



**Grado de Ingeniería Informática
Escuela Superior de Ingeniería de Cádiz**

-Proyectos Informáticos-

“Sistema de comunicación con robot humanoide bioinspirado”

Álvarez Gallego, Laura
Gómez Benítez, Sergio
Jiménez Cordón, Marta
Martos Wegener, Nicolás
Sevillano Hernández, Juan
Tocino Sierra, José Manuel

Agradecimientos

- A Arturo Morgado Estévez, por su implicación y predisposición con el proyecto desde el primer momento.
- Al Grupo de Investigación de Robótica Aplicada, por su gran trabajo para que tuviésemos el hardware disponible en tiempo y forma.
- A Ramón Ruiz Moreno, del Grupo de Investigación de Robótica Aplicada, por su entrega para/con nuestro trabajo.

Índice general

1. Introducción	6
1.1. Motivación	6
1.2. Alcance	6
1.3. Objetivos	7
1.3.1 Objetivos Adicionales:	7
1.3.2 Objetivos específicos:	7
1.4. Estado del arte	7
1.5. Investigación con profesionales del sector	8
1.6. Palabras clave	8
1.7. Organización del documento	9
2. Planificación	10
2.1. Metodología de desarrollo	10
2.2. Planificación del proyecto	10
2.3. Organización	11
2.3.1. Recursos humanos	11
2.3.2. Recursos hardware	12
2.3.3. Recursos software	12
2.3.3.1. Sistemas operativos	12
2.3.3.2. Software de diseño gráfico	12
2.3.3.3. Software para documentación/gestión	13
2.3.3.4. Software para desarrollo	13
2.4. Costes	13
2.5. Riesgos	14
3. Requisitos del sistema	16
3.1. Catálogo de actores	16
3.2. Objetivos del sistema	16
3.3. Catálogo de requisitos	18
3.3.1. Requisitos funcionales	18
3.3.2. Requisitos de información	18
3.3.3. Requisitos no funcionales	18
3.4. Estudio de alternativas tecnológicas	20
4. Diseño del sistema	21
4.1. Diagrama de diseño	21
4.2. Diseño de la arquitectura	21
4.2.1. Arquitectura física	21
4.2.2. Arquitectura lógica	22

4.2.3. Arquitectura de diseño	22
4.3. Diseño de la interfaz de usuario	23
4.4. Diseño de datos	23
4.5. Diseño de componentes	23
4.5.1. Modelo caso de uso	23
4.5.2. Modelos de comportamiento	23
4.6. Parametrización del software base	23
5. Implementación del Sistema	24
5.1 Entorno Tecnológico	24
5.2 Código fuente	25
5.3 Calidad de código	26
6. Pruebas del Sistema	27
6.1. Descripción del Entorno de Pruebas	27
6.2. Pruebas Unitarias	28
6.3. Pruebas de Integración	36
6.4. Pruebas de Sistema	37
6.5. Pruebas de Aceptación	38
7. Conclusiones	38
7.1. Objetivos alcanzados	38
7.2. Lecciones aprendidas	38
7.3. Trabajo futuro	39
Anexos	39
Manual de implantación y explotación	39
B. Manual de usuario	41
C. Documento de Requisitos de Usuario	46
D. Documento de recogida de material	52
E. Reuniones	54
F. Diagrama de Gantt	55
G. Diagramas de diseño	56
H. Código Fuente	61

Índice de tablas

1. Tabla de roles	12
2. Catálogo de actores	16
3. Objetivos del sistema-Iniciar interfaz	16
4. Objetivos del sistema-Movimiento humanoide	17
5. Requisitos no funcionales	18
6. Requisitos no funcionales Usabilidad	18
7. Requisitos no funcionales Rendimiento	19
8. Verificación Mover Mano Derecha	29
9. Verificación Mover Mano Izquierda	29
10. Verificación Mover Codo Derecho	29
11. Verificación Mover Codo Izquierdo	30
12. Verificación Mover Hombro Derecho	30
13. Verificación Mover Hombro Izquierdo	30
14. Verificación Mover Pie Derecho	31
15. Verificación Mover Pie Izquierdo	31
16. Verificación Mover Rodilla Derecha	31
17. Verificación Mover Rodilla Izquierda	31
18. Verificación Mover Cadera Derecha Vertical	32
19. Verificación Mover Cadera Derecha Horizontal	32
20. Verificación Mover Cadera Izquierda Vertical	32
21. Verificación Mover Cadera Izquierda Horizontal	33
22. Verificación Mover Tobillo Derecho	33
23. Verificación Mover Tobillo Izquierdo	33
24. Verificación Botón Reproducir Secuencia de Movimiento	34
25. Verificación Botón Stop Secuencia de Movimiento	34
26. Verificación Botón Reset Posición de los Servos	34
27. Verificación Botón Guardar Secuencia de Movimiento	35
28. Verificación Botón Nosotros	35
29. Verificación Botón Información	35
30. Verificación Completa de Pruebas Unitarias	36
31. Verificación Botón Reproducir Secuencia de Movimientos Pruebas Unitarias	36
32. Verificación Mover Brazo Derecho	37
33. Verificación Mover Brazo Izquierdo	37
34. Verificación Mover Pierna Derecha	37
35. Verificación Mover Pierna Izquierda	38

1. Introducción

Los sistemas bioinspirados son sistemas contruidos por medio de hardware configurables y sistemas electrónicos que emulan la forma de pensar, el modo de procesar información y resolución de problemas de los sistemas biológicos.

La realización de este proyecto consiste en el desarrollo de una interfaz gráfica donde aparezca un humanoide con las articulaciones marcadas, con posibilidad de movimiento mediante una serie de botones. Una vez pulsado esos botones, generará una respuesta en los servos del Robot Humanoide Bioinspirado, realizando un movimiento en esa articulación.

La interfaz permitirá dotar de movimiento al robot humanoide, según se desplace el *scrollbar*, iluminando a su vez el servo en cuestión. Además, se podrán guardar una serie de posiciones, de forma que reproduzca una secuencia de movimientos, pudiendo además, volver a la posición inicial.

La interfaz ha sido realizada con Processing, siendo software libre. Para representar el movimiento en los servos, se ha usado Arduino.

1.1. Motivación

Desde hace mucho tiempo se ha soñado siempre con autómatas que ayudan al hombre en el trabajo pesado o peligroso, incluso sustituirlo o hacer tareas que el hombre no está capacitado para realizar. Autómatas guerreros, exploradores o muy inteligentes capaces de razonar.

La realidad es lejana de la ciencia ficción, algo sencillo e intuitivo como es para el hombre interactuar con el entorno, moverse, explorar, sentir o incluso algo tan obvio como ver, cuando intentamos realizarlo con un ordenador se convierte en una tarea complicada, de alto requerimiento computacional, o nuestro conocimiento no nos lo permite.

La realización de este proyecto nos ha servido para plantearnos el reto de aprender este tipo de programación, tan en auge.

1.2. Alcance

El propósito de este proyecto es producir el movimiento de un servo mediante la interfaz gráfica del programa y haciendo uso de los servos *PFM*.

1.3. Objetivos

Ha sido un proyecto desarrollado para el Grupo de Investigación de Robótica Aplicada, cuyo objetivo general ha sido crear una interfaz gráfica donde aparezca un humanoide con las articulaciones marcadas con posibilidad de movimiento mediante una serie de botones. Una vez pulsado esos botones generará una respuesta en los servos del Robot Humanoide Bioinspirado que realizarán un movimiento en esa articulación.

Los servos a usar para la movilidad del Robot serán del tipo *PFM* (Modulación por Frecuencia de Pulsos). Estos servos, han sido generados por un grupo de alumnos pertenecientes al Grado en Ingeniería en Electrónica Industrial.

En particular, estos servos han sido modificados de la siguiente forma, funcionaban con un sistema de modulación por anchura de pulsos y ahora funcionan con un nuevo sistema de modulación por frecuencia de pulsos.

1.3.1 Objetivos Adicionales:

- Una vez generada la interfaz gráfica sin movimientos, nuestro próximo objetivo será que nuestra interfaz reproduzca los movimientos que deberá realizar a su vez el humanoide conectado al ordenador.
- Crear una librería para poder manejar de forma más fácil el nuevo servo mediante frecuencia, como sugerencia del cliente.

1.3.2 Objetivos específicos:

- Buscar información sobre Arduino.
- Conectar y mover un servo mediante Arduino
- Buscar información UNity
- Buscar información sobre Blender
- Buscar información programación Processing
- Crear funciones para movimiento servo automático
- Servos mediante Pulso
- Crear interfaz inmóvil.
- Pruebas con el robot
- Documentación del proyecto

1.4. Estado del arte

Hoy en día existen diversas aplicaciones que producen y reproducen el movimiento de un robot.

Por lo general, los robots son sistemas electromecánicos que normalmente son conducidos por un programa de una computadora o por un circuito eléctrico. Este sistema electromecánico, por su apariencia o sus movimientos, ofrece la sensación de tener un propósito propio. La independencia creada en sus movimientos hace que sus acciones sean la razón de un estudio razonable y profundo en el área de la ciencia y tecnología. Esta independencia es debida al uso de servomotores en cada una de sus articulaciones.

Estos servomotores, conocidos más comúnmente como servos, son un tipo especial de motor capaz de posicionarse dentro de un intervalo de operación (entre 0° y 180°). Para ello, el servo espera un tren de pulsos que se corresponde con el movimiento a realizar.

Los sistemas de servos más empleados son del tipo *PWM* (Modulación por Anchura de Pulsos). Este sistema consiste en generar una onda cuadrada en la que se vería el tiempo que el pulso está a nivel alto, manteniendo el mismo período. Sin embargo, los sistemas de servos con los que trabaja nuestro robot humanoide bioinspirado, son del tipo *PFM*, y han sido proporcionados por un grupo de compañeros del Grado en Ingeniería en Electrónica Industrial.

Cierto es que, aunque existan aplicaciones que representan movimientos de robots humanoides, ninguna lo hace con el tipo de servos *PFM*, ya que han sido creados por la propia Universidad de Cádiz.

1.5. Investigación con profesionales del sector

En cierta etapa del desarrollo del proyecto, nos vimos en la necesidad de contar con asesoramiento profesional, debido a que nuestro grupo de trabajo tenía carencias formativas sobre robótica.

Desde el primer momento que nos reunimos con Arturo, se ofreció para darnos unas clases sobre Arduino.

El resto de reuniones con nuestro cliente no han sido de carácter formativo, sino informativo. Arturo ha estado en todo momento al tanto del proceso de desarrollo en el que se encontraba su proyecto.

1.6. Palabras clave

Robot, control servo, movimiento, PFM, PWM.

1.7. Organización del documento

En este apartado se ofrece un breve resumen del contenido de cada uno de los capítulos y anexos que contiene este documento.

En el capítulo 1, “Introducción” se intenta dar una visión global del proyecto que se ha desarrollado y se mencionan cuales son los principales objetivos del proyecto, así como la motivación obtenida para realizar el proyecto. Por último se define la estructura de este documento.

En el capítulo 2 se presentan las tecnologías utilizadas, así como los costes necesarios y los riesgos implicados en la realización del proyecto.

Los capítulos 3, 4, 5 y 6 comprenden el análisis, diseño, implementación y pruebas, respectivamente, de la aplicación desarrollada. En ellos podrá encontrar un estudio detallado sobre los casos de uso, requisitos, arquitectura utilizada, y detalles de implementación y pruebas.

Al final de este documento también podrá encontrar una serie de anexos que incluyen: Manual de usuario y las conclusiones del trabajo realizado, así como las líneas futuras a la hora de mejorar la aplicación.

2. Planificación

2.1. Metodología de desarrollo

En primer lugar, destacar que, se creyó conveniente usar la metodología SCRUM, en la cual se divide el proyecto en un conjunto de etapas (iteraciones) que realizan de forma continua todos los integrantes del equipo, completando una serie de requisitos fijados en el diagrama de Gantt (ver punto 2.3).

Una vez que el producto evoluciona en cada iteración podremos fijar nuevos requisitos a partir de los anteriormente completados, e incluso mejorarlos, si procede. Para poder guiar de manera efectiva el proyecto, se han priorizado los objetivos / requisitos en función del valor que aportan al cliente.

Además, se ha optado por usar Trello, que es un gestor de tareas que permite el trabajo de forma colaborativa. De esta forma, quedan marcadas y eliminadas de las tareas pendientes aquellas que se han realizado, lo cual ha servido de ayuda a la hora de desarrollar el proyecto.

2.2. Planificación del proyecto

Para la estimación del tiempo, se ha realizado un diagrama de Gantt, dividiendo el tiempo en seis tareas principales. Este documento se adjunta en anexo.

- **Realización de la documentación:** En esta etapa, se realizarán los siguiente documento: DRS, DAS, el presente documento, la presentación para la versión final. Como este documento se podrá modificar en cualquier momento, esta etapa perdurará desde el principio hasta el final del proyecto.
- **Implementación:** En esta etapa, se realizará la implementación restante para completar la aplicación. Esta etapa durará hasta el 29/05/2017, fecha de la presentación de la versión final.
- **Asesoramiento:** En esta etapa, se dedicará a la búsqueda de asesoramiento externo con la idea de adquirir los conocimientos necesarios para el desarrollo del proyecto.
- **Diseño:** En esta etapa se realizarán los diseños correspondientes para la aplicación (diseño lógico, diseño de la interfaz). Esta etapa durará hasta 15/05/2017.
- **Pruebas:** En esta etapa se realizarán las pruebas de la aplicación. Las pruebas empezarán casi el mismo tiempo que la implementación y durará hasta el día 31/05/2017.

- Revisión final: En esta etapa, se realizará una versión completa de todo el proyecto. Con esto se intentará asegurar la corrección para el día de la entrega final.

2.3. Organización

A continuación indicaremos los recursos materiales necesarios para poder realizar el proyecto en su totalidad, incluyéndose en esta categoría los recursos hardware y software.

2.3.1. Recursos humanos

Rol	Integrante	Funciones y responsabilidades
Manager	Nicolás Martos Wegener	Coordina el proyecto y a los distintos integrantes del grupo. Gestionar el tiempo de cada fase del proyecto y realizar las reuniones entre los miembros del equipo.
Analista	Marta Jiménez Cordón	Es la encargada de hablar con el cliente y transmitir las necesidades del cliente al equipo, además es la encargada de realizar el análisis de requisitos y funcionalidades software y el análisis de requisitos del usuario.
Diseñador	Laura Álvarez Gallego	Planifica, descompone, determina y especifica cómo se irá llevando a cabo el proyecto. Es la encargada del diseño arquitectónico del producto a desarrollar y de los distintos prototipos.
Programador	Sergio Gómez Benítez	Son los encargados de pasar a línea de código lo que anteriormente se ha especificado en el Diseño. Para ello determinan el lenguaje más apropiado para el trabajo que el diseñador les ha solicitado.

Programador	José M. Tocino Sierra	Son los encargados de pasar a línea de código lo que anteriormente se ha especificado en el Diseño. Para ello determinan el lenguaje más apropiado para el trabajo que el diseñador les ha solicitado.
Tester	Juan Sevillano Hernández	Realiza las pruebas necesarias para comprobar el correcto funcionamiento del sistema de comunicación. Su función es de gran importancia ya que de estas pruebas dependerá la calidad del software.

Tabla 1. Roles de cada integrante

2.3.2. Recursos hardware

El hardware necesario será aportado por el cliente;

- Robot Hovis Lite realizado con servos *PFM*.
- Servos *PFM*.
- Placas de Arduino

2.3.3. Recursos software

2.3.3.1. Sistemas operativos

- Windows; para el desarrollo del proyecto.
- Distribución Ubuntu; para realizar las diferentes pruebas.
- Mac OS; Para realizar las diferentes pruebas.

2.3.3.2. Software de diseño gráfico

- Gimp; descomposición de la imagen del robot en partes para representar el movimiento de los servos.

2.3.3.3. Software para documentación/gestión

- GanttProject; desarrollo de diagramas de Gantt.
- Google Drive; para alojar tanto el proyecto como toda la documentación.
- Trello; para gestionar el desarrollo del proyecto.

2.3.3.4. Software para desarrollo

- Processing; lenguaje de programación y entorno con el que se ha desarrollado la interfaz.
- Arduino; lenguaje de programación para microcontroladores con el que se ha desarrollado el movimiento de los servos.

2.4. Costes

En esta categoría se incluyen tanto las personas necesarias para el desarrollo del software como el tiempo de realización del trabajo, y el coste de los recursos hardware y software;

- Será necesario un grupo de al menos 6 empleados, entre ellos necesitaremos un analista, un diseñador, dos programadores, un tester y un manager. El tiempo para realizar el proyecto está estimado en unos 3 meses desde su inicio, el día 8 de Marzo.

Según las tablas salariales de Ingenieros recogidos en el BOLETÍN OFICIAL DEL ESTADO del 18 de Enero de 2017, se estima que el coste por hora de cada uno de nuestros integrantes será de 50€. Por lo tanto, si se supone que cada uno de ellos ha dedicado una hora al día al proyecto, sumaría un total de 22 horas trabajadas al mes, lo que haría un total de 1100€ al mes por cada uno de ellos.

El tiempo de desarrollo del proyecto ha sido de tres meses, con lo cual, el precio de cada miembro del equipo de desarrollo se estipularía en 3300€ .

Teniendo en cuenta que el equipo está formado por 6 personas, el coste total del equipo humano sería de 19800€.

- Para el desarrollo del robot y la creación de su interfaz se necesita una serie de materiales hardware, suministrados por el cliente. Estos productos estarían constituidos por 6 Kits Arduino Leonardo con un precio de 34.95€ cada uno y con un “Robot Hovis Lite” con un precio de 799€. En total el coste de recursos hardware ascendería a 1008.70€.
- Usaremos software de licencia gratuita, por lo tanto el coste de los recursos software ascenderá a 0 €.

- El coste total de desarrollo del proyecto ascendería a 20808,70€.

2.5. Riesgos

Los riesgos del proyecto que han sido recogidos y clasificados siguiendo, en cierta medida, la metodología descrita en [Pressman, 2002], son los siguientes:

Riesgo 1: No disponibilidad del Hardware

- **Descripción:** Que el robot no esté construido a tiempo.
- **Probabilidad:** 80%
- **Impacto:** No se verá el resultado sobre el robot humanoide, sino sobre los servos.
- **Plan de contingencia:** Para evitarlo, el equipo se reunirá de forma periódica con el cliente, siendo conscientes de que es un riesgo que no depende del propio equipo de desarrollo.
- **Plan de actuación:** Se realizarán las pruebas sobre distintos servos sin que estén conectados entre sí.

Riesgo 2: Cambio en la tecnología

- **Descripción:** No disponer de los servo *PFM* a tiempo.
- **Probabilidad:** 50%
- **Impacto:** Incumplimiento del objetivo general.
- **Plan de contingencia:** Para evitarlo, el equipo se reunirá de forma periódica con el cliente, siendo conscientes de que es un riesgo que no depende de nuestro equipo de desarrollo.
- **Plan de actuación:** Según el acuerdo firmado con el cliente, se modificarán los requisitos para hacer tan sólo la interfaz del Sistema de Comunicación. Para ello se cambiará el entorno de desarrollo por uno que nos facilite la representación en la propia interfaz del movimiento del robot.

Riesgo 3: Cambio en los requisitos

- **Descripción:** El cliente solicita una serie de funcionalidades no recogidas en el documento de requisitos de usuario.
- **Probabilidad:** 80%.
- **Impacto:** La entrega de la totalidad del proyecto con los nuevos requisitos, se verá pospuesta.
- **Plan de contingencia:** El cliente ha sido informado de las funcionalidades que tendrá el proyecto en una primera entrega, y ha dado su beneplácito para recibir estos posibles cambios con posterioridad.
- **Plan de actuación:** Previendo posibles cambios en los requisitos del proyecto, se ha implementado haciendo uso de módulos que nos faciliten su futura modificación.

Riesgo 4: Retraso en la especificación

Dentro de este, encontramos dos riesgos asociados, siendo el segundo una consecuencia del primero.

4.1 Falta de número de placas de arduino o fallo en alguna de ellas

- **Probabilidad:** 10%
- **Impacto:** El aprendizaje en arduino será más lento.
- **Plan de contingencia:** Para evitarlo, se usarán simuladores de Arduino, de modo que podamos ir avanzando en nuestro aprendizaje.
- **Plan de actuación:** Se tendrán que hacer por grupos para que todos los integrantes del equipo de desarrollo tengan posibilidades de usarlo.

4.2 Aprendizaje de Arduino más lento del esperado

- **Probabilidad:** 10%
- **Impacto:** No podremos cumplir alguno de los objetivos adicionales.
- **Plan de contingencia:** Para evitarlo se investigará sobre dicha materia.
- **Plan de actuación:** Se pedirá ayuda al propio cliente, experto en el tema, en caso de ser necesario.

Riesgo 5: Sobreestimar el tamaño del proyecto

- **Descripción:** Llevar a cabo una mala planificación que afecte a los tiempos de desarrollo del proyecto.
- **Probabilidad:** 15%
- **Impacto:** La entrega del proyecto se verá retrasada.
- **Plan de contingencia:** Para evitarlo, el equipo se reunirá semanalmente para informar sobre el estado de desarrollo del proyecto.
- **Plan de actuación:** El equipo verá incrementada las horas dedicadas a este proyecto, para que la entrega sea retrasada el menor tiempo posible.

3. Requisitos del sistema

3.1. Catálogo de actores

ACT-01:	Usuario
Versión:	1.0
Autores:	Laura Álvarez Gallego Sergio Gómez Benítez Marta Jiménez Cordón Nicolás Martos Wegener Juan Sevillano Hernández José Manuel Tocino Sierra
Fuentes:	Ninguna
Descripción:	Se trata principalmente de la persona que va a controlar el humanoide
Comentarios:	Ninguno

Tabla 2. Catálogo de actores

3.2. Objetivos del sistema

OBJ-001	Iniciar Interfaz
Versión	1.0 (24/04/17)
Autores	Laura Álvarez Gallego Sergio Gómez Benítez Marta Jiménez Cordón Nicolás Martos Wegener Juan Sevillano Hernández José Manuel Tocino Sierra
Fuentes	Ninguna
Descripción	El sistema debe ser capaz de mostrar al

	usuario la interfaz del humanoide
Subobjetivos	OBJ-002,
Importancia	Poco importante
Urgencia	Media
Estado Estabilidad	Terminado Media
Comentario	Ninguno

Tabla 3. Objetivos del sistema-Iniciar interfaz

OBJ-002	Movimiento humanoide
Versión	1.0 (24/04/17)
Autores	Laura Álvarez Gallego Sergio Gómez Benítez Marta Jiménez Cordón Nicolás Martos Wegener Juan Sevillano Hernández José Manuel Tocino Sierra
Fuentes	Ninguna
Descripción	El sistema debe ser capaz de representar los movimientos generados por la interfaz
Subobjetivos	Ninguno
Importancia	Muy alta
Urgencia	Inmediata
Estado	Terminado
Estabilidad	Media
Comentario	Ninguno

Tabla 4. Roles de cada integrante-Movimiento humanoide

3.3. Catálogo de requisitos

3.3.1. Requisitos funcionales

Los requisitos funcionales han sido descritos mediante la especificación de casos de uso en el capítulo de Análisis del sistema que se expone a continuación.

3.3.2. Requisitos de información

Los datos con los que se trabajan, son proporcionados por el robot humanoide, siendo el único documento de texto usado el fichero de configuración utilizado para indicar el puerto serial al que debemos acceder.

La información que se utiliza se corresponde al puerto serial por donde se envía la información al robot para que realice el movimiento correspondiente. Esta información estaría formada por el identificador del servo que se desea mover y el grado al que se desea posicionar.

3.3.3. Requisitos no funcionales

NFR-001	Extensibilidad
Versión	1.0 (25/04/17)
Autores	Laura Álvarez Gallego Sergio Gómez Benítez Marta Jiménez Cordón Nicolás Martos Wegener Juan Sevillano Hernández José Manuel Tocino Sierra
Fuentes	Ninguna
Dependencias	Ninguna
Descripción	El sistema deberá permitir ampliaciones futuras de funcionalidades
Importancia	Muy importante
Urgencia	Inmediatamente
Estado	En construcción

Estabilidad	Media
Comentario	Ninguno

Tabla 5. Requisitos no funcionales Extensibilidad

NFR-002	Usabilidad
Versión	1.0 (25/04/17)
Autores	Laura Álvarez Gallego Sergio Gómez Benítez Marta Jiménez Cordón Nicolás Martos Wegener Juan Sevillano Hernández José Manuel Tocino Sierra
Fuentes	Ninguna
Dependencias	Ninguna
Descripción	El tiempo de aprendizaje del sistema por un usuario deberá ser menor de 4 horas.
Importancia	Muy importante
Urgencia	Inmediatamente
Estado	En construcción
Estabilidad	Media
Comentario	Ninguno

Tabla 6. Requisitos no funcionales Usabilidad

NFR-003	Rendimiento
Versión	1.0 (25/04/17)
Autores	Laura Álvarez Gallego Sergio Gómez Benítez Marta Jiménez Cordón Nicolás Martos Wegener Juan Sevillano Hernández José Manuel Tocino Sierra
Fuentes	Ninguna

Dependencias	Ninguna
Descripción	El sistema deberá contar con precisión en el tiempo de respuesta.
Importancia	Muy importante
Urgencia	Inmediatamente
Estado	En construcción
Estabilidad	Media
Comentario	Ninguno

Tabla 7. Requisitos no funcionales Rendimiento

3.4. Estudio de alternativas tecnológicas

- **Unity**

Se trata de un motor de desarrollo para la creación de juegos y contenidos 3D interactivos, con las características de que es completamente integrado y que ofrece innumerables funcionalidades para facilitar el desarrollo de videojuegos, aunque su aprendizaje es mucho más complejo que el software utilizado para la realización del proyecto.

Finalmente el equipo se decantó por Processing ya que está basado en Java y además, es muy similar al IDE de Arduino, con el que hay que conectar la interfaz para poder mover el robot.

4. Diseño del sistema

En este capítulo se van a distinguir cinco apartados diferenciados. En primer lugar se explica de forma detallada la arquitectura del sistema, el modelo seguido para su desarrollo y los componentes que lo forman. Posteriormente, a más bajo nivel, se detallan todas las clases y métodos que forman las diferentes partes de la aplicación así como el diseño de todas las interfaces con las que el usuario puede interactuar. Por último, se presentarán algunos diagramas de secuencia que detallarán algunas acciones relevantes a realizar en la aplicación, y se detallarán las modificaciones que ha sufrido el software base.

4.1. Diagrama de diseño

El diagrama representa la generalización de pantallas de la aplicación, junto los componentes asociados a cada una de ellas, tales como los botones y otros elementos, necesarios para asegurar el correcto funcionamiento de dicha aplicación. Se adjunta en el Anexo G.1.

4.2. Diseño de la arquitectura

A continuación se explicará detalladamente la arquitectura del sistema a desarrollar. Se establecerá qué tipo de aplicación se ha desarrollado, el modelo que sigue y los componentes que la forman.

4.2.1. Arquitectura física

Como componente hardware vamos a utilizar PCs para el desarrollo de la aplicación así como de la documentación y las pruebas.

Además, se hará uso de placas de Arduino, servos PFM y un robot creado por el Grupo de Investigación de Robótica Aplicada de la Escuela Superior de Ingeniería, con estos mismos servos.

4.2.2. Arquitectura lógica

Para realizar la interfaz desde la que controlar los movimientos del robot humanoide, primero hubo que estudiar la manera de comportarse de estos. Están formados por una serie de servos conectados a una placa principal, desde la que se mandan los movimientos que realizará cada uno.

Para poder implementar los movimientos, se han usado Processing y Arduino. Con Processing se ha desarrollado la interfaz, conectándola con Arduino por medio de los puertos seriales. En Arduino habrá que cargar el programa encargado de hacer que los servos tengan movimiento.

4.2.3. Arquitectura de diseño

El patrón arquitectónico del sistema se divide en diferentes capas, entre las cuales se encuentran:

- Capa de presentación: Compuesta por las diferentes vistas o pantallas de la aplicación, que a su vez pueden incluir o no botones de selección con los que interactuará el usuario. Esta capa consta de las siguientes características;

- Permite la generación de movimiento de los diferentes servos a través de los scrollbars.
- Almacena movimientos de servos para luego reproducir secuencias.
- Tratamiento de botones y elementos.

- Capa de dominio o negocio: Incluirá toda la lógica de la aplicación, de la recepción de los movimientos, cambios de pantalla o acciones determinadas al pulsar los botones. Consta de las siguientes características;

- Control de aserciones y validaciones de los eventos
- Cálculo de la frecuencia para el movimiento de los servos a través de los grados
- Control de la respuesta ante eventos
- Ejecución de acciones
- Envío de respuestas al usuario (capa de presentación)

- Capa de gestión de ficheros o integración: En esta capa se sabe dónde y cómo están almacenados los datos, y cómo tratarlos. Es decir, aquí es donde, una vez que se realizan invocaciones a los métodos (realizadas en la capa de negocio), se genera el fichero correspondiente para proceder al correcto envío de información a través del puerto.

- Permite que la capa de negocio pueda ignorar dónde están los datos.

Se adjunta el Anexo G.2, donde se especifica un esquema del patrón arquitectónico de la aplicación.

4.3. Diseño de la interfaz de usuario

En el Anexo G.3 se incluye un diagrama de navegación, con el que se pretende reflejar la secuencia de pantallas a las que tiene acceso el usuario y la conexión que hay entre ellas.

4.4. Diseño de datos

Tal y como se explicó en el apartado anterior, los datos con los que se trabajan son proporcionados por el robot humanoide, siendo el único documento de texto usado el fichero de configuración utilizado para indicar el puerto serial al que debemos acceder.

La información que se usa se corresponde al puerto serial por donde se envía la información al robot para que realice el movimiento correspondiente. Esta información estaría formada por el identificador del servo que se desea mover y el grado al que se desea posicionar.

4.5. Diseño de componentes

4.5.1. Modelo caso de uso

El diagrama correspondiente se adjunta en el Anexo G.4.

4.5.2. Modelos de comportamiento

El diagrama correspondiente se adjunta en el Anexo G.5.

4.6. Parametrización del software base

La aplicación se ha desarrollado sin contar con software base.

El Grupo de Investigación de Robótica Aplicada proporcionó una función que servía para mover un solo servo. Esta función ha sido mejorada y adaptada para poder realizar el movimiento de los servos, encapsulándola en una clase.

5. Implementación del Sistema

En este apartado se analiza la codificación del proyecto, el entorno de desarrollo utilizado, los lenguajes empleados, y las distintas pautas realizadas para generar un código legible y de calidad.

Además, se informará sobre las principales funciones utilizadas del código fuente y su utilidad.

5.1 Entorno Tecnológico

Para realizar la aplicación se han utilizado los siguientes entornos;

Marco tecnológico

Se ha realizado el producto para dispositivos como ordenadores portátiles o ordenadores de sobremesa, así como para los distintos Sistemas Operativos que estos puedan tener, como son Windows, Linux o MacOS.

Entorno de desarrollo

- **Processing**

Con él se ha realizado la capa de presentación. Fue escogido principalmente, por su fácil integración con el software de Arduino. Trabaja bajo un lenguaje conocido por el equipo de desarrollo, como es *Java* y, además, el consumo de recursos es bastante bajo con respecto a otras herramientas. También ha influido su fácil exportación a otros entornos software.

- **Arduino**

El cliente pidió que trabajásemos con esta plataforma. Usada para implementar la capa de negocio y de integración.

Lenguaje de programación

Se han utilizado varios lenguajes de programación, dependiendo de la parte del desarrollo del proyecto.

Para la parte de programación de Arduino, donde se realizan los movimientos del robot, se ha usado el lenguaje *C#*, en cambio, para el desarrollo de la parte de la interfaz se ha hecho uso de *Java*.

5.2 Código fuente

A continuación, se enumeran y explican las partes más relevantes del código fuente y su principal funcionamiento. El código se encuentra en el Anexo H.

Los métodos más relevantes desarrollados para crear la aplicación son:

- **Clase ServoPFM:** Esta clase encapsula los distintos métodos para realizar los movimientos de los nuevos servos *PFM*. A cada objeto de la clase se le especificará el pin de la placa de Arduino al cual estará conectado. Será durante la construcción del objeto donde se le asignará dicho pin, no pudiendo modificarlo más adelante.

Mediante el método `write()` se podrá indicar el posicionamiento que le queremos dar al servo, el valor de los grados deben estar entre los valores 0 y 180.

Mediante el método `read()` se podrá saber en qué posición se encuentra el servo actualmente.

- **Captación de Información por el puerto Serial:** Esta parte del código, es una de las más importantes, ya que recibimos por el puerto serial una serie de información que debemos procesar para generar los movimientos de los servos.

El programa recibe una cadena de cinco caracteres de enteros, los dos primeros caracteres indican el servo a mover (01-16) y los siguientes tres, el grado al que queremos poner el servo especificado.

- **Enviar información del ScrollBar por el puerto Serial:** Se realiza el envío de información hacia el robot a través de la interacción del usuario mediante los scrollbars de la interfaz.

Cada vez que el usuario mueve uno de los scrollbars asociados a una articulación, el programa envía una cadena con información sobre el servo que se está moviendo, y el grado al que se debe posicionar. Solo se enviará información, cuando el sistema

capte que el grado de alguno de los servos ha sido modificado. Esto se hará mediante un array de cambios, que nos comprueba si el grado de alguno de ellos ha sido modificado en la última iteración del programa.

- **Reproducción de movimientos Guardados:** Anteriormente se deben haber guardado todos los movimientos que queramos reproducir mediante la función de guardado. Estos movimientos son incluidos en una matriz de tamaño (16x16), guardando así las 16 posiciones de los servos durante 16 movimientos predeterminados. Una vez pulsado el botón de reproducir, la posición o grado de los servos se irán enviando a través del puerto serial para que el robot los reproduzca según la información que reciba. Esta función podrá ser interrumpida en cualquier momento mediante el botón "STOP".

5.3 Calidad de código

Para que el código del proyecto sea de la mayor calidad posible, se han seguido una serie de pautas durante la implementación de la aplicación.

La disposición de los archivos que componen la aplicación están organizados por un sistema de carpetas específicas para cada función.

El código necesario para la parte del movimiento del robot está diferenciado de la parte de la interfaz. Dentro de esta última parte, se pueden diferenciar carpetas para las imágenes, el código, y los ficheros de configuración previa.

El código está documentado de forma interna, señalando las partes más relevantes para comprender mejor su funcionamiento, y especificado el tipo de variables según el funcionamiento que debe cumplir.

Por último se ha realizado la mayoría de los elementos que componen la interfaz en distintos tipos de clases que interactúan entre sí para disminuir la cohesión y mejorar la reutilización del código.

6. Pruebas del Sistema

Esta sección está dedicada a las pruebas realizadas al sistema software, asegurando un cumplimiento de los requisitos especificados con anterioridad. Se muestra el entorno en el que se han llevado a cabo las pruebas y, seguidamente, se detallan las diferentes pruebas realizadas.

6.1. Descripción del Entorno de Pruebas

A continuación se muestran las características de los diferentes dispositivos donde se han realizado las pruebas;

- **MACBOOK AIR**
 - Sistema Operativo MacOS Sierra.
 - CPU: Intel Core i5 1,6Ghz.
 - Memoria RAM: 8Gb.
 - Tarjeta Gráfica: Intel HD Graphics 6000.
 - Disco Duro SSD 128Gb.
- **MSI**
 - Sistema Operativo Windows 10 64bits
 - CPU: Intel Core i7 7700hq.
 - Memoria RAM: 16Gb.
 - Tarjeta Gráfica: Nvidia Gforce GTX1050Tii.
 - Disco Duro SSD 128Gb.

- **Asus**
 - Sistema Operativo Ubuntu.
 - CPU: Intel Core i7 3720hq.
 - Memoria RAM: 8Gb.
 - Tarjeta Gráfica: Nvidia Gforce GTX 930m
 - Disco duro SSD 128Gb.

6.2. Pruebas Unitarias

A continuación se muestran los casos de pruebas unitarias que se han realizado sobre el software:

Código	P-01.
Nombre	Mover Mano Derecha.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la mano derecha del robot.
Estado	Comprobado y Verificado.

Tabla 8. Verificación Mover Mano Derecha

Código	P-02.
Nombre	Mover Mano Izquierda.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la mano izquierda del robot.
Estado	Comprobado y Verificado.

Tabla 9. Verificación Mover Mano Izquierda

Código	P-03
---------------	------

Nombre	Mover Codo Derecho.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del codo derecho del robot.
Estado	Comprobado y Verificado.

Tabla 10. Verificación Mover Codo Derecho

Código	P-04.
Nombre	Mover Codo Izquierdo.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del codo izquierdo del robot.
Estado	Comprobado y Verificado.

Tabla 11. Verificación Mover Codo Izquierdo

Código	P-05.
Nombre	Mover Hombro Derecho.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del hombro derecha del robot.
Estado	Comprobado y Verificado.

Tabla 12. Verificación Mover Hombro Derecho

Código	P-06.
Nombre	Mover Hombro Izquierdo.

Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del hombro izquierdo del robot.
Estado	Comprobado y Verificado.

Tabla 13. Verificación Mover Hombro Izquierdo

Código	P-07.
Nombre	Mover Pie Derecho.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del pie derecha del robot.
Estado	Comprobado y Verificado.

Tabla 14. Verificación Mover Pie Derecho

Código	P-08.
Nombre	Mover Pie Izquierdo.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento al pie izquierdo del robot.
Estado	Comprobado y Verificado.

Tabla 15. Verificación Mover Pie Izquierdo

Código	P-09.
Nombre	Mover Rodilla Derecha.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la rodilla derecha del robot.

Estado	Comprobado y Verificado.
---------------	--------------------------

Tabla 16 Verificación Mover Rodilla Derecha

Código	P-10.
Nombre	Mover Rodilla Izquierda.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la rodilla izquierda del robot.
Estado	Comprobado y Verificado.

Tabla 17. Verificación Mover Rodilla Izquierda

Código	P-11.
Nombre	Mover Cadera Derecha Vertical.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la cadera derecha vertical del robot.
Estado	Comprobado y Verificado.

Tabla 18. Verificación Mover Cadera Derecha Vertical

Código	P-12.
Nombre	Mover Cadera Derecha Horizontal.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la cadera derecha horizontal del robot.
Estado	Comprobado y Verificado.

Tabla 19. Verificación Mover Cadera Derecha Horizontal

Código	P-13.
Nombre	Mover Cadera Izquierda Vertical.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la cadera izquierda vertical del robot.
Estado	Comprobado y Verificado.

Tabla 20. Verificación Mover Cadera Izquierda Vertical

Código	P-14.
Nombre	Mover Cadera Izquierda Horizontal.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento de la cadera izquierda horizontal del robot.
Estado	Comprobado y Verificado.

Tabla 21. Verificación Mover Cadera Izquierda Horizontal

Código	P-15.
Nombre	Mover Tobillo Derecho.
Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del tobillo derecho del robot.
Estado	Comprobado y Verificado.

Tabla 22. Verificación Mover Tobillo Derecho

Código	P-16.
Nombre	Mover Tobillo Izquierdo.

Descripción	En la página principal de la aplicación, pulsar sobre el scrollbar asignado al movimiento del tobillo izquierdo del robot.
Estado	Comprobado y Verificado.

Tabla 23. Verificación Mover Tobillo Izquierdo

Código	P-17.
Nombre	Botón Reproducir Secuencia de Movimiento.
Descripción	En la página principal de la aplicación, pulsar sobre el botón asignado a la reproducción de la secuencia de movimientos establecidas sobre los servos.
Estado	Comprobado y Verificado.

Tabla 24. Verificación Botón Reproducir Secuencia de Movimiento

Código	P-18.
Nombre	Botón Stop Secuencia de Movimiento.
Descripción	En la página principal de la aplicación, pulsar sobre el botón asignado al stop de la secuencia de movimientos establecidas sobre los servos ya se encuentra en ejecución.
Estado	Comprobado y Verificado.

Tabla 25. Verificación Botón Stop Secuencia de Movimiento

Código	P-19.
Nombre	Botón Reset Posición de los Servos.

Descripción	En la página principal de la aplicación, pulsar sobre el botón asignado al reset de las posiciones en la que se encuentran actualmente los servos.
Estado	Comprobado y Verificado.

Tabla 26. Verificación Botón Reset Posición de los Servos

Código	P-20.
Nombre	Botón Guardar Secuencia de Movimiento.
Descripción	En la página principal de la aplicación, pulsar sobre el botón asignado para guardar la secuencia de movimiento establecida sobre los servos para posteriormente reproducir.
Estado	Comprobado y Verificado.

Tabla 27. Verificación Botón Guardar Secuencia de Movimiento

Código	P-21.
Nombre	Botón Nosotros.
Descripción	En la página principal de la aplicación, pulsar sobre el botón asignado a nosotros que mostrará la información sobre los desarrolladores de la aplicación
Estado	Comprobado y Verificado.

Tabla 28. Verificación Botón Nosotros

Código	P-22.
Nombre	Botón Información.

Descripción	En la página principal de la aplicación, pulsar sobre el botón asignado a la información de la aplicación, que mostrará información de interés para el usuario que ejecute la aplicación.
Estado	Comprobado y Verificado.

Tabla 29. Verificación Botón Información

A continuación se muestran los casos de pruebas unitarias que se han realizado sobre el software para el movimiento específico de cada servo, para ello se han realizado pruebas que se muestre el grado correcto, que la barra de desplazamiento funcione en el rango correspondiente, que la articulación asociada a cada barra se ilumine como seleccionada y por último que mande la información por el serial para mover el servo específico.

Nombre de la Prueba	Grado	Scrollbar	Articulación	Serial
P-01	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-02	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-03	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-04	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-05	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-06	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-07	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-08	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-09	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-10	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-12	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-13	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-14	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-15	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
P-16	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Tabla 30. Verificación Completa de Pruebas Unitarias

A continuación se muestran los casos de pruebas unitarias que se han realizado sobre el botón reproducir secuencia de movimiento, para ello se han realizado pruebas que reproduzca una a una la secuencia de movimientos, que se muestre seleccionado el cuadro de la secuencia correspondiente y que no se pueda reproducir si no hay secuencia guardada:

Nombre de la Prueba	Reproducir	Cuadro seleccionado	Comprobar si existe secuencia.
P-17	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Tabla 31. Verificación Botón Reproducir Secuencia de Movimientos Pruebas Unitarias

En la aplicación, no existe ninguna vista exclusiva para el usuario administrador, ya que todas las vistas serán usadas por todos los usuarios.

6.3. Pruebas de Integración

Una vez que se han realizado las pruebas unitarias al sistema, se llevan a cabo las pruebas integradas, consistentes en realizar las pruebas a los elementos unitarios que actúan entre sí para comprobar que no existen errores. A continuación se muestran los casos de pruebas integradas que se ha realizado sobre el software:

Código	P-23.
Nombre	Mover Brazo Derecho
Descripción	En la página principal de la aplicación, según una secuencia programadas de movimientos de las partes que forma el Brazo Derecho.
Estado	Comprobado y Verificado.

Tabla 32. Verificación Mover Brazo Derecho

Código	P-24.
Nombre	Mover Brazo Izquierdo

Descripción	En la página principal de la aplicación, según una secuencia programadas de movimientos de las partes que forma el Brazo Izquierdo.
Estado	Comprobado y Verificado.

Tabla 33. Verificación Mover Brazo Izquierdo

Código	P-25.
Nombre	Mover Pierna Derecha
Descripción	En la página principal de la aplicación, según una secuencia programadas de movimientos de las partes que forma la Pierna Derecha.
Estado	Comprobado y Verificado.

Tabla 34. Verificación Mover Pierna Derecha

Código	P-26.
Nombre	Mover Pierna Izquierda.
Descripción	En la página principal de la aplicación, según una secuencia programadas de movimientos de las partes que forma la Pierna Izquierda.
Estado	Comprobado y Verificado.

Tabla 35. Verificación Mover Pierna Izquierda

6.4. Pruebas de Sistema

Al finalizar el proceso de desarrollo de la aplicación, se llevaron a cabo prueba complejas del sistema realizadas por los distintos miembros del equipo de desarrollo en sus respectivos dispositivos.

6.5. Pruebas de Aceptación

Finalmente, una vez realizados todos los casos de prueba sobre el software, se ha comprobando que el funcionamiento del mismo es correcto y que no se producen ningún tipo de fallo o error en la aplicación, el producto final es enviado al cliente para su prueba en un entorno real.

7. Conclusiones

En este último capítulo se detallan las lecciones aprendidas tras el desarrollo del presente proyecto y se identifican las posibles oportunidades de mejora sobre el software desarrollado.

7.1. Objetivos alcanzados

- Realización de una aplicación a través de la cual se puedan controlar los movimientos de un robot humanoide.
- Dar la posibilidad de que la aplicación pueda almacenar posiciones de los diferentes servos para así poder reproducir secuencias de movimientos a posteriori.

7.2. Lecciones aprendidas

Gracias a este proyecto se ha experimentado un primer acercamiento hacia el Trabajo de Fin de Grado.

A nivel académico, se han adquirido nuevos conocimientos, ya que no todos los miembros del equipo de desarrollo habían trabajado antes con Arduino.

A nivel profesional, se han adquirido aptitudes relativas al desarrollo de proyectos informáticos en equipos, con roles diferenciados, aunque aún queda mejorar y perfeccionar las diferentes técnicas usadas.

Por último, a nivel personal, se ha aprendido a realizar ajustes en la planificación y a repartir el trabajo de forma equitativa. Se ha ganado experiencia en el trabajo colaborativo, dadas las distintas tomas de decisiones, los diferentes puntos de vista, etc.

7.3. Trabajo futuro

Tras haber cumplido los objetivos marcados, se plantean una serie de mejoras para un trabajo futuro;

- El próximo objetivo será que la interfaz reproduzca los movimientos que deberá realizar a su vez el humanoide conectado al ordenador, es decir, realizar una interfaz en 3D.
- Crear una librería para poder manejar de forma más fácil el nuevo servo mediante frecuencia de pulsos, como sugerencia del cliente.
- Adaptar la aplicación para que sea capaz de funcionar con cualquier tipo de robots, independientemente del tipo de servo que use.
- Exportar la aplicación a dispositivos móviles, para que el robot no tenga por qué estar conectado a un PC, y pueda manejarse desde un smartphone o tablet.

Anexos

A. Manual de implantación y explotación

Las instrucciones de instalación y explotación del sistema se detallan a continuación.

a. Introducción

En este apartado se describe todo lo referente a los requisitos, la instalación y puesta en marcha de la aplicación.

b. Requisitos mínimos

- Robot construido con servos *PFM*.
- Equipo con Sistema Operativo Windows 7.
- i3 con 4 GB de Ram.
- JAVA.
- Arduino.
- Processing.

c. Requisitos recomendados:

- Robot construido con servos *PFM*.
- Equipo con S.Operativo Windows 10.
- i7 con 8 GB de Ram.

- JAVA.
- Arduino.
- Processing.

d. Inventario de componentes

- Para el sistema operativo Windows se crearán dos carpetas:
 - aplicación.windows32: Contendrá un ejecutable .exe para la ejecución de la aplicación.
 - aplicación.windows64: Contendrá un ejecutable .exe para la ejecución de la aplicación.
- Para el sistema operativo Linux se crearán cuatro carpetas:
 - aplicación.linux32: Contendrá un ejecutable .exec para la ejecución de la aplicación.
 - aplicación.linux64: Contendrá un ejecutable .exec para la ejecución de la aplicación.
 - aplicación.linux-arm64: Contendrá un ejecutable .exec para la ejecución de la aplicación.
 - aplicación.linux-armv6hf: Contendrá un ejecutable .exec para la ejecución de la aplicación.
- Para el sistema operativo MACOS X se creará una carpeta:
 - aplicación.macosx: Contendrá un ejecutable .app para la ejecución de la aplicación.

e. Procedimiento de instalación.

- Descargar u obtener una de las anteriores versiones dependiendo del sistema operativo en que vaya a ejecutar la aplicación.
- Iniciar aplicación.

f. Pruebas de implantación.

Realización de pruebas de regresión (repetición de todas las pruebas anteriormente descritas) una vez instalado.

g. Nivel de servicio

Se recomienda el cumplimiento de los requisitos recomendados para un funcionamiento seguro con un rendimiento aceptable de la aplicación.

En dispositivos con memorias RAM o procesadores de bajas prestaciones (inferior a los requisitos mínimos) pueden ocurrir situaciones inesperadas, tales como finalización

inesperada de la aplicación, ralentizaciones en ciertos puntos de la misma, bloqueo de la aplicación o pérdida de señal con el robot.

B. Manual de usuario

Las instrucciones de uso del sistema se detallan a continuación.

a. Introducción

En el siguiente manual vamos a explicar el uso de la aplicación para el manejo de robot donde se podrán mover las articulaciones del robot por separado o realizando una secuencia de movimiento.

b. Características

En esta aplicación se ha implementado diferente tipos de funcionalidades:

- Mover mano derecha.
- Mover mano izquierda.
- Mover codo derecho.
- Mover codo izquierdo.
- Mover hombro derecho.
- Mover hombro izquierdo.
- Mover pie derecho.
- Mover pie izquierdo.
- Mover rodilla derecha.
- Mover rodilla izquierda.
- Mover cadera derecha vertical.

- Mover cadera derecha horizontal.
- Mover cadera izquierda vertical.
- Mover cadera izquierda horizontal.
- Mover tobillo derecho.
- Mover tobillo izquierdo.
- Botón reproducir secuencia de movimiento.
- Botón stop secuencia de movimiento.
- Botón reset posición inicial de los servos.
- Botón guardar secuencia de movimiento.
- Botón autores.
- Botón información.

c. Requisitos previos

Los requisitos necesarios para poder ejecutar la aplicación son los siguientes:

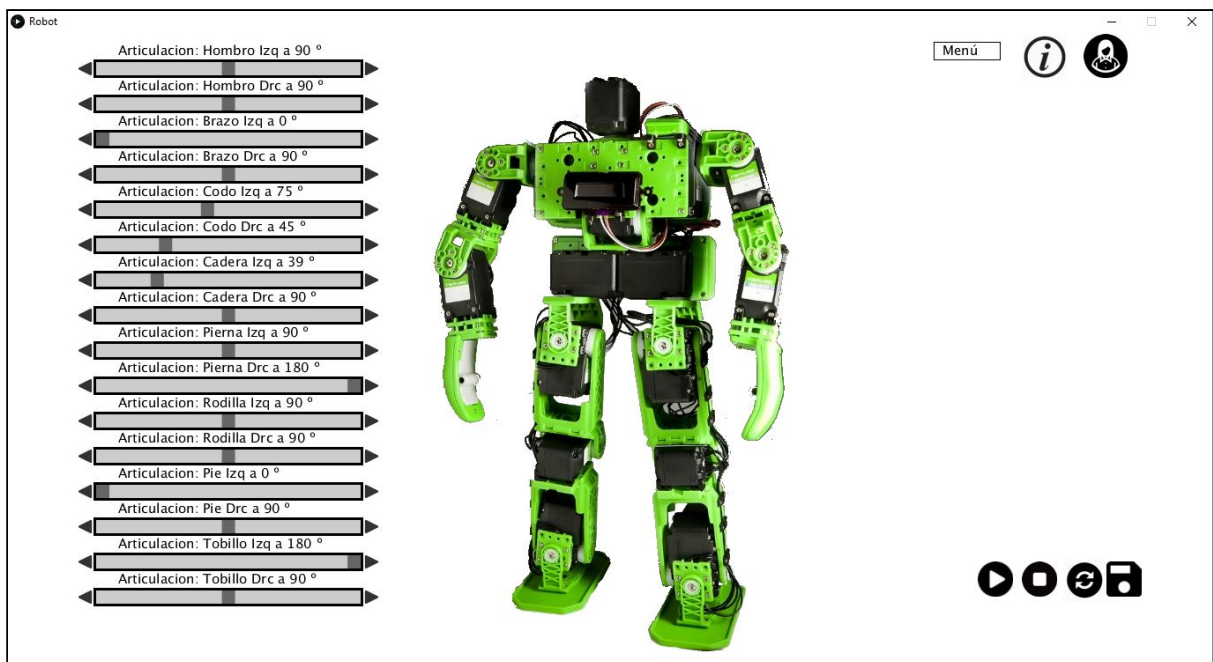
- Tener instalado Processing.
- Resolución de pantalla como mínimo de 1366x768.

* Si tienes una configuración inferior que está no se verá la pantalla principal de la aplicación de forma completa.

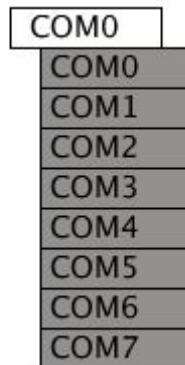
d. Uso del sistema


A continuación se describen todos los aspectos necesarios para la utilización de forma efectiva y eficiente del sistema por parte del usuario.

Ventana Principal:




- Botón Menú, antes de nada tenemos que hacer clic en este botón para seleccionar el puerto al que está conectado el robot al ordenador donde estamos ejecutando la aplicación.



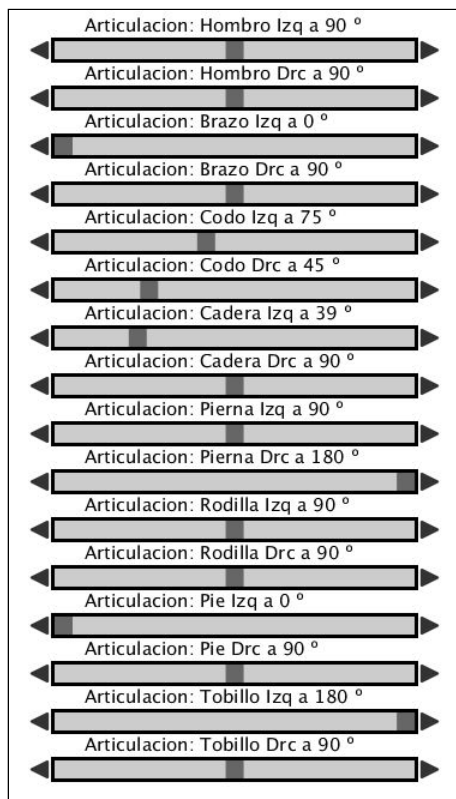
- Botón Indicaciones , en dicho apartado se informará sobre el uso de la aplicación y de las diferentes partes en la que se encuentra dividida la aplicación.



- Botón Información , en dicho apartado se informa de la autoría de la aplicación.







- Área de movimientos, donde se le asignan los valores en grados del movimiento deseado para cada articulación del robot.



Al realizar un movimiento de la extremidad que queramos la parte al que corresponde en el robots se iluminará.



- Botón guardar secuencia de movimientos  , al hacer clic en este botón se guardará la secuencia de movimiento que se ha establecido en el área de movimientos que después serán reproducidas por el robot.
- Botón Iniciar secuencia de movimientos  , al hacer clic en este botón se iniciará la secuencia de movimientos guarda en la aplicación que anteriormente ha tenido que realizar el usuario.
- Botón reset  , donde se resetean los grados establecidos para cada articulación a la posición inicial de los mismos.
- El Botón stop secuencia de movimiento  , al hacer clic en este botón se para la reproducción de la secuencia de movimientos.

C. Documento de Requisitos de Usuario

A continuación se muestra el documento que el cliente firmó al principio del desarrollo del proyecto;



Universidad de Cádiz
Escuela Superior de Ingeniería

Documento de Requisitos de Usuario

- Sistema de Comunicación con Robot Humanoide Bioinspirado -

Fecha: 05/04/2017

Versión: 2.0

Equipo de Desarrollo:

Nicolás Martos Wegener	Manager	nicolas.martoswe@alum.uca.es
------------------------	---------	------------------------------

Marta Jiménez Cordón	Analista	marta.jimenezcordon@alum.uca.es
Laura Álvarez Gallego	Diseñadora	laura.alvarezgallego@alum.uca.es
José M. Tocino Sierra	Desarrollador	jose.tocinosierra@alum.uca.es
Sergio Gómez Benítez	Desarrollador	segio.gomezbe@alum.uca.es
Juan Sevillano Hernández	Tester	juan.sevillanohernandez@alum.uca.es

Contraparte:

Arturo Morgado Estevez	Cliente	arturo.morgado@uca.es
------------------------	---------	-----------------------

Historia del Documento

Versión	Fecha	Razón del Cambio	Autor(es)
1.0	03/04/2017	Primer borrador	Laura y Marta
2.0	05/04/2017	Primera versión	Laura y Marta

1 Introducción

Los sistemas bioinspirados son sistemas contruidos por medio de hardware configurables y sistemas electrónicos que emulan la forma de pensar, el modo de procesar información y resolución de problemas de los sistemas biológicos.

1.1 Propósito del Sistema

El objetivo general del proyecto será crear una interfaz gráfica donde aparezca un humanoide con las articulaciones marcadas, con posibilidad de movimiento mediante una serie de botones. Una vez pulsado esos botones, generará una respuesta en los servos del Robot Humanoide Bioinspirado, realizando un movimiento en esa articulación.

Los servos a usar para la movilidad del Robot serán del tipo PFM (Modulación por Frecuencia de Pulsos).

1.2 Alcance del Proyecto

El cliente pidió realizar una interfaz gráfica, en la cual mediante la asignación de botones a las articulaciones del robot que aparece a la interfaz, podamos asignarle movimientos a las propias articulaciones del Robot Humanoide. Éste estará montado mediante una serie de prototipos de Servos los cuales trabajarán mediante un nuevo sistema de PFM, a diferencia del sistema de servos PWM (Modulación por Anchura de Pulsos) con los que antes se trabajaba.

El objetivo general será representar dicho movimiento de los servos mediante la interfaz gráfica del programa y haciendo uso de los servos PFM.

Además, se plantean los siguientes objetivos adicionales, cuya cumplimentación dependerá de haber alcanzado el objetivo general en tiempo y forma.

- ☐ Una vez generada la interfaz gráfica sin movimientos, nuestro próximo objetivo será que nuestra interfaz reproduzca los movimientos que deberá realizar a su vez el humanoide conectado al ordenador.
- ☐ Crear una librería para poder manejar de forma más fácil el nuevo servo mediante frecuencia de pulsos, como sugerencia del cliente.

1.3 Recursos

A continuación indicaremos los recursos materiales necesarios para poder realizar el proyecto en su totalidad, incluyéndose en esta categoría los recursos hardware y software.

El hardware necesario será aportado por el cliente;

- Robot Humanoide realizado con servos PFM.
- Servos PFM.
- Placas de Arduino

Se hará uso de software libre, por lo que no tendrán un coste asociado;

- Processing

- Arduino
- GanttProject
- Unity
- Blender
- Trello

2 Descripción General

Esta sección describe los requisitos funcionales de los Usuarios/Clientes.

2.1 Características de los Usuarios

Los usuarios del sistema son:

Tipo de Usuario	Descripción	# Actual	# Futura (1 año)	Usuarios Contactables
Administrador	Realiza todo el manejo del robot, es decir, puede manejar la interfaz para la manipulación de todas las articulaciones del robot.	1	1	Arturo Morgado arturo.morgado@uca.es

2.2 Perspectiva del Producto según los Usuarios/Clientes

El cliente espera una interfaz gráfica en la cual aparece representado el robot. En principio se realizará una interfaz estática donde se iluminarán las articulaciones que estén en movimiento. Una vez llegados a este punto sin ningún inconveniente, se espera poder realizar una interfaz gráfica en 3D y dinámica.

2.3 Ambiente Operacional de la Solución

El ambiente operacional involucrado en este sistema es el siguiente:

- Placa de Arduino Leonardo
- Equipo con S.Operativo Windows 10
- i7 con 8 GB de Ram
- Processing

2.4 Relación con Otros Proyectos

El sistema no depende de otros sistemas, ni al contrario. Sin embargo, al ser un proyecto para el Grupo de Investigación de Robótica Aplicada, se ve involucrado un grupo de alumnos pertenecientes al Grado en Ingeniería en Electrónica Industrial.

En particular, estos alumnos tienen que facilitarnos la entrega de unos servos que han sido

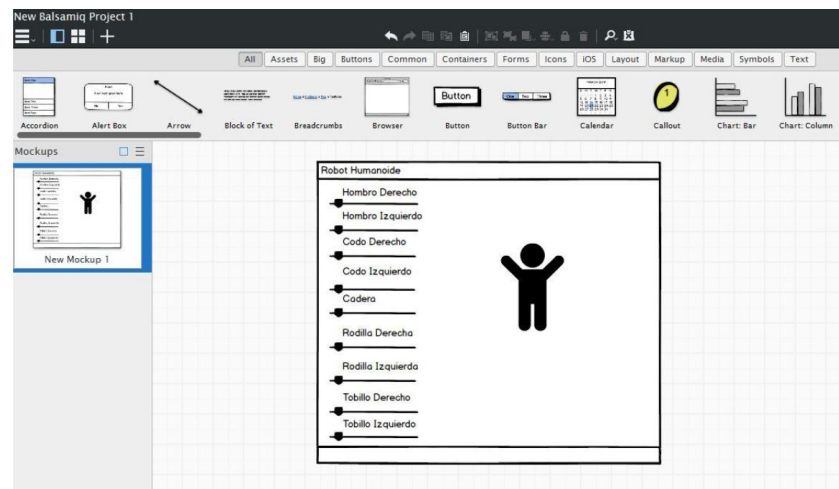
modificados de la siguiente forma, funcionaban con un sistema de modulación por anchura de pulsos y están siendo modificados para que funcionen con un nuevo sistema de modulación por frecuencia de pulsos.

Dichos servos deben ser entregados en la semana del 17 al 21 de Abril, de no ser así el desarrollo del proyecto será modificado;

La idea principal del proyecto es poseer los servos y diseñar una interfaz gráfica estática donde se iluminen las articulaciones que se quieran mover, representando el movimiento en los propios servos.

- ❑ Si dichos servos no están disponibles, se desarrollará una interfaz gráfica donde se represente el robot en movimiento, pero el movimiento no se verá reflejado en el robot humanoide.
- ❑ Si disponemos de al menos un servo, se desarrollará una interfaz estática, donde se iluminen las articulaciones que se estén moviendo y, además, el movimiento se verá reflejado en el robot humanoide.

2.5 Descripción del Modelo



La interfaz presentará un aspecto similar a este. Se contemplarán los 16 servos;

- Brazo derecho
- Brazo izquierdo
- Hombro derecho
- Hombro izquierdo
- Codo derecho
- Codo izquierdo
- Cadera izquierda horizontal
- Cadera izquierda vertical
- Cadera derecha horizontal
- Cadera derecha vertical
- Rodilla izquierda
- Rodilla derecha
- Tobillo izquierdo

- Tobillo derecho
- Pie izquierdo
- Pie derecho

Fdo: Contraparte	Fdo: Equipo de desarrollo

En Puerto Real, a 5 de Abril de 2017.

D. Documento de recogida de material

A continuación se muestra el documento que el cliente firmó cuando nos entregó los servos;



“Sistema de comunicación con robot humanoide bioinspirado”

Documento de recogida de material

REUNIDOS

De una parte, don Arturo Morgado Estévez ,con DNI _____

De otra parte, don Nicolás Martos Wegener, con DNI _____ y como Jefe de Equipo del Proyecto *“Sistema de Comunicación con Robot Humanoide Bioinspirado”* de la asignatura Proyectos Informáticos del Grado en Ingeniería Informática de la Universidad de Cádiz.

HACEN CONSTAR

Que, a fecha 24 de Abril de 2017 se hace entrega por parte de don Arturo Morgado Estévez de ____ servomotores *PFM* para el desarrollo del proyecto que implica a ambas partes.

Fdo: Contraparte	Fdo: Equipo de desarrollo

En Puerto Real, a 24 de Abril de 2017.

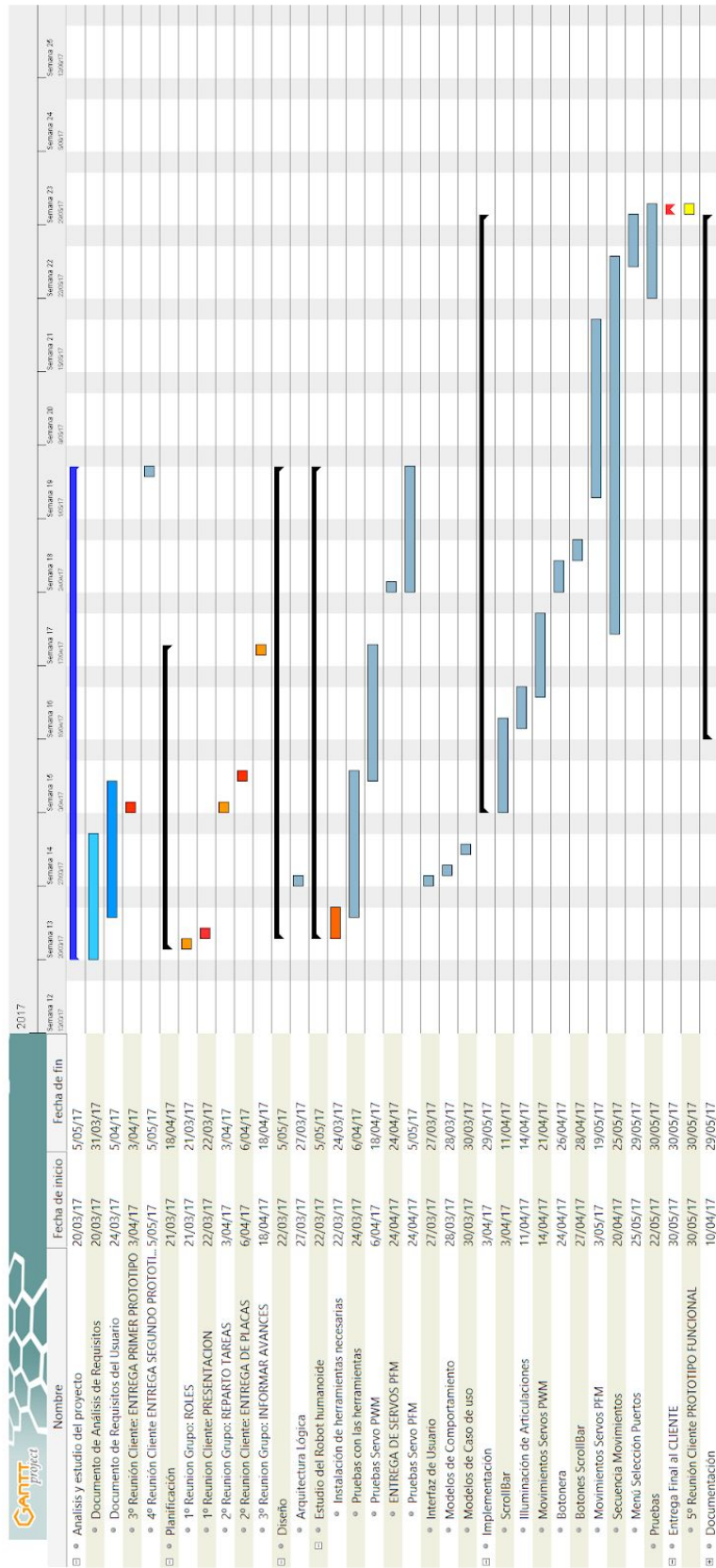
E. Reuniones

El equipo ha estado haciendo reuniones semanales desde el comienzo del proceso de desarrollo en Marzo, además ha tenido diversas reuniones con el cliente para comentar el desarrollo del Sistema de Comunicación con el robot humanoide bioinspirado, así como mostrar prototipos.

Se recogen las que entendemos más relevantes para el desarrollo:

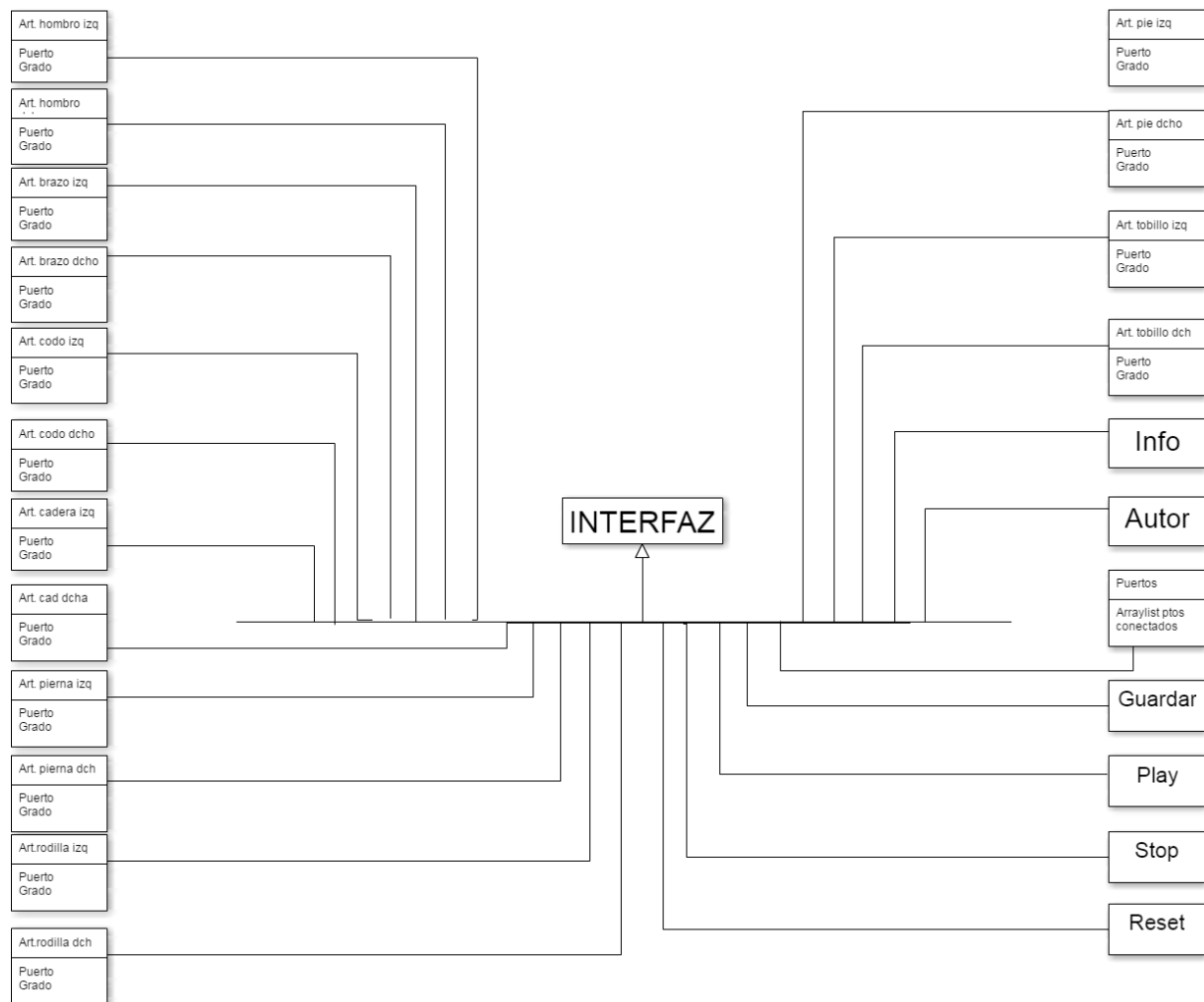
FECHA	OBJETIVO DE LA REUNIÓN
21/Marzo/2017	Asignación de roles
22/Marzo/2017	Primera reunión con el cliente
5/Abril/2017	Firma del DRU
24/Abril/2017	Entrega servos <i>PFM</i>
30/Mayo/2017	Última reunión con el cliente

F. Diagrama de Gantt

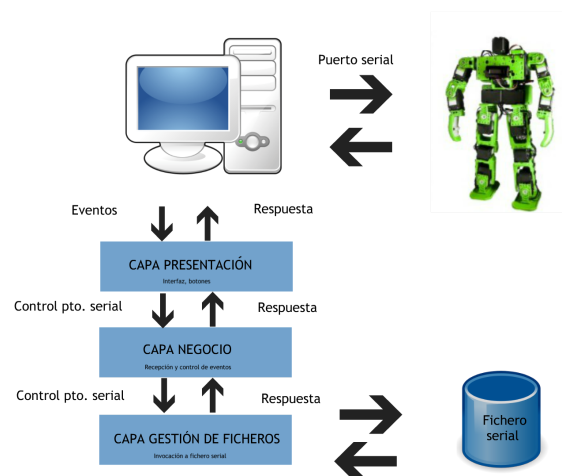


G. Diagramas de diseño

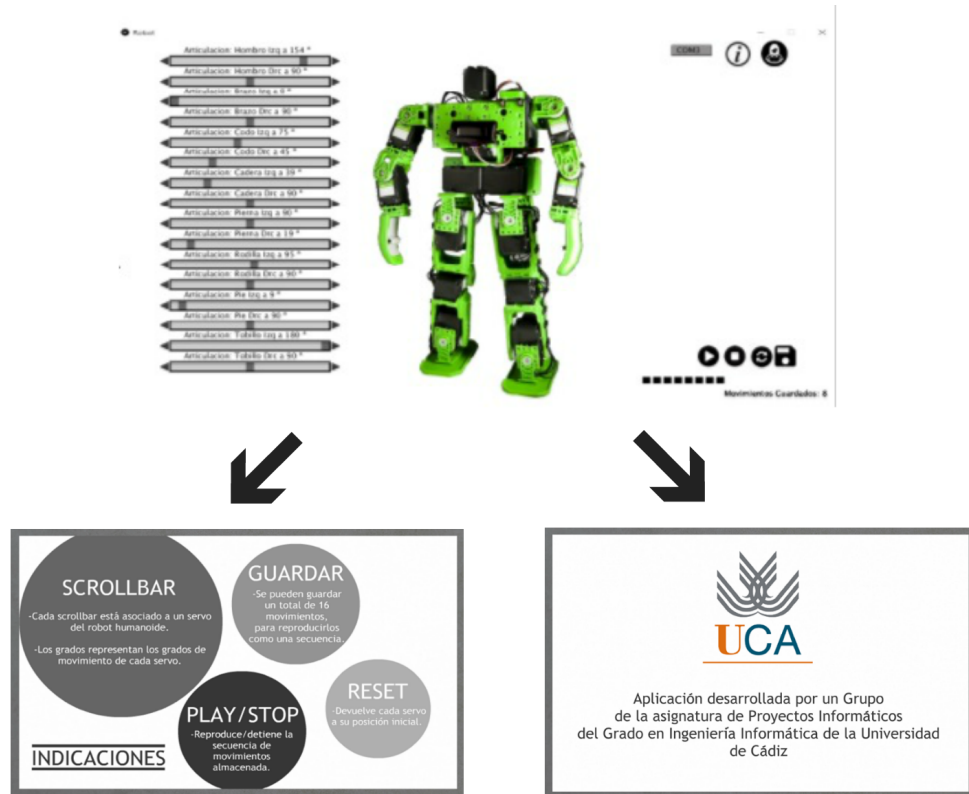
1. Diagrama de diseño



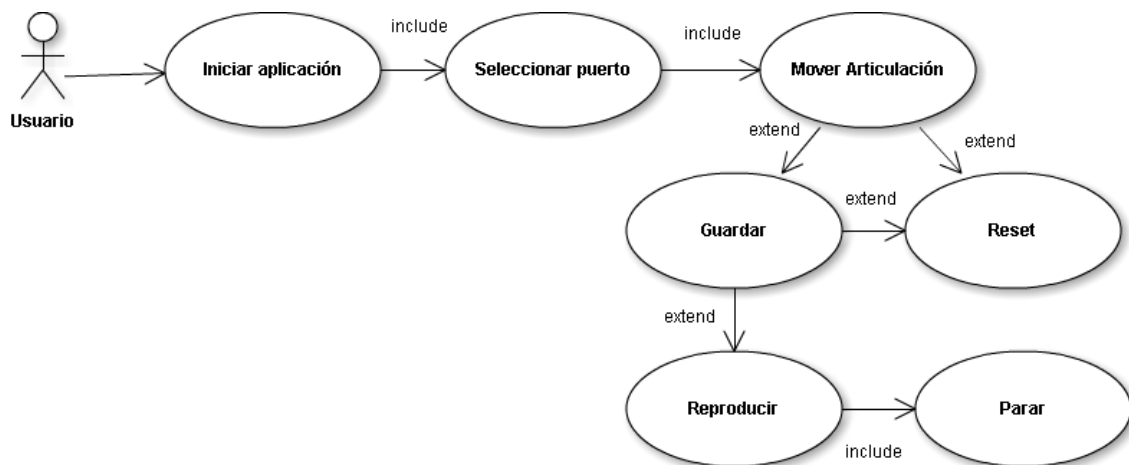
2. Patrón arquitectónico



3. Diseño de la interfaz de usuario

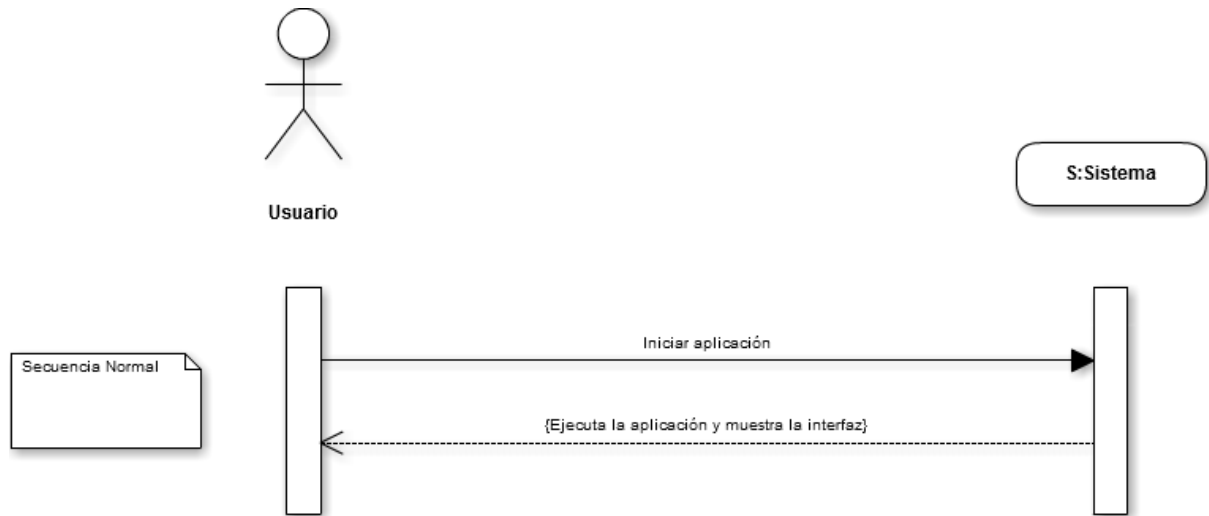


4. Modelo de casos de uso

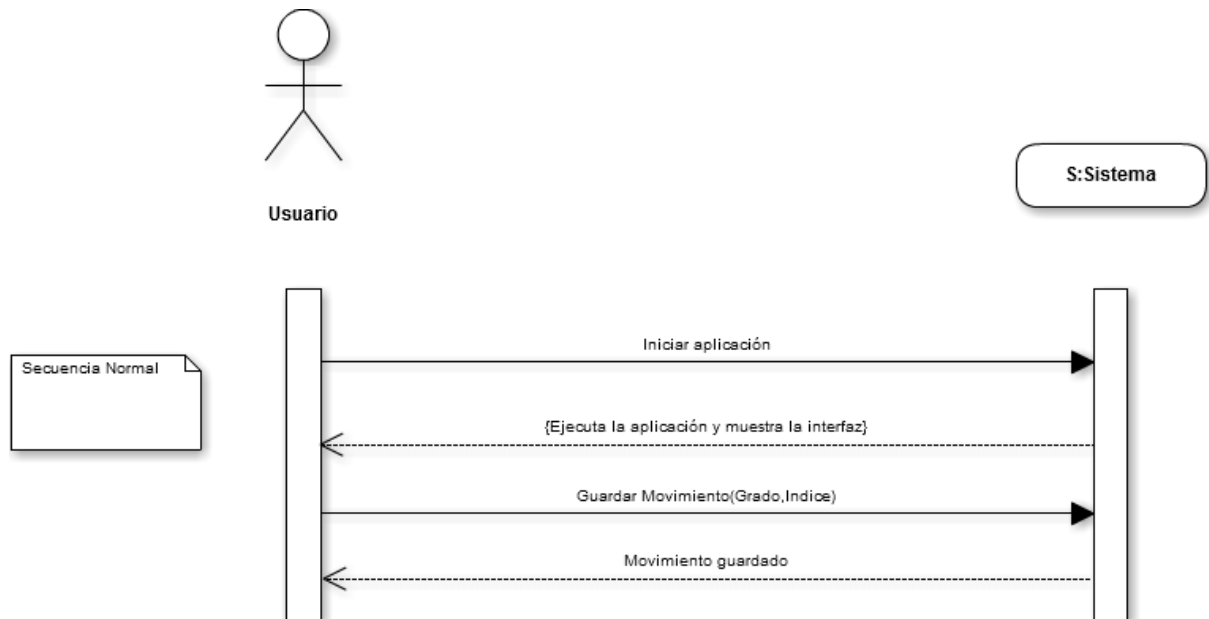


5. Modelo de comportamiento

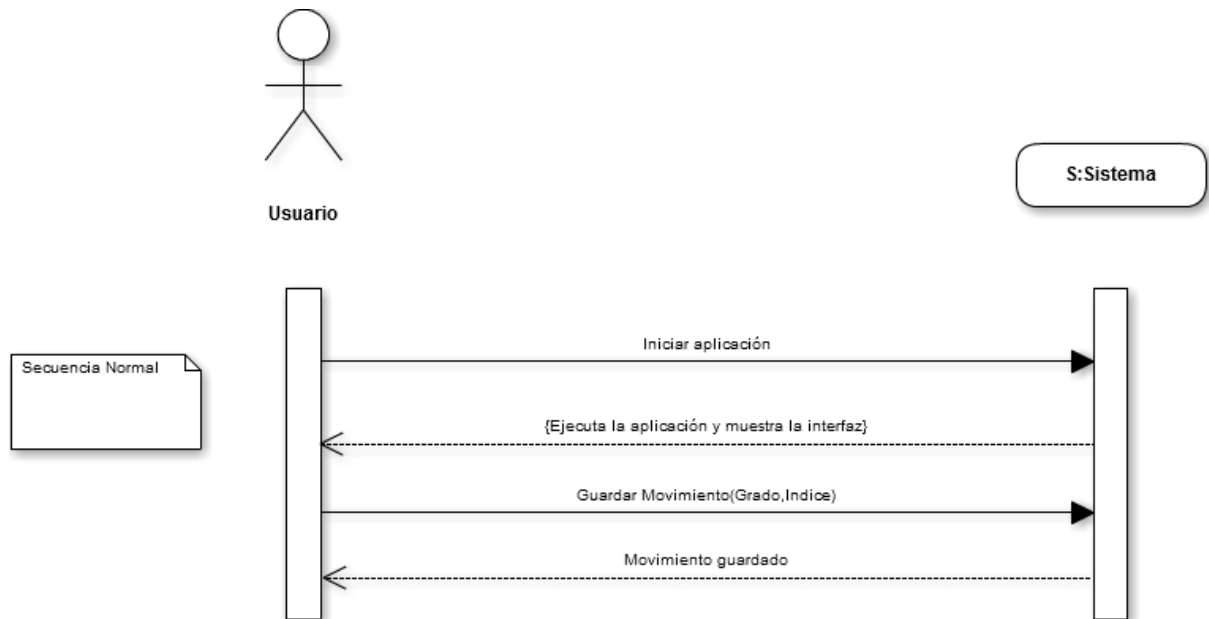
5.1. Operación “Iniciar Aplicación”



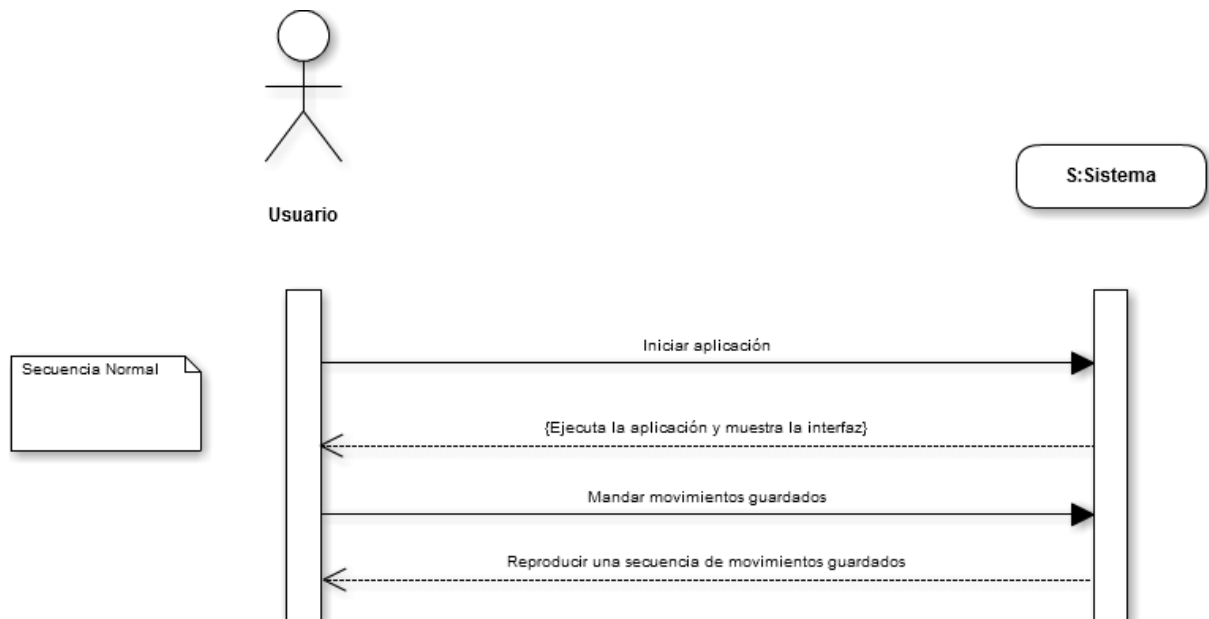
5.2. Operación “Mover Articulación”



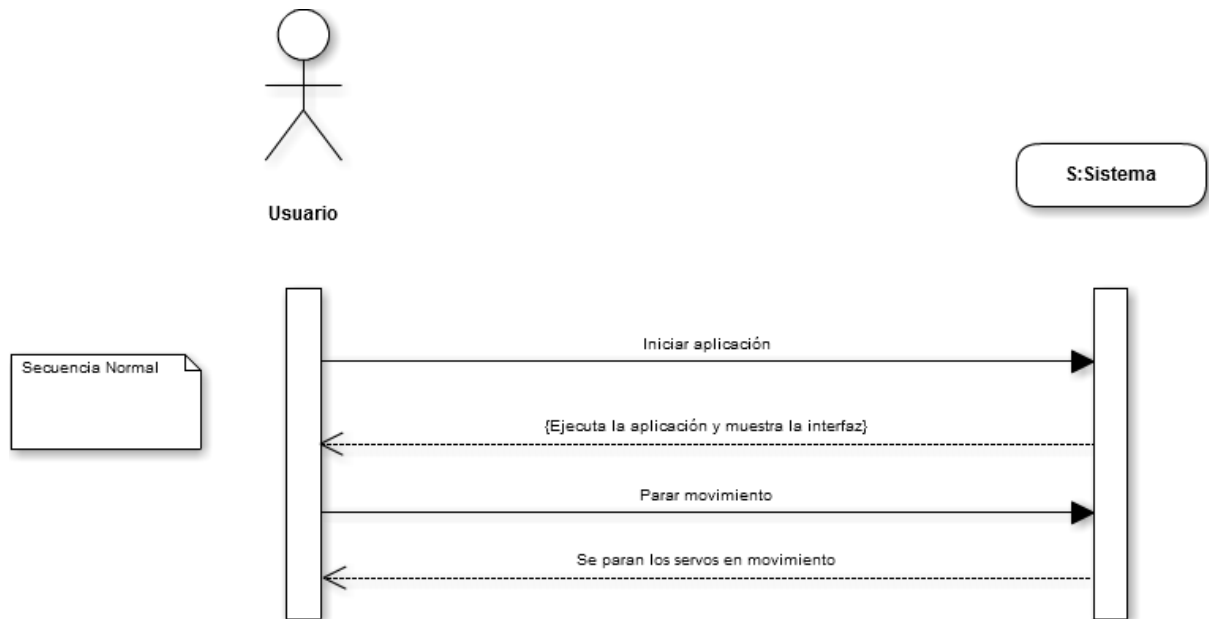
5.3. Operación “Guardar Movimiento Articulación”



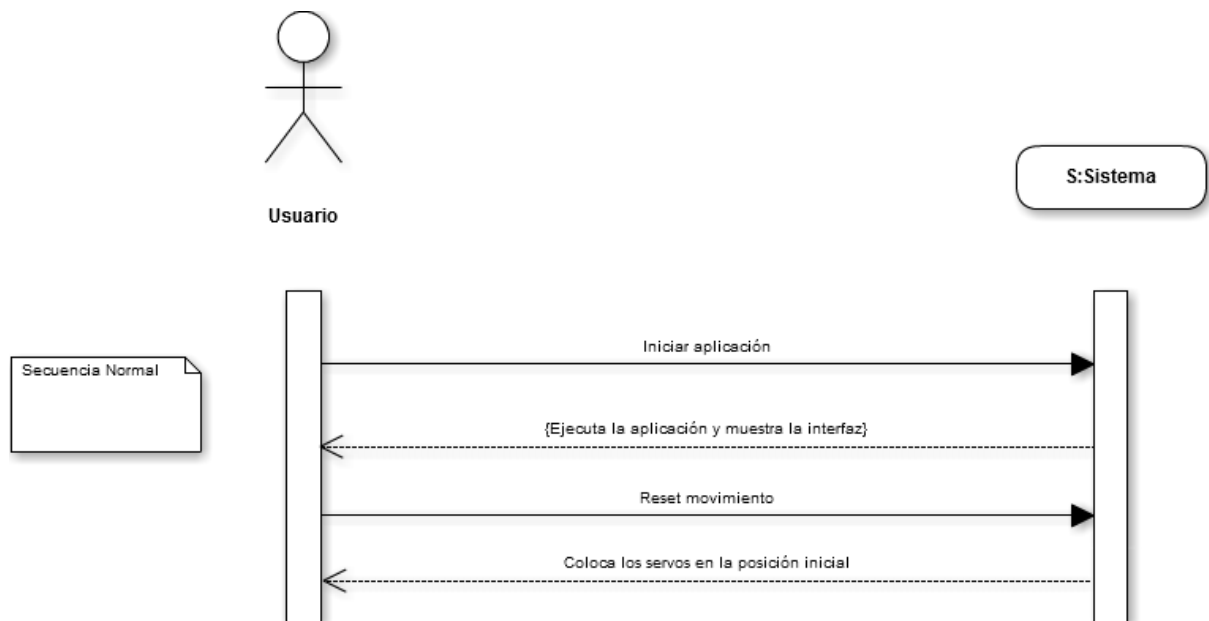
5.4. Operación “Reproducir Secuencia de Movimientos”



5.5. Operación “Parar Secuencia de Movimientos”



5.6. Operación “Reset Movimientos Iniciales Articulaciones”



H. Código Fuente

- Clase ServoPFM

```
//Clase que encapsula los metodos y variables necesarias para mover un servo PFM
class ServoPFM {
private:
    const int pul = 20;
    const int limiteAlto = 2000;
    const int limiteBajo = 1000;
    int grados;
    int pin;
    float freq;
public:
    //Constructor de la clase que recibe un entero que será el pin donde está conectado.
    ServoPFM(int p){
        pin = p;
        pinMode(pin, OUTPUT);
    }
    //Metodo que através de un grado, genera el movimiento del servo
    void write(int grad){
        grados = grad;
        freq = map(grad, 180, 0, limiteAlto, limiteBajo);
        digitalWrite(pin, HIGH);
        delayMicroseconds(pul);
        digitalWrite(pin, LOW);
        delayMicroseconds(freq);
    }
    //Metodo que devuelve la posición actual del servo
    int read(){ return grados;}
};
```

- Captación de Información por el puerto Serial

```
//Comprobamos si el puerto está libre
while(Serial.available()){
    delay(3);
    //Si hay datos pendientes de leer en el puerto se leen
    //caracter a caracter y se guardan en una cadena
    if(Serial.available()>0){
        char c = Serial.read();
        cadena +=c;
    }
}
//La cadena resultante se trata y se saca la información del servo y los grados a mover
//y se guarda en variables de tipo int
op = cadena.substring(0,2);
angulo = cadena.substring(2, 5);
int o = op.toInt();
int grado = angulo.toInt();
```

- Enviar información del ScrollBar por el puerto Serial

```
//MOVIMIENTOS DE LOS SERVOS cuando se realizan uno a uno

for(int i = 0; i < 16; i++){
    //Buscamos el grado de cada servo y comprobamos si se han realizado cambios sobre dicho servo
    if(cambios[i]!=Barras[i].getPosition()){
        //Si se han realizado, formateamos la información para generar la cadena en el formato necesario
        String cadena="";
        int id = Barras[i].IdEstructura();//Dada el índice del servo y el grado que queremos mandar
        //Los formateamos según el formato IDGRADO Ejmplo(01180)
        int grados = Barras[i].getPosition();
        if(id<10){cadena+=00+" "+id;}
        else{cadena+=" "+id;}
        if(grados <10){cadena+="00"+grados;}
        else if(grados < 100){cadena+="0"+grados;}
        else{cadena+=" "+grados;}
        try{
            port.write(cadena);//Enviamos la información de cada servo cuando este es modificado
        }catch(Exception e){println("Error al enviar los datos");
        }
        cambios[i] = grados; //los grados antiguos se igualan con los recientes
    }
}
for(int i = 0; i < 180; i++){
    try{

    }catch(Exception e){print("Error al leer fichero").1
}
```

- Reproducción de movimientos Guardados

```
//Boton Reproducir
if(Play.isPressed() && movimientos.SetMoves()>=1){reproducir = true;}
//Comprobamos si el boton de stop está presionado
if(Stop.isPressed() || contReproducir>movimientos.SetMoves()){reproducir = false;contReproducir=0;}
//Si el estado es reproduciendo
if(reproducir==true){
    if(Stop.isPressed() || contReproducir>movimientos.SetMoves()){reproducir = false;contReproducir=0;}
    delay(100);
    //Llamamos al metodo que reproduce la secuencia
    movimientos.reproducir(contReproducir);
    contReproducir++;
}
//Metodo que reproduce los movimientos que están guardado en la matriz en una posicion j y los envias por el puerto serial
void reproducir(int j){
    int k = j+1;
    String cadena="";
    if(k <= cont){
        strokeWeight(1);
        //Dibuja los rectangulos que indican que la secuencia se está reproduciendo
        fill(179,47,47);
        rect(xpos+(k*20)+25,ypos,15,15);
        for(int i = 1; i < 17; i++){
            //Crea el formato según los datos para que se adapten al especificado para mover los servos 01180
            if(i<10){cadena+=00+" "+i;}
            else{cadena+=" "+i;}
            if(Mov[j][i-1] <10){cadena+="00"+Mov[j][i-1];}
            else if(Mov[j][i-1] < 100){cadena+="0"+Mov[j][i-1];}
            else{cadena+=" "+Mov[j][i-1];}
            //
            try{
                port.write(cadena);
                delay(25);
            }catch(Exception e){}
        }
    }
}
```

Bibliografía

[Boletín Oficial del Estado, 2017] XVIII Convenio Colectivo Nacional de Empresas de Ingeniería y Oficinas de Estudios Técnicos

URL: <https://www.boe.es/boe/dias/2017/01/18/pdfs/BOE-A-2017-542.pdf>

Fecha último acceso: 25/04/2017

Página Oficial Arduino, URL: <https://www.arduino.cc/>, Fecha último acceso: 30/05/2017

Página Oficial Processing, URL: <https://processing.org/>, Fecha último Acceso: 30/05/2017

[IEE, 2008] Estándar para la especificación de requisitos

Accesible: http://www.ctr.unican.es/asignaturas/is1/IEEE830_esp.pdf

Tojeiro Calaza, Germán. (2014), "Taller de Arduino. un enfoque práctico para principiantes"

Froufe Quintas, Agustín. "Java 2, Manual de usuario y tutorial" 4º Edición.