejecutan dentro de un bucle principal, en ellas se examina el estado *hardware* y de ser necesario se realiza el tratamiento correspondiente. Una vez terminada una tarea, se ejecuta la siguiente, y al completar la última se vuelve a empezar el ciclo con la primera de todas.



Figura C.2: Tareas en bucle infinito

Para no tener que estar sondeando constantemente el *hardware*, se puede mejorar la arquitectura anterior incluyendo interrupciones. Estas señales permiten indicar al sistema empotrado cuando ha ocurrido un evento y lo libera del sondeo constante.

Si bien ambas arquitecturas serían factibles de aplicar al sistema empotrado, con intención de dar respuesta al requisito no funcional sobre el rendimiento y respuesta del sistema se va a usar una arquitectura basada en un Sistema operativo en tiempo real (RTOS).

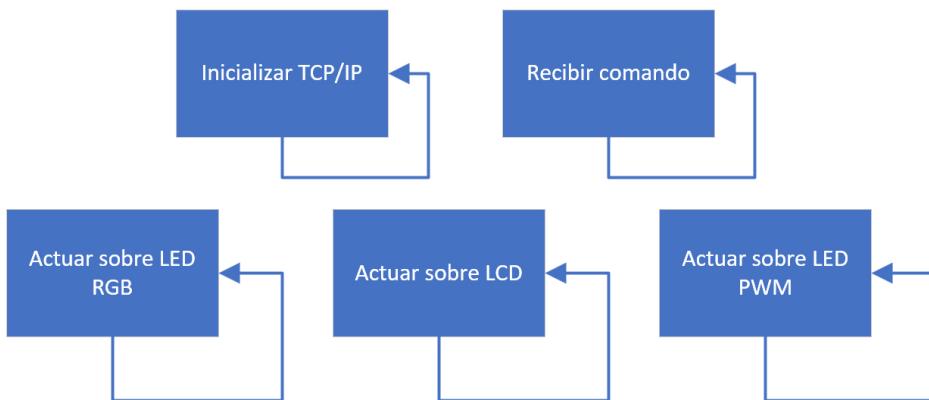
La decisión se fundamenta en la flexibilidad y el tiempo de respuesta que el RTOS es capaz de proporcionar. Cada función del sistema se implementa dentro de una tarea individual con una determinada prioridad. El RTOS se encarga de planificar el orden de ejecución de las tareas en base a la prioridad de cada una de ellas.

Por lo tanto, una tarea con prioridad alta, porque necesita un tiempo de respuesta estricto, es ejecutada antes que las demás. Más aún, se puede configurar el RTOS para que desaloje una tarea de menor prioridad para poder ejecutar una de mayor prioridad.



Figura C.3: Bucle principal del *software*

Con las tareas **C.4** se da respuesta a los casos de usos planteados para el sistema empotrado durante la especificación de los requisitos del *software*. Gracias a la flexibilidad que aporta esta solución se podrían añadir o quitar funciones rápidamente. Por ejemplo, bastaría con agregarlas o quitarlas del RTOS durante la inicialización **C.3**.

Figura C.4: Tareas del *software*

Diseño arquitectónico de la aplicación web

Por lo que se refiere al *software* de aplicación web, representa un cambio de contexto completo. La aplicación se encuentra alojada en un servidor de aplicaciones que permite al usuario acceder a la interfaz web desde un navegador web cualquiera.

La tecnología a utilizar en conjunción con el servidor de aplicaciones es JavaServer Faces (JSF) [21]. Su especificación precisa su propósito para construir aplicaciones web e interfaces de usuarios basadas en componentes. Cada componente de *software* sirve para encapsular un conjunto de funciones o datos estrechamente relacionados.

Además, JSF utiliza el patrón de arquitectura Modelo-Vista-Controlador (MVC). Este patrón separa la lógica de la aplicación en tres partes separadas. Con la separación se impulsa el desarrollo modular, facilitando la colaboración y la reutilización durante el desarrollo. En

Los tres componentes de MVC son:

Modelo: Define los datos y su estructura empleados por la aplicación. Si los datos se modifican, la vista se actualiza para reflejar los cambios.

Vista: Define la interfaz y como se muestran los datos al usuario en ella.

Controlador: Contiene la lógica responsable de manipular el modelo en función de las acciones del usuario.

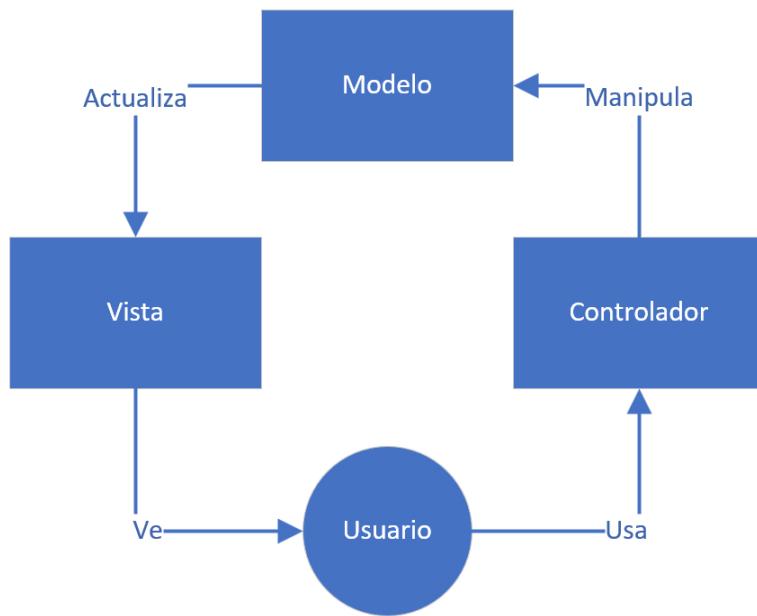


Figura C.5: Relación entre MVC y el usuario

Diseño del modelo

En JSF el modelo se define mediante *beans*. Los *beans* se crean con clases Java. Están compuestos por una conjunto de atributos y sus correspondientes métodos *getters* y *setters*.

Como resultado, se puede consultar el valor de un atributo, también llamado propiedad, mediante su método *get* y se puede actualizar con su método *set*.

Diseño de la vista

En JSF la vista se define mediante ficheros escritos en Extensible HyperText Markup Language (XHTML) que incluyen etiquetas especiales para añadir componentes específicos de JSF. Posteriormente, el código de los ficheros XHTML es traducido a código Hypertext Markup Language (HTML) interpretable por los navegadores web.

La conexión entre la interfaz y el resto de la aplicación se realiza con ayuda del Lenguaje de Expresiones de JSF (EL). Con estas expresiones se enlazan los componentes de la interfaz con las propiedades de los *beans*.

Diseño del controlador

En JSF el controlador se puede definir con *beans* al igual que en el modelo C.3. Pero a diferencia de los *beans* del modelo, los *beans* del controlador cuentan con los métodos necesarios para realizar la lógica de negocio requerida por la aplicación.

Diseño conjunto del sistema

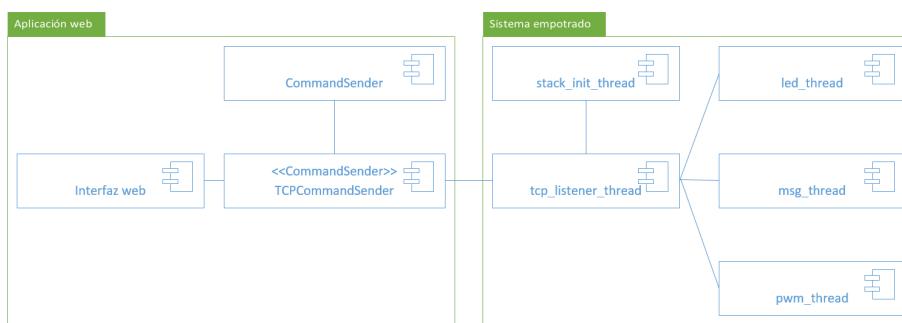


Figura C.6: Diagrama de componentes del sistema

Diseño de la interfaz

Para que el usuario pueda controlar el sistema empotrado la aplicación web pone su disposición una interfaz web. En una sola página el usuario puede ver las funciones disponibles y operar con ellas.

En primer lugar la interfaz tiene que permitir que el usuario introduzca los datos de conexión con el sistema empotrado.

Este formulario es un cuadro rectangular con tres campos horizontalmente dispuestos: 'Dirección IP', 'Puerto' y 'Confirmar'. Los campos 'Dirección IP' y 'Puerto' tienen un efecto de resaltado cuando se seleccionan.

Figura C.7: Introducción de datos

Una vez establecidos los datos se muestran unos paneles con las funciones disponibles.

Con unos botones se puede escoger el color iluminado por los LED RGB.

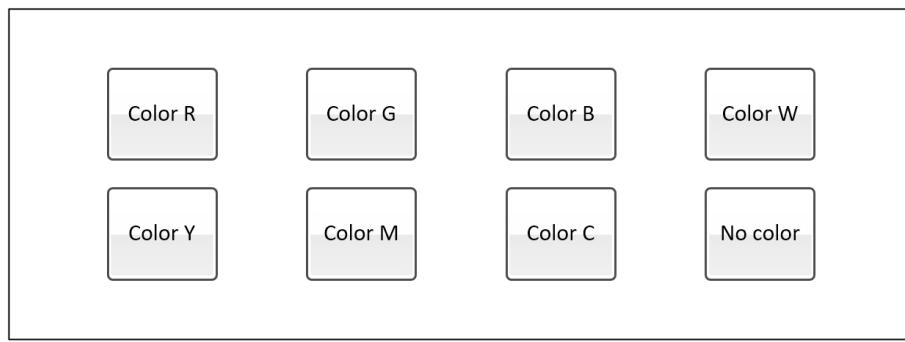


Figura C.8: Panel para cambiar el color de los LED RGB

En unos cuadros de texto, el usuario puede introducir su mensaje y pulsando un botón lo envía.

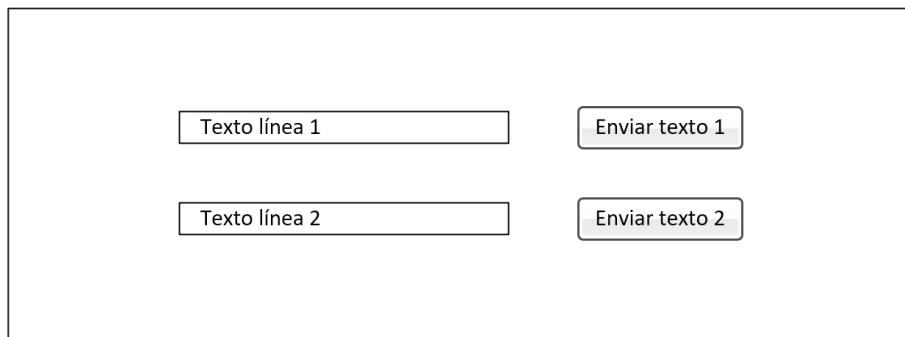


Figura C.9: Panel para cambiar enviar un texto al LCD

Con unos controles deslizantes, el usuario puede cambiar la intensidad del brillo de los LED regulados por PWM.

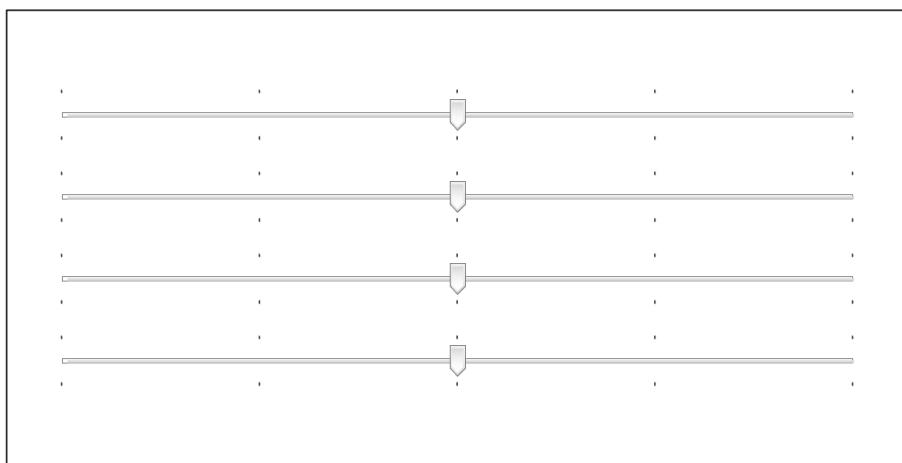


Figura C.10: Panel para regular el brillo de los LED PWM

La interfaz compuesta se construye según lo definido en el siguiente diseño.

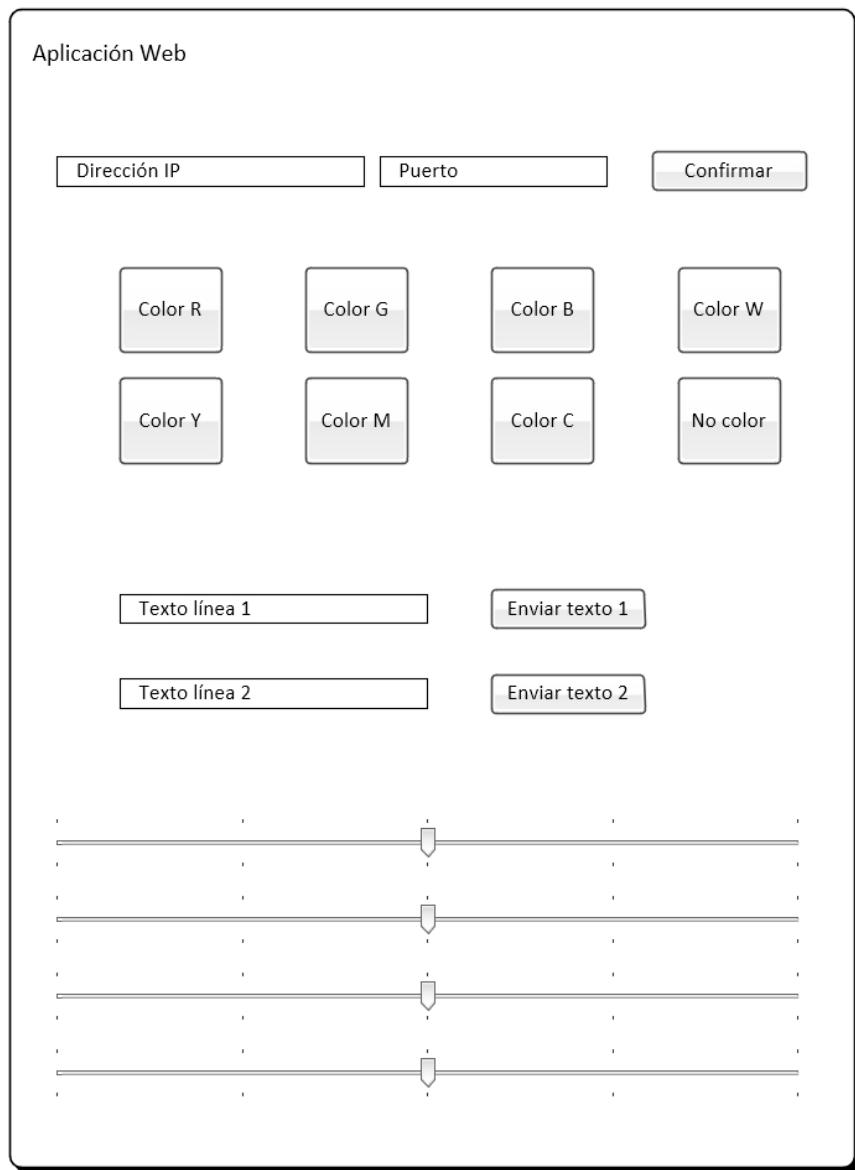


Figura C.11: Diseño de la interfaz

C.4. Diseño procedimental

Apéndice D

Documentación técnica de programación

D.1. Introducción

Para que cualquier persona interesada en conocer, modificar o extender el sistema desarrollado, este apéndice muestra los aspectos a tener en cuenta para lograrlo.

Como los componentes del sistema, el sistema empotrado y la aplicación web, tienen características tan diferentes cada uno necesita herramientas específicas para su desarrollo.

Se detallan las particularidades de cada herramienta, los ajustes de configuración, la estructura que toman los directorios con los ficheros y los pasos necesarios para poner en funcionamiento el proyecto.

D.2. Estructura de directorios

Para empezar a trabajar con el proyecto lo primero que se necesita es una copia del código fuente. La manera más sencilla de consultar y obtener el código es acudir a los repositorios de GitHub. Hay un repositorio dedicado al *software* del sistema empotrado [22] y otro para el *software* de la aplicación web [23].

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

Estructura de directorios del sistema empotrado

El código fuente del *software* del sistema empotrado toma la siguiente estructura:

/ Directorio raíz, contiene el resto de directorios. Incluye el fichero LICENSE que contiene los términos y condiciones del licenciamiento del *software*. Además contiene un fichero de tipo “mex” generado por el entorno de desarrollo (IDE) que sirve para configurar el *hardware* de la placa de desarrollo. Y el fichero Doxyfile que configura la herramienta de documentación.

/CMSIS/ Fuentes pertenecientes al Cortex Microcontroller Software Interface Standard (CMSIS). Proporciona interfaces al procesador y sus periféricos.

/amazon-freertos/ Fuentes pertenecientes a FreeRTOS, el RTOS usado por el sistema empotrado.

/board/ Fuentes autogenerados por el IDE y sus Config Tools que permiten habilitar y configurar el *hardware* de la placa de desarrollo.

/doc/ Contiene la documentación del código y del proyecto.

/doc/amazon-freertos/ Documentación incluida con FreeRTOS.

/doc/lwip/ Documentación incluida con lwIP.

/doc/memoria/ Documentación del proyecto, la memoria descriptiva junto con sus apéndices.

/doc/html Documentación del código fuente generada por Doxygen.

/drivers/ Fuentes con los controladores necesarios para trabajar con el *hardware*.

/lwip/ Fuentes relativos a lwIP, la implementación de la pila de protocolos TCP/IP.

/source/ Código fuente del proyecto. A destacar, el fichero “main.c” encargado del funcionamiento general del sistema. Están presentes los ficheros encargados de tratar con *hardware*. También se encuentran los ficheros de configuración.

/startup/ Código de arranque generado por el IDE.

/utilities/ Código generado por el IDE con utilidades auxiliares usadas para depuración o registro de eventos.

Estructura de directorios de la aplicación web

El código fuente del *software* de la aplicación web toma la siguiente estructura:

/ Directorio raíz, contiene el resto de directorios. Aquí se ubica el fichero “pom.xml” usando por la herramienta Maven para la gestión y construcción del proyecto.

/doc/ Documentación del código fuente generada por Javadoc.

/src/main/ Contiene todo el código perteneciente a la aplicación web.

/src/main/java/ Código Java que implementa la lógica de negocio.

/src/main/webapp/ Contiene el fichero XHTML con el código de la interfaz web.

/src/main/webapp/WEB-INF Contiene los ficheros XML usados por el servidor de aplicaciones.

/src/main/webapp/resources/ Recursos adicionales de la interfaz web.

/src/main/webapp/resources/css/ Contiene el fichero CSS que modifica el aspecto de la interfaz.

/src/main/webapp/resources/images/ Contiene las imágenes y otros recursos gráficos a usar en la interfaz.

D.3. Manual del programador

Para desarrollar el proyecto se han utilizado dos IDE diferentes, uno para cada *software*. Aunque no es indispensable usar las mismas herramientas, a continuación se muestra como obtener, instalar, configurar y utilizarlas de la misma manera que se ha hecho durante el proyecto.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

MCUXpresso IDE

Instalación de MCUXpresso IDE

El IDE se puede obtener gratuitamente desde la zona de Recursos para desarrolladores de NXP [24]. El único requisito establecido por NXP es tener una cuenta registrada, gratuita también, en su sitio web.

Una vez iniciada la sesión y accedido al área de descarga se puede obtener la versión más reciente del IDE. De ser necesario, también es posible descargar una versión antigua desde la pestaña *Previous*.

Tras aceptar los términos y condiciones de uso se muestran los enlaces de descarga de los instaladores del IDE.

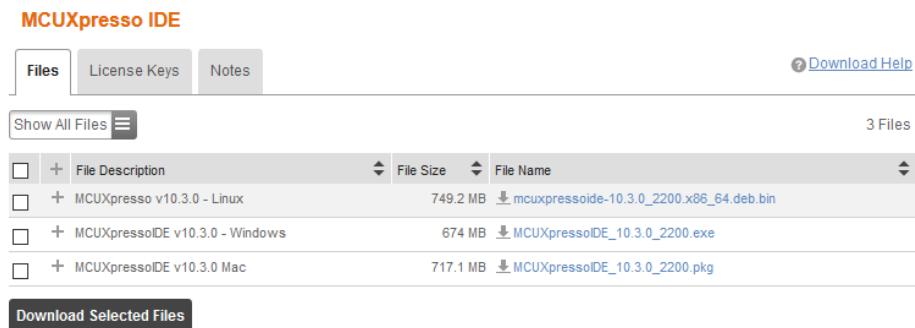


Figura D.1: Descarga de MCUXpresso IDE

El instalador sigue el proceso de instalación habitual a la mayoría de programas. Aceptar licencia de uso, indicar la ubicación del programa e informar que se van a instalar controladores de depuración son los pasos más relevantes.

Concluida la instalación MCUXpresso solicita la ubicación del espacio de trabajo o *workspace* donde situar los proyectos. Cuando MCUXpresso está listo para su uso presenta el aspecto habitual a cualquier otro IDE basado en Eclipse IDE.

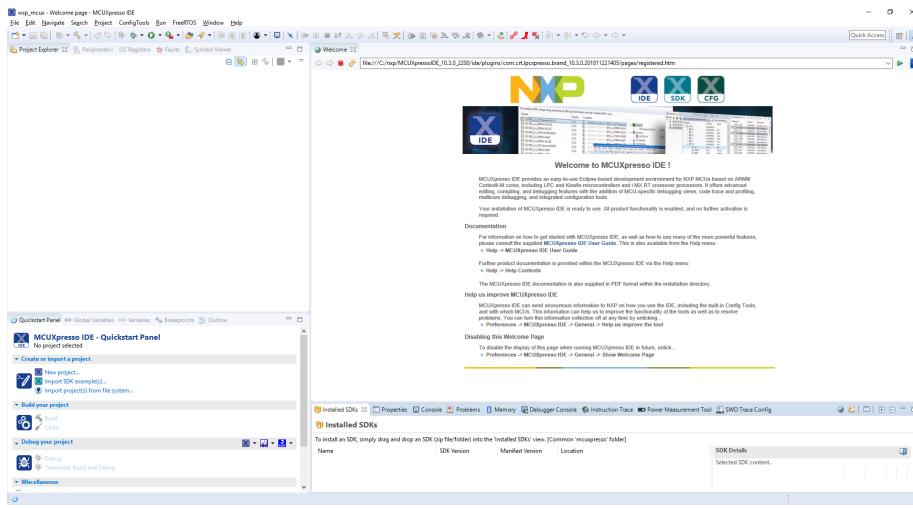


Figura D.2: MCUXpresso recién instalado

Instalación de MCUXpresso SDK

MCUXpresso IDE lleva integrado las MCUXpresso Config Tools, sin embargo, es necesario instalar MCUXpresso SDK, el kit de desarrollo de software (SDK), que incluye los *drivers*, *middleware* y otras utilidades necesarias para el desarrollo.

Las características del SDK están descritas en la web de NXP [25] al igual que las de MCUXpresso IDE. En cambio, la descarga se realiza desde un sitio dedicado a la configuración de los SDK.

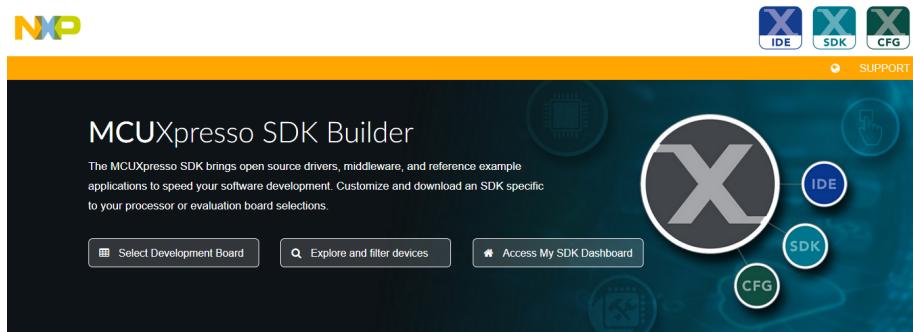


Figura D.3: Sitio de MCUXpresso SDK Builder [26]

El sitio de configuración de SDK permite personalizar, o construir, un SDK específico para la placa de desarrollo utilizada. Además permite seleccionar que componentes incluir o no en el SDK.

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

Para poder descargar el SDK, NXP requiere iniciar sesión con una cuenta de usuario de la misma manera que en la descarga del MCUXpresso IDE.

Dentro del configurador, el primer paso necesario consiste en buscar el *hardware* para el que está destinado el SDK. En este caso, el *hardware* de desarrollo va a ser la placa de desarrollo FRDM-K64F.

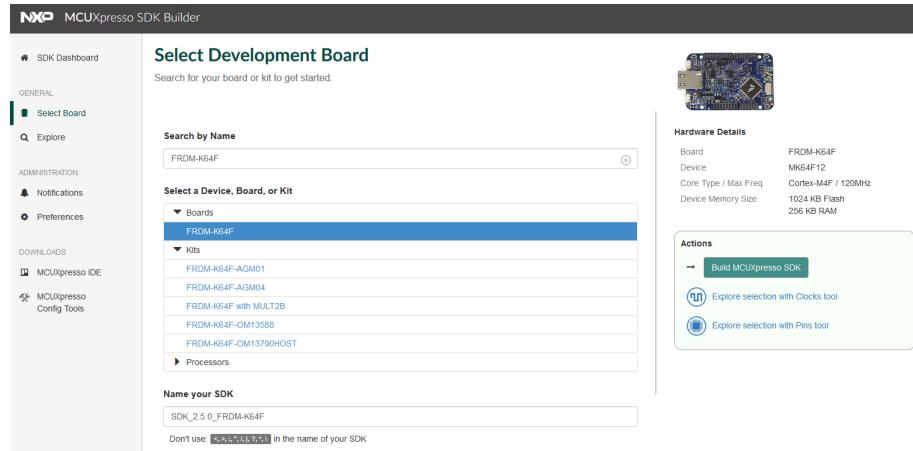


Figura D.4: Selección de la placa de desarrollo [26]

Seleccionada la placa de desarrollo falta configurar el SDK a las necesidades del proyecto. La elección del sistema operativo puede variar entre desarrolladores por lo que queda en su mano la elección. Por otro lado, el IDE va a ser necesariamente MCUXpresso IDE.

Por último, hay que configurar el *middleware* incluido con el SDK. Los componentes mínimos a seleccionar son FreeRTOS y lwIP aunque no supone ningún inconveniente elegir más componentes.



Hardware Details

Board	FRDM-K64F
Device	MK64F12
Core Type / Max Freq	Cortex-M4F / 120MHz
Device Memory Size	1024 KB Flash 256 KB RAM

SDK Details

SDK Version:	2.5.0 (released 2018-12-17)
Host OS:	Linux
Toolchain:	MCUXpresso IDE SDK v2.5.x requires MCUXpresso IDE v10.3.x or later
Middleware:	CMSIS DSP Library, Aml, AWS IoT, Azure IoT, emWin, FatFS, ISSDK, LVHB, lwIP, mbedtls, mcu-boot, MMCAU, multicore, NTAG I2C, sdmmc stack, Secure Element, secure-subsystem, Sigfox, USB stack, wifi_qca, wolfssl, Amazon-FreeRTOS Kernel

Documentation

Base SDK:	MCUXpresso SDK API Reference Manual
Middleware:	ISSDK API Reference Manual

Figura D.5: Resumen del SDK completo [26]

MCUXpresso IDE ofrece una vista con los SDK instalados. Para agregar el SDK recién descargado basta con hacer clic derecho en dicha vista e importar el fichero comprimido con el SDK. Como alternativa, también sirve arrastrar el fichero a la vista para para importar el SDK.

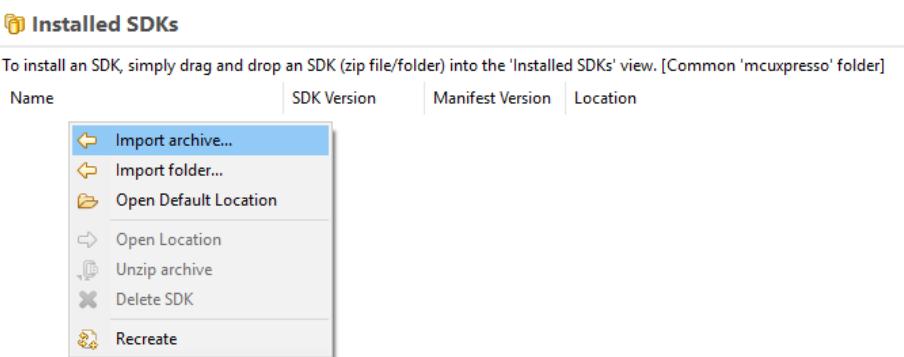


Figura D.6: Importación del SDK

APÉNDICE D. DOCUMENTACIÓN TÉCNICA DE PROGRAMACIÓN

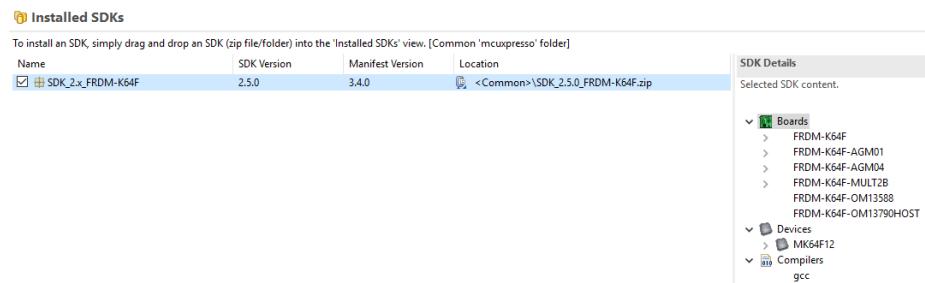


Figura D.7: Detalles del SDK correctamente importado

D.4. COMPILACIÓN, INSTALACIÓN Y EJECUCIÓN DEL PROYECTO

**D.4. Compilación, instalación y ejecución
del proyecto**

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

- E.1. Introducción**
- E.2. Requisitos de usuarios**
- E.3. Instalación**
- E.4. Manual del usuario**

Bibliografía

1. CONSEJERÍA DE HACIENDA, JUNTA DE CASTILLA Y LEÓN. *Decreto Legislativo 1/2013, de 12 de septiembre, por el que se aprueba el texto refundido de las disposiciones legales de la Comunidad de Castilla y León en materia de tributos propios y cedidos* [online] [visitado 2019-02-06]. Disponible desde: <http://bocyl.jcyl.es/boletines/2013/09/18/pdf/B0CYL-D-18092013-1.pdf>.
2. SEGURIDAD SOCIAL. *Bases y tipos de cotización 2019* [online] [visitado 2019-02-06]. Disponible desde: <http://www.seg-social.es/wps/portal/wss/internet/Trabajadores/CotizacionRecaudacionTrabajadores/36537>.
3. JEFACTURA DEL ESTADO. *Ley 27/2014, de 27 de noviembre, del Impuesto sobre Sociedades* [online] [visitado 2019-02-06]. Disponible desde: <https://www.boe.es/buscar/pdf/2014/BOE-A-2014-12328-consolidado.pdf>.
4. MICROSOFT. *Windows 10 Pro* [online] [visitado 2019-02-06]. Disponible desde: <https://www.microsoft.com/es-es/p/windows-10-pro/df77x4d43rkt/48DN>.
5. MICROSOFT. *Office 365* [online] [visitado 2019-02-06]. Disponible desde: <https://products.office.com/es-ES/business/office>.
6. ADOBE. *Creative Cloud para equipos* [online] [visitado 2019-02-06]. Disponible desde: <https://www.adobe.com/es/creativecloud/business/teams.html>.
7. FARRELL COMPONENTS SL. *FRDM-K64F - Placa de Desarrollo* [online] [visitado 2019-02-06]. Disponible desde: <https://es.farnell.com>.

- farnell.com/nxp/frdm-k64f/placa-desarrollo-ethernet-usb-dp/2406741.
8. MK ELECTRÓNICA. *BASIC I/O* [online] [visitado 2019-02-06]. Disponible desde: <https://mkelectronica.com/producto/basic-i-o/>.
 9. FARRELL COMPONENTS SL. *Pantalla LCD Alfanumérica, 16 x 2* [online] [visitado 2019-02-06]. Disponible desde: <https://es.farnell.com/midas/mc21605c6w-bnmlwi-v2/pantalla-alfanum-rica-16x2-blanca/dp/2748649>.
 10. FARRELL COMPONENTS SL. *SOLDERLESS BREADBOARD* [online] [visitado 2019-02-06]. Disponible desde: <https://es.farnell.com/bud-industries/bb-32655/solderless-breadboard-830-tie/dp/2885079>.
 11. FARRELL COMPONENTS SL. *WIRE BUNDLE, BREADBOARD* [online] [visitado 2019-02-06]. Disponible desde: <https://es.farnell.com/adafruit-industries/153/wire-bundle-breadboard/dp/2409349>.
 12. FARRELL COMPONENTS SL. *Cable de Ethernet* [online] [visitado 2019-02-06]. Disponible desde: <https://es.farnell.com/videk/2996-1y/cable-de-conexion-cat6-amarillo/dp/1525699>.
 13. THE APACHE SOFTWARE FOUNDATION. *Applying the Apache License, Version 2.0* [online] [visitado 2019-02-06]. Disponible desde: <https://www.apache.org/dev/apply-license.html>.
 14. THE APACHE SOFTWARE FOUNDATION. *Apache License, Version 2.0* [online] [visitado 2019-02-06]. Disponible desde: <https://www.apache.org/licenses/LICENSE-2.0.txt>.
 15. CREATIVE COMMONS. *CC BY-NC-SA 4.0* [online] [visitado 2019-02-06]. Disponible desde: <https://creativecommons.org/licenses/by-nc-sa/4.0/deed.es>.
 16. INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. *IEEE 830-1998: IEEE Recommended Practice for Software Requirements Specifications* [online]. 1998 [visitado 2019-02-07]. Disponible desde: <https://standards.ieee.org/standard/830-1998.html>.
 17. INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. *IEEE 29148-2011: ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering* [online]. 2011 [visitado 2019-02-07]. Disponible desde: <https://standards.ieee.org/standard/29148-2011.html>.

18. INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. *IEEE 29148-2018: ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering* [online]. 2018 [visitado 2019-02-07]. Disponible desde: <https://standards.ieee.org/standard/29148-2018.html>.
19. WIKIMEDIA COMMONS. *Tcp-handshake* [online]. 2010 [visitado 2019-02-07]. Disponible desde: <https://commons.wikimedia.org/wiki/File:Tcp-handshake.svg>.
20. INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. *IEEE 42010-2011: ISO/IEC/IEEE Systems and software engineering – Architecture description* [online]. 2011 [visitado 2019-02-08]. Disponible desde: <https://commons.wikimedia.org/wiki/File:Tcp-handshake.svg>.
21. ORACLE. *javaserverfaces-spec* [online] [visitado 2019-02-08]. Disponible desde: <https://javaee.github.io/javaserverfaces-spec/>.
22. RPC0027. *k64f-lwip* [online] [visitado 2019-02-09]. Disponible desde: <https://github.com/rpc0027/k64f-lwip>.
23. RPC0027. *web-app* [online] [visitado 2019-02-09]. Disponible desde: <https://github.com/rpc0027/web-app>.
24. SEMICONDUCTORS, NXP. *MCUXpresso Integrated Development Environment (IDE)* [online] [visitado 2019-02-09]. Disponible desde: <https://www.nxp.com/support/developer-resources/software-development-tools/mcuxpresso-software-and-tools/mcuxpresso-integrated-development-environment-ide:MCUXpresso-IDE>.
25. SEMICONDUCTORS, NXP. *MCUXpresso Software Development Kit (SDK)* [online] [visitado 2019-02-09]. Disponible desde: <https://www.nxp.com/support/developer-resources/software-development-tools/mcuxpresso-software-and-tools/mcuxpresso-software-development-kit-sdk:MCUXpresso-SDK>.
26. SEMICONDUCTORS, NXP. *MCUXpresso SDK Builder* [online] [visitado 2019-02-09]. Disponible desde: <https://mcuxpresso.nxp.com/en/welcome>.