



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
CARRERA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES
INDUSTRIALES

**“CONSTRUCCIÓN DE UN PROTOTIPO ELECTROMECAÁNICO PARA
REHABILITACIÓN TRAS TRAUMATISMOS DE CODO BASADO EN LA
INTEGRACIÓN DE SISTEMAS DE CONTROL, MONITOREO Y EL
INTERNET DE LAS COSAS”**

Trabajo de titulación

Tipo: Propuesta Tecnológica

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA, CONTROL Y REDES
INDUSTRIALES

AUTORES: TANIA CRISTINA QUINATO A CHICAIZA

WILMER LUIS MORALES RIVERA

DIRECTOR: ING. EDWIN VINICIO ALTAMIRANO SANTILLÁN

Riobamba – Ecuador

2020

©2020, Tania Cristina Quinatoa Chicaiza; & Wilmer Luis Morales Rivera

Se autoriza la reproducción total o parcial, con fines académicos, por cualquier medio o procedimiento, incluyendo la cita bibliográfica del documento, siempre y cuando se reconozca el Derecho de Autor.

Nosotros, Tania Cristina Quinatoa Chicaiza y Wilmer Luis Morales Rivera, declaramos que el presente trabajo de titulación es de nuestra autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autores asumimos la responsabilidad legal y académica de los contenidos de este trabajo de titulación; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 20 de noviembre del 2020.



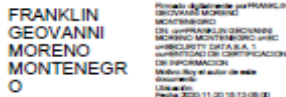
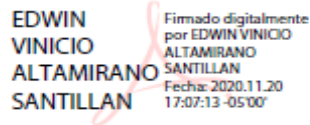
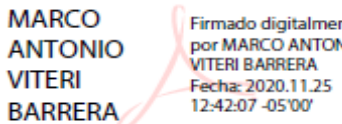
Tania Cristina Quinatoa Chicaiza
C.I: 0503889511



Wilmer Luis Morales Rivera
C.I: 1805359609

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
CARRERA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES
INDUSTRIALES

El Tribunal de Trabajo de Titulación certifica que: Propuesta Tecnológica: “**CONSTRUCCIÓN DE UN PROTOTIPO ELECTROMECAÁNICO PARA REHABILITACIÓN TRAS TRAUMATISMOS DE CODO BASADO EN LA INTEGRACIÓN DE SISTEMAS DE CONTROL, MONITOREO Y EL INTERNETDELAS COSAS**” de responsabilidad de los señores **TANIA CRISTINA QUINATO A CHICAIZA Y WILMER LUIS MORALES RIVERA**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Titulación, quedando autorizada su presentación.

	FIRMA	FECHA
<p>Ing. Franklin Geovanni Moreno Montenegro Msc. PRESIDENTE DEL TRIBUNAL</p>		<p>2020-11-20</p>
<p>Ing. Edwin Vinicio Altamirano Santillán Msc. DIRECTOR DEL TRABAJO DE TITULACIÓN</p>		<p>2020-11-20</p>
<p>Ing. Marco Antonio Viteri Barrera Msc. MIEMBRO DEL TRIBUNAL</p>		<p>2020-11-20</p>

DEDICATORIA

Con humildad y respeto dedico este trabajo primeramente a Dios, al creador de todas las cosas, el que me ha dado fortaleza para seguir adelante a pesar de las adversidades.

A mis queridos padres Joselito y Clarita, que han sabido formarme con buenos sentimientos, hábitos y valores.

A mis hermanos que siempre han estado junto a mi brindándome su apoyo durante todo este proceso.

Tania

Este trabajo le dedico principalmente a Dios, por haberme dado la vida, guiarme por el buen camino, darme fuerzas para seguir adelante y permitirme llegar hasta este momento tan importante de mi formación profesional.

A mis padres porque son la motivación de mi vida, siempre me han apoyado incondicionalmente.

A mis hermanos por creer en mí, por su cariño y apoyarme siempre durante todo este proceso.

Wilmer

AGRADECIMIENTO

Agradezco a Dios, por darme la vida y permitirme cumplir uno de mis preciados anhelos, gracias por derramar sobre mi amor, bendiciones, enseñanzas, alegrías, éxitos, sin ti nada es posible.

A mis padres a quienes sin escatimar esfuerzo alguno han sacrificado gran parte de su vida para formarme y educarme, porque gracias a su apoyo y consejos he llegado a realizar una de mis más grandes metas, la cual constituye la herencia más valiosa que pudiera recibir.

A todos los docentes que han compartido sus conocimientos, gracias a sus enseñanzas y apoyo he constituido la base de mi vida profesional.

Tania

Primeramente, agradezco a Dios por permitirme compartir este logro con mi familia, gracias a mi familia por apoyarme en cada decisión y anhelos que he tenido en el transcurso de la vida.

Gracias a la Escuela Superior Politécnica de Chimborazo por haberme permitido ser parte de ella y abierto las puertas para poder seguir mi carrera académica, así también a todos los docentes quienes supieron brindarme sus conocimientos.

Agradezco también a mi Director de tesis el Ing. Edwin Altamirano por haberme dado la oportunidad de recurrir a su capacidad y conocimiento.

Al igual agradezco a todos mis compañeros y amigos por compartir tantos momentos que se han presentado.

Wilmer

TABLA DE CONTENIDO

ÍNDICE DE TABLAS.....	X
ÍNDICE DE FIGURAS.....	xiii
ÍNDICE DE ANEXOS.....	xviii
RESUMEN.....	xix
SUMMARY	xx
1. INTRODUCCIÓN	
1.1 Planteamiento del Problema.....	2
1.2 Justificación del proyecto.....	3
1.3 Objetivos.....	4
1.3.1 <i>Objetivo General</i>	4
1.3.2 <i>Objetivos Específicos</i>	4
1.4 Alcance.....	4
2. MARCO TEÓRICO	
2.1 Internet de las cosas (IoT).....	6
2.1.1 <i>Campos de aplicaciones de la IoT</i>	7
2.1.2 <i>Componentes de un sistema IoT</i>	8
2.2 Complejo articular del codo	9
2.2.1 <i>Fracturas de codo</i>	10
2.2.2 <i>La rehabilitación física</i>	11
2.2.3 <i>Sistemas ciberfísicos</i>	12
2.3 Plataforma LabVIEW.....	13
2.4 SolidWorks	14
2.5 Python.....	14
2.6 Tarjeta Raspberry	15
2.7 Tarjeta Arduino.....	15
2.8 Motores eléctricos paso a paso.....	16
2.9 Drivers	16
2.10 Fuente de Alimentación.....	17

3.	METODOLOGÍA	
3.1	Determinación de requerimientos para la rehabilitación física	19
3.1.1	<i>Determinación de requerimientos de diseño mecánico</i>	20
3.1.2	<i>Determinación de requerimientos del hardware</i>	20
3.1.3	<i>Determinación de requerimientos del software</i>	21
3.2	Descripción del prototipo esperado	21
3.3	Diseño del prototipo rehabilitador	23
3.3.1	<i>Diseño Mecánico - Modelación CAD</i>	23
3.3.2	<i>Registro de diseños – Planos</i>	24
3.3.3	<i>Ensamble del rehabilitador</i>	25
3.3.4	<i>Análisis estático del prototipo rehabilitador</i>	29
3.3.5	<i>Hardware del prototipo rehabilitador</i>	51
3.3.5.1	<i>Actuadores</i>	51
3.3.5.2	<i>Tarjetas Raspberry Pi3 y Arduino UNO</i>	56
3.3.5.3	<i>Fuente de alimentación</i>	58
3.3.5.4	<i>Diseño - Conexiones de hardware</i>	59
3.3.6	<i>Software del prototipo rehabilitador</i>	62
3.3.6.1	<i>Carga del IDE de Arduino en la Raspberry</i>	62
3.3.6.2	<i>Creación de la cuenta y registro del dispositivo en Remot3 IT</i>	63
3.3.6.3	<i>Programación Microcontrolador</i>	69
3.3.6.4	<i>Programación en Python</i>	74
3.3.7	<i>Enlace LabVIEW y la plataforma Remote it</i>	87
3.3.8	<i>Enlace de LabVIEW con SolidWorks</i>	91
3.3.8.1	<i>Configuraciones en SolidWorks</i>	92
3.3.8.2	<i>Configuraciones en LabVIEW</i>	93
3.4	Implementación del prototipo rehabilitador	95
3.4.1	<i>Construcción del prototipo electromecánico</i>	95
3.4.2	<i>Implementación circuito eléctrico y electrónico</i>	98
3.5	Análisis y Resultados	99
3.5.1	<i>Resultados del Análisis Estático</i>	99
3.5.2	<i>Análisis de resultados estructurales del prototipo</i>	114
3.5.2.1	<i>Regulación de mecánica para la altura del prototipo</i>	114
3.5.2.2	<i>Regulación de mecánica para el apoyo del brazo</i>	115
3.5.3	<i>Pruebas de funcionalidad</i>	117
3.5.3.1	<i>Puesta en marcha de la interfaz gráfica</i>	117
3.5.3.2	<i>Ejecución de conectividad de LabVIEW con Remote it</i>	118

3.5.3.3	<i>Prueba funcionamiento del prototipo en Modo de Calibración</i>	120
3.5.3.4	<i>Prueba funcionamiento del prototipo en Modo de Maniobra</i>	120
3.5.4	<i>Consumo de corriente del prototipo</i>	124
4.	GESTIÓN DEL PROYECTO	
4.1	Cronograma	125
4.2	Recursos y materiales: humanos, equipos, financiamiento	125
4.2.1	<i>Costos</i>	<i>125</i>
4.2.2	<i>Talento Humano</i>	<i>126</i>
4.2.3	<i>Recursos Materiales</i>	<i>126</i>
	CONCLUSIONES	127
	RECOMENDACIONES	128
	BIBLIOGRAFÍA	
	ANEXOS	

ÍNDICE DE TABLAS

Tabla 1-3:	Listado de piezas en el diseño del rehabilitador (Extraída del ANEXO B)....	27
Tabla 2-3:	Propiedades de la pieza - Resorte	30
Tabla 3-3:	Propiedades del material - Resortes	31
Tabla 4-3:	Sujeciones - Resorte.....	31
Tabla 5-3:	Cargas - Resorte	31
Tabla 6-3:	Propiedades de la pieza – Base principal	32
Tabla 7-3:	Propiedades del material – Base principal	33
Tabla 8-3:	Sujeciones - Base principal	33
Tabla 9-3:	Cargas - Base principal	33
Tabla 10-3:	Propiedades de la pieza – Base del motor del antebrazo	34
Tabla 11-3:	Propiedades del material – Base del motor del antebrazo.....	35
Tabla 12-3:	Sujeciones - Base del motor del antebrazo	35
Tabla 13-3:	Cargas - Base del motor del antebrazo.....	35
Tabla 14-3:	Propiedades de la pieza – Base de alojamiento del antebrazo	36
Tabla 15-3:	Propiedades del material – Base de alojamiento del antebrazo	37
Tabla 16-3:	Sujeciones – Base de alojamiento del antebrazo	37
Tabla 17-3:	Cargas – Base de alojamiento del antebrazo.....	37
Tabla 18-3:	Propiedades de la pieza – Soporte del cilindro y resorte	38
Tabla 19-3:	Propiedades del material – Soporte del cilindro y resorte.....	39
Tabla 20-3:	Sujeciones – Soporte del cilindro y resorte.....	39
Tabla 21-3:	Cargas – Soporte del cilindro y resorte.....	39
Tabla 22-3:	Propiedades de la pieza – Soporte para el giro de la muñeca	41
Tabla 23-3:	Propiedades del material – Soporte para el giro de la muñeca	41
Tabla 24-3:	Sujeciones – Soporte para el giro de la muñeca.....	42
Tabla 25-3:	Cargas – Soporte para el giro de la muñeca.....	42
Tabla 26-3:	Propiedades de la pieza – Elemento de movilidad para la mano	43
Tabla 27-3:	Propiedades del material – Elemento de movilidad para la mano	44
Tabla 28-3:	Sujeciones – Elemento de movilidad para la mano	44
Tabla 29-3:	Cargas – Elemento de movilidad para la mano.....	44
Tabla 30-3:	Propiedades de la pieza – Apoyo para rehabilitación de mano.....	45
Tabla 31-3:	Propiedades del material – Apoyo para rehabilitación de mano.....	46
Tabla 32-3:	Sujeciones – Apoyo para rehabilitación de mano	46
Tabla 33-3:	Cargas – Apoyo para rehabilitación de mano	46
Tabla 34-3:	Propiedades de la pieza – Soporte de movimiento de mano.....	47

Tabla 35-3:	Propiedades del material – Soporte de movimiento de mano	48
Tabla 36-3:	Sujeciones – Soporte de movimiento de mano	48
Tabla 37-3:	Cargas – Soporte de movimiento de mano	48
Tabla 38-3:	Propiedades de la pieza – Brazo de movilidad del Antebrazo	49
Tabla 39-3:	Propiedades del material– Brazo de movilidad del Antebrazo	50
Tabla 40-3:	Sujeciones – Brazo de movilidad del Antebrazo	50
Tabla 41-3:	Cargas – Brazo de movilidad del Antebrazo.....	50
Tabla 42-3:	Selección de actuadores	52
Tabla 43-3:	Características eléctricas del Motor NEMA 23 MODEL 23HS9430B.....	53
Tabla 44-3:	Características eléctricas del Motor NEMA 23 MODEL 23HD76002Y-21B.....	54
Tabla 45-3:	Características eléctricas del Motor NEMA 17 MODEL17HS13-0404S.....	54
Tabla 46-3:	Características eléctricas del Driver DM542T	55
Tabla 47-3:	Características del Driver BL-TB6560-V2.0	56
Tabla 47-3:	Características de la Raspberry Pi3.....	57
Tabla 49-3:	Características plataforma Arduino UNO.....	58
Tabla 50-3:	Procesos del CASE.....	72
Tabla 51-3:	Resultado Stress del resorte	100
Tabla 52-3:	Resultado del desplazamiento por esfuerzo del resorte	100
Tabla 53-3:	Resultado Stress de la base principal	101
Tabla 54-3:	Resultado del desplazamiento por esfuerzo de la base principal	102
Tabla 55-3:	Resultado Stress de la base del motor del antebrazo.....	103
Tabla 56-3:	Resultado del desplazamiento por esfuerzo de la base del motor antebrazo	103
Tabla 57-3:	Resultado Stress de la base de alojamiento del antebrazo	104
Tabla 58-3:	Resultado del desplazamiento por esfuerzo de alojamiento del antebrazo ...	105
Tabla 59-3:	Resultado Stress del soporte del cilindro y resorte	105
Tabla 60-3:	Resultado desplazamiento por esfuerzo del soporte del cilindro y resorte ...	106
Tabla 61-3:	Resultado Stress del soporte para el giro de la muñeca	107
Tabla 62-3:	Resultado desplazamiento por esfuerzo del soporte para el giro de muñeca	107
Tabla 63-3:	Resultado Stress del elemento de movilidad para la mano	108
Tabla 64-3:	Resultado desplazamiento por esfuerzo elemento de movilidad para la mano	109
Tabla 65-3:	Resultado Stress del elemento de apoyo para rehabilitación de mano.....	110
Tabla 66-3:	Resultado desplazamiento por esfuerzo apoyo para rehabilitación de mano .	110
Tabla 67-3:	Resultado Stress del elemento del soporte de movimiento de mano	111
Tabla 68-3:	Resultado desplazamiento por esfuerzo soporte de movimiento de mano....	112
Tabla 69-3:	Resultado Stress del elemento del brazo de movilidad del antebrazo.....	113
Tabla 70-3:	Resultado desplazamiento por esfuerzo del brazo de movilidad antebrazo..	113

Tabla 71-3:	Resultados de coberturas corporales del equipo	116
Tabla 72-3:	Consumo de los dispositivos del prototipo	124
Tabla 73-3:	Cronograma.....	125
Tabla 74-3:	Costos directos e indirectos del proyecto.....	126

ÍNDICE DE FIGURAS

Figura 1-2:	Internet de las cosas.....	6
Figura 2-2:	Beneficios IoT	7
Figura 3-2:	Arquitectura IoT	9
Figura 4-2:	Articulaciones húmero-cubital; humero- Radial y radio cubital proximal ..	10
Figura 5-2:	Punto de venta físico	11
Figura 6-2:	Rehabilitación física con robots	12
Figura 7-2:	Sistema Ciberfísico.....	12
Figura 8-2:	Desarrollo Interfaz Gráfica - LabVIEW.....	13
Figura 9-2:	Diseño mecánico - SolidWorks.....	14
Figura 10-2:	Python logo.	14
Figura 11-2:	Raspberry Pi3.	15
Figura 12-2:	Modelos de Arduinos	16
Figura 13-2:	Estructura interna del motor paso a paso.....	16
Figura 14-2:	Modelos Drivers para motores paso a paso NEMA 17	17
Figura 15-2:	Circuito electrónico de una fuente de alimentación	17
Figura 1-3:	Etapas de ejecución del trabajo	18
Figura 2-3:	Flexoextensión del codo.....	19
Figura 3-3:	Pronosupinación del codo	20
Figura 4-3:	Flexoextensión muñeca	20
Figura 5-3:	Prototipo propuesto.	22
Figura 6-3:	Modelación de piezas auxiliares para el rehabilitador en SolidWorks.....	23
Figura 7-3:	Modelación de piezas base para el rehabilitador en SolidWorks.	24
Figura 8-3:	Acotación de la pieza base del prototipo rehabilitador.....	25
Figura 9-3:	Diagrama explosionado del rehabilitador.....	26
Figura 10-3:	Cotas del ensamble total.....	28
Figura 11-3:	Modelo tridimensional ensamblado del rehabilitador	29
Figura 12-3:	Vista de la pieza resorte en SolidWorks.....	30
Figura 13-3:	Vista de la pieza base principal en SolidWorks	32
Figura 14-3:	Vista de la pieza base del motor del antebrazo en SolidWorks.....	34
Figura 15-3:	Vista de la pieza base de alojamiento del antebrazo en SolidWorks.....	36
Figura 16-3:	Vista de la pieza soporte del cilindro y resorte en SolidWorks.....	38
Figura 17-3:	Vista de la pieza soporte para el giro de la muñeca en SolidWorks.....	40
Figura 18-3:	Vista de la pieza elemento de movilidad para la mano en SolidWorks.....	43

Figura 19-3:	Vista de la pieza apoyo para rehabilitación de mano en SolidWorks.....	45
Figura 20-3:	Vista de la pieza soporte de movimiento de mano	47
Figura 21-3:	Vista de la pieza brazo de movilidad del antebrazo.	49
Figura 22-3:	Ubicación de los motores en el rehabilitador	52
Figura 23-3:	NEMA 23 MODEL 23HS9430B.	53
Figura 24-3:	NEMA 23 23HD76002Y-21B.....	53
Figura 25-3:	NEMA 17 MODEL17HS13-0404S	54
Figura 26-3:	Driver DM542T.....	55
Figura 27-3:	Driver BL-TB6560-V2.0.....	56
Figura 28-3:	Raspberry Pi3 Modelo B	57
Figura 29-3:	Tarjeta Arduino UNO.....	58
Figura 30-3:	Fuente de Alimentación - Motores	59
Figura 31-3:	Diagrama de Conexión de la Raspberry Pi3 con el Arduino UNO	59
Figura 32-3:	Diagrama de Conexión Arduino - Driver BL-TB6560-V2.0 – NEMA 17... ..	60
Figura 33-3:	Diagrama de Conexión Arduino - Driver DM542T – NEMA 23	60
Figura 34-3:	Diagrama eléctrico del prototipo	61
Figura 35-3:	Actualización paquetes en la Raspberry.....	62
Figura 36-3:	Instalación paquete de Arduino.	62
Figura 37-3:	Activación del puerto serial.....	63
Figura 38-3:	Instalación paquete de Arduino.	63
Figura 39-3:	Creación de cuenta en la plataforma remot3 it.	64
Figura 40-3:	Inserción de un nuevo dispositivo en la plataforma remot3 it.....	64
Figura 41-3:	Definición del tipo de dispositivo a agregarse en la plataforma remot3 it. ..	65
Figura 42-3:	Actualización de paquetes en S.O Linux Raspbian.	65
Figura 43-3:	Instalación del paquete de Remot3 it.....	66
Figura 44-3:	Formas de enlazar el dispositivo a la plataforma remot3 it.	66
Figura 45-3:	Ingreso a la plataforma remot3 it desde la Raspberry –	67
Figura 46-3:	Identificación del dispositivo a la plataforma IoT.....	67
Figura 47-3:	Levantamiento de un servicio en remot3 it	67
Figura 48-3:	Definición del servicio	68
Figura 49-3:	Asignación de nombre al servicio activado.....	68
Figura 50-3:	Verificación del dispositivo agregado a la cuenta de remot3 it.....	68
Figura 51-3:	Declaración de variables en el IDE de Arduino	69
Figura 52-3:	Función setup().....	70
Figura 53-3:	Decodificación de la información adquirida	70
Figura 54-3:	Selección de acciones – Estructura CASE	71

Figura 55-3:	Funciones para calibración.....	72
Figura 56-3:	Sentencias para ejecución de la maniobra del motor A.....	73
Figura 57-3:	Impresión de datos para monitoreo de actuadores.....	74
Figura 58-3:	Importación de módulos.....	75
Figura 59-3:	Declaración de variables y habilitación del puerto serial.....	75
Figura 60-3:	Modo Calibración motor A.....	76
Figura 61-3:	Modo Calibración motor B.....	76
Figura 62-3:	Modo Calibración motor C.....	77
Figura 63-3:	Modo Maniobra – Motores A, B, C.....	78
Figura 64-3:	Validación de información.....	78
Figura 65-3:	Validación de información “ángulo”, “repeticiones”.....	79
Figura 66-3:	Impresión en el Puerto Serial - Información solo para calibración.....	79
Figura 67-3:	Impresión en el Puerto Serial - Información solo para maniobra.....	80
Figura 68-3:	Hilo de programación – Recolección y Gestión de información.....	80
Figura 69-3:	Bloque de autenticación y conexión a la plataforma IoT.....	81
Figura 70-3:	Envío de datos a la plataforma IoT – Remot3 it.....	81
Figura 71-3:	Funciones bloqueo/desbloqueo de los procesos de calibración y maniobra.....	82
Figura 72-3:	Creación de pantalla - medidas.....	82
Figura 73-3:	Asignación etiquetas a recursos de la interfaz gráfica.....	83
Figura 74-3:	Asignación etiquetas y variables en la interfaz gráfica.....	84
Figura 75-3:	Asignación de etiquetas de valores de ingreso en la interfaz gráfica.....	84
Figura 76-3:	Asignación de etiquetas a valores ingreso en la interfaz gráfica-Maniobra.....	85
Figura 77-3:	Asignación etiquetas valores de ingreso en interfaz gráfica-Conectividad.....	85
Figura 78-3:	Creación botones para el manejo de motores en el proceso de calibración.....	86
Figura 79-3:	Creación de botón para inicio de conexión a la plataforma IoT.....	87
Figura 80-3:	Función que permite cerrar el bucle de la interfaz gráfica.....	87
Figura 81-3:	Creación Proyecto y VI.....	88
Figura 82-3:	Bloque de programación para conexión TCP.....	88
Figura 83-3:	Interfaz gráfica para verificación de conexión.....	89
Figura 84-3:	Boque programación para el procesamiento de cadena de texto recibida.....	89
Figura 85-3:	Sentencias de ejecución para cada espacio de la estructura CASE.....	90
Figura 86-3:	Ingreso a la información del servicio TCP en Remote it.....	90
Figura 87-3:	Obtención, carga de dirección TCP y puerto.....	91
Figura 88-3:	Modelo 3D de brazo.....	91
Figura 89-3:	Habilitación complementos en SolidWorks.....	92
Figura 90-3:	Asignación de un Motor Rotatorio.....	93

Figura 91-3:	Configuración de recursos para el enlace de las plataformas	94
Figura 92-3:	Configuración flujo de información de LabVIEW hacia el ensamble	95
Figura 93-3:	Piezas construidas.....	96
Figura 94-3:	Ensamble mecánico de las piezas.....	97
Figura 95-3:	Acople de actuadores eléctricos	97
Figura 96-3:	Instalación de drives y motores de paso	98
Figura 97-3:	Instalación dispositivos encargados de la gestión de las señales de control.	99
Figura 98-3:	Stress del resorte.....	100
Figura 99-3:	Desplazamiento por esfuerzo del resorte.....	101
Figura 100-3:	Stress de la base principal	102
Figura 101-3:	Desplazamiento por esfuerzo de la base principal.....	102
Figura 102-3:	Stress de la base del motor del antebrazo	103
Figura 103-3:	Desplazamiento por esfuerzo de la base del motor del antebrazo	104
Figura 104-3:	Desplazamiento por esfuerzo de la base principal.....	104
Figura 105-3:	Desplazamiento por esfuerzo de la base de alojamiento del antebrazo.....	105
Figura 106-3:	Stress del soporte del cilindro y resorte.....	106
Figura 107-3:	Desplazamiento por esfuerzo del soporte del cilindro y resorte.....	106
Figura 108-3:	Stress del soporte del soporte para el giro de la muñeca	107
Figura 109-3:	Desplazamiento por esfuerzo del soporte para el giro de la muñeca.....	108
Figura 110-3:	Stress del soporte del elemento de movilidad para la mano.....	109
Figura 111-3:	Desplazamiento por esfuerzo del elemento de movilidad para la mano	109
Figura 112-3:	Stress del elemento de apoyo para rehabilitación de mano	110
Figura 113-3:	Desplazamiento esfuerzo del elemento apoyo para rehabilitación mano...	111
Figura 114-3:	Stress del soporte de movimiento de mano	112
Figura 115-3:	Desplazamiento por esfuerzo del soporte de movimiento de mano	112
Figura 116-3:	Stress del brazo de movilidad del antebrazo	113
Figura 117-3:	Desplazamiento por esfuerzo de la movilidad del antebrazo	114
Figura 118-3:	Regulación de la base principal para ajuste de la altura del equipo	115
Figura 119-3:	Regulación del apoyo del brazo	116
Figura 120-3:	Sistema Electromecánico	117
Figura 121-3:	Resultado – Interfaz gráfica	118
Figura 122-3:	Resultado - Conectividad de la Raspberry Pi con Remote it.....	119
Figura 123-3:	Enlace LabVIE y el dispositivo Raspberry Pi	119
Figura 124-3:	Resultado – Ejecución de Calibración del equipo.....	120
Figura 125-3:	Resultado – Recepción remota de datos de la maniobra en LabVIEW.....	121
Figura 126-3:	Ejecución monitoreo remoto del rehabilitador-Flexoextensión de muñeca	122
Figura 127-3:	Resultado – Ejecución del Rehabilitador Flexoextensión del codo	123

Figura 128-3: Resultado – Ejecución del Rehabilitador Pronosupinación.....	123
Figura 129-3: Resultado – Ejecución del Rehabilitador Flexoextensión de muñeca	124

ÍNDICE DE ANEXOS

ANEXO A: LÁMINAS – PLANOS DE PIEZAS

ANEXO B: LÁMINAS - ENSAMBLES

ANEXO C: LÁMINA – DIAGRAMA DE CONEXIONES ELÉCTRICAS

ANEXO D: PROGRAMACIÓN ARDUINO UNO

ANEXO E: PROGRAMACIÓN PYTHON

RESUMEN

El trabajo describe el desarrollo para la construcción de un prototipo para rehabilitación pos traumatismos de codo con el fin de integrar sistemas de control, monitoreo y el internet de las cosas. Se determinó que la flexoextensión del codo, pronosupinación y flexoextensión de la muñeca son movimientos considerados en un protocolo de rehabilitación tras traumatismos de codo, convirtiéndose en requerimientos para el diseño del equipo electromecánico. Se modeló el prototipo haciendo uso de una herramienta CAD, para el caso de estudio se utilizó SolidWorks y se lo validó mediante un análisis estático. Se emplearon como elementos hardware del sistema una Raspberry Pi3 para dar: soporte a la aplicación informática, enlace a la plataforma IoT Remote it y emisora de instrucciones hacia un microcontrolador albergado en la plataforma Arduino UNO, el mismo que se encarga de procesar dicha información reciba para gestionar señales de control hacia los drivers de los actuadores eléctricos, en este caso dos motores NEMA 23 y un NEMA 17 empleados para gobernar el movimiento del prototipo. Se empleó Python para la implementación de la interfaz gráfica y carga de los algoritmos de control. Se consiguió integrar LabVIEW y SolidWorks en una interfaz para realizar el monitoreo remoto del rehabilitador. LabVIEW fue empleado para adquirir información de la plataforma IoT Remote it y fuente de información para un modelo de brazo en 3D montado el SolidWorks como recurso animado para dar seguimiento a los movimientos que realiza el rehabilitador. Implementado y probado el prototipo, se obtuvo una correcta funcionalidad en la ejecución de movimientos, configuración desde la interfaz gráfica y monitoreo remoto, en cuanto a la postura del paciente el especialista evaluó el prototipo calificándola como aceptable.

Palabras clave: <ELECTROMECAÁNICA>, <REHABILITACIÓN DE CODO>, <INTERNET DE LAS COSAS>, <PROGRAMACIÓN GRÁFICA>, <DISEÑO ASISTIDO POR COMPUTADOR>, <MONITOREO REMOTO>.



Firmado electrónicamente por:
JHONATAN RODRIGO
PARREÑO UQUILLAS

SUMMARY

This study describes the development process for constructing a prototype for elbow post-trauma rehabilitation in order to integrate control systems, monitoring and the internet of things. It was determined that elbow flexion and extension and wrist pronosupination and flexion are movements that make up rehabilitation protocol for elbow trauma and are thus requirements for the design of electromechanical equipment. The prototype was modeled using a CAD tool, SolidWorks was used for the case study and it was validated through static analysis. A Raspberry Pi3 was used as hardware elements of the system to provide support for the IT application, link to the IoT Remote IT platform and issue instructions to a microcontroller hosted on the Arduino UNO platform, which in turn was responsible for processing said received information to manage control signals to the electric actuator drivers, in this case two NEMA 23 motors and a NEMA 17 used to govern the prototype's movement. Python was used to implement the graphical interface and load the control algorithms. It was possible to integrate LabVIEW and SolidWorks in an interface for remote monitoring of the rehabilitator. LabVIEW was used to acquire information from the IoT Remote IT platform and as a source of information for a 3D arm model mounted on SolidWorks as an animated resource to track the movements performed by the rehabilitator. Once the prototype was implemented and tested, correct functionality was obtained in the execution of movements, configuration from the graphic interface and remote monitoring. The specialist evaluated the prototype in terms of the patient's position, rating it as acceptable.

Keywords: <ELECTROMECHANICS>, <ELBOW REHABILITATION>, <INTERNET OF THINGS>, <GRAPHIC PROGRAMMING>, <COMPUTER-AIDED DESIGN>, <REMOTE MONITORING>

1. INTRODUCCIÓN

En la actualidad el auge de la industria 4.0 ha hecho posible el monitoreo y control remoto de sistemas industriales, médicos, domiciliarios entre otros gracias a la tecnología denominada: Internet de las cosas (IoT). El impacto que Internet ha tenido sobre la educación, la comunicación, las empresas, la ciencia, el gobierno y la humanidad, muestran claramente que es una de las creaciones más importantes y poderosas de toda la historia de la humanidad.

Está compuesto por una colección dispersa de redes diferentes y con distintos fines. Por ejemplo, los automóviles actuales tienen múltiples redes para controlar el funcionamiento del motor, las medidas de seguridad, los sistemas de comunicación y así sucesivamente. De forma similar, los edificios comerciales y residenciales tienen distintos sistemas de control para la calefacción, la ventilación y el aire acondicionado, la telefonía, la seguridad y la iluminación. A medida que IoT evoluciona, estas redes y muchas otras estarán conectadas con la incorporación de capacidades de seguridad, análisis y administración. Esta inclusión permitirá que el IoT sea una herramienta aún más poderosa. (Evans, 2011, pp. 2-5)

Las recientes innovaciones de la electrónica, la informática y las tecnologías de la información y la comunicación (TIC) han propiciado, de una parte, un crecimiento exponencial de la capacidad material de procesamiento de los sistemas de tratamiento de información y, de otra parte, han permitido la miniaturización de microprocesadores que son empleados como sensores para la captación de datos. Todo acompañado, en paralelo, de una gradual reducción sustancial de los costes de fabricación y comercialización de dicho hardware. Estas invenciones, además, se construyen y amplifican mutuamente en una convergencia de tecnologías a través de los mundos físico y digital. (Barrio Andrés, 2018, pp. 17-21)

En el campo médico el desarrollo de esta tecnología está proyectada a incrementar la cobertura de asistencia que ofrecen los fisioterapeutas y hacer más eficiente el uso de la medicina a distancia, orientándose a la descarga del expediente médico del paciente con facilidad y en tiempo real omitiendo la presencia física del mismo para controlar y monitorear la mejora de este. Por su parte, la asistencia de pacientes no es un tema ajeno a IoT, vemos como, comercialmente se presentan productos de hardware y servicios que dependen de su conectividad a internet. Para dar soporte a los pacientes involucrados, normalmente la palabra asistencia involucra un conjunto de personas que va más allá del paciente, involucrando a sus cuidadores y personal de salud, en varios de los trabajos analizados se hace evidente la importancia de mantener intercomunicados a estos actores del proceso asistencial y una forma intuitiva de lograrlos es valiéndose de la red de redes. (Murillo, 2015, p. 30)

1.1 Planteamiento del Problema

Según las estadísticas la patología del codo ocupa un alto porcentaje de lesiones en el ámbito deportivo y hasta un 3% en la población general, siendo la epicondilitis (dolor en el codo) la afección más frecuente. (Club Cortijo del Aire, 2013)

El trabajo pionero de investigadores como Krebs, Volpe y Hogan, del Massachusetts Institute of Technology y del Burke Medical Research Institute, ha demostrado que el robot MIT-MANUS reduce de manera eficaz el tiempo de recuperación motriz del paciente al realizar los ejercicios sistemáticos apropiados para la rehabilitación del hombro y el codo. (Krebs et al., 2004)

En Colombia en la universidad de Santo Tomás se realizó el diseño mecatrónico de un robot exoesqueleto de extremidad superior para rehabilitación de personas con discapacidad parcial en el codo con el fin de replicar las condiciones biomecánicas de un proceso eficiente de rehabilitación asistida de la flexo-extensión del codo y de esta manera poder evaluar el desempeño de este tipo de dispositivos en labor de asistencia directa o remota en la kinesioterapia de esta articulación, en el cual se obtuvo el modelo cinemático y dinámico del exoesqueleto, y en el proceso de ello se logró conocer la tribología típica de los dispositivos robóticos, las consecuencias en el amortiguamiento de la estructura a diferentes viscosidades dinámicas y las consecuencias de la selección del tipo de trayectoria. (CELEDÓN FLÓREZ, 2016, p. 66).

En este contexto Ecuador no se ha quedado atrás, pues a pesar de que los centros médicos del país no disponen de robots de última tecnología para efectuar protocolos de terapias de rehabilitación debido al elevado costo que representa su adquisición, razón por lo cual desde el sector académico en las instituciones de nivel superior de Ecuador se vienen trabajando proyectos de desarrollo en esta área, razón por la cual, como caso relevante se enuncia que en asociación de la Escuela Politécnica Nacional, la Escuela Superior Politécnica de Chimborazo, Escuela Superior Politécnica del Ejercito, la Universidad San Francisco, la Universidad Central, la Universidad de las Américas generaron un espacio de difusión de avances en el campo de la Bioingeniería y Sistemas Inteligentes de Rehabilitación mediante un congreso internacional, en cuyas memorias reposa información sobre sistemas inteligentes de rehabilitación, prótesis robóticas de miembros superiores e inferiores basados en procesamiento de señales electromiográficas, electroencefalográficas, modelos matemáticos para definición de movimientos corporales, desarrollo de equipo asistente para movilización de personas con discapacidad bipedestor, entre otros. (CIBSIR, 2017).

1.2 Justificación del proyecto

La revolución tecnológica mediante la automatización de máquinas no solo ha ayudado en el sector industrial, si no también se ha dado uso en la rehabilitación y la fisioterapia y se ha ido perfeccionando en los últimos años. En la actualidad a nivel mundial se conocen muchos prototipos robóticos incluso algunas empresas ya ofertan estos sistemas robóticos como productos de acceso público, no todos estos robots son humanoides, pero se encuentran dotados de inteligencia y automatismos aplicados. (AMERIKE, 2019)

La robótica vinculada a elementos electromecánicos es una estrategia que se la viene desarrollando desde los años ochenta siendo que actualmente se ha desarrollado a pasos gigantescos por la accesibilidad a recursos tecnológicos de bajo costo que permiten el prototipado de modelos, y son varios trabajos que han ido enfocados a su inserción en la medicina a través de los usos tecnológicos biomédicos. (AMERIKE, 2019)

En rehabilitación, la terapia asistida por robots ha generado varios beneficios desde la creación de miembros artificiales, robots de soporte y robots que brindan asistencia médica personal en los hospitales. (NIBIB, 2019)

El desarrollo del presente trabajo consiste en el diseño e implementación de un prototipo electromecánico para la rehabilitación de codo basado en la integración de sistemas de control, monitoreo y el internet de las cosas con la finalidad de dar un seguimiento verídico al proceso de rehabilitación en el domicilio. La integración de software aumenta la potencialidad de un sistema al sumarse las utilidades que proporcionan cada uno de estos, LabVIEW potenciado para la adquisición de datos y con herramientas IoT y SolidWorks herramientas potenciales para modelado CAD.

Se plantea la generación de un modelo original de rehabilitador electromecánico para el codo. Se establece como requerimiento que los movimientos que incite el rehabilitador al paciente con la afección, se vinculen directamente con la flexo extensión - pronosupinación del codo y flexo extensión de la muñeca, definidos como necesarios en el proceso de rehabilitación del caso de estudio, razón por la cual el propósito es que el paciente pueda además contar con el prototipo en su domicilio y continúe con el proceso de rehabilitación sin la necesidad de permanecer en el centro médico, ni requiera la presencia del galeno, manteniendo un monitoreo continuo para validar su recuperación en base al cumplimiento de la rehabilitación, el paciente al ejecutar su rehabilitación en casa por medio de la activación del prototipo y su conexión a un punto con acceso a internet permitirá que éste inicie el envío de información a la plataforma de soporte IoT, que paralelamente será extraída desde el punto de conexión remota (centro médico) para el monitoreo y registro de información. El sistema planteado resulta de utilidad para evaluar la

eficiencia de los protocolos de rehabilitación planteados por los especialistas y el seguimiento al cumplimiento de los mismos por parte de los pacientes, mejorando la recuperación y calidad de vida de pacientes con determinada patología.

1.3 Objetivos

1.3.1 Objetivo General

Construir un prototipo electromecánico para rehabilitación tras traumatismos de codo basado en la integración de sistemas de control, monitoreo y el Internet de las cosas.

1.3.2 Objetivos Específicos

Definir los requerimientos necesarios para el desarrollo del proceso de rehabilitación tras traumatismos de codo por medio de revisión bibliográfica y asesoría de un especialista.

Dimensionar y seleccionar el hardware y software necesario para cumplir con los requerimientos planteados.

Implementar los sistemas de control, monitoreo y configuración local del prototipo.

Integrar LabVIEW y SolidWorks en una interfaz para realizar el monitoreo remoto del rehabilitador con adquisición de información con un enlace al prototipo mediante el internet de las cosas.

Validar el prototipo electromecánico para la rehabilitación de codo mediante pruebas de funcionalidad.

1.4 Alcance

El trabajo de titulación está encaminado a la construcción de un prototipo electromecánico para ser utilizado en la rehabilitación después de un traumatismo en el codo, por medio de la integración de sistemas de control y monitoreo enlazados mediante la tecnología del Internet de las cosas (IOT). La diversidad de trabajos registrados en el campo de la rehabilitación promueve el interés de aportar por medio de la ejecución de este proyecto con la construcción de un modelo de prototipo original de equipo electromecánico que ayude en el caso específico de rehabilitación

de código y el desarrollo de un sistema que relacione la inserción de la tecnología del internet de las cosas como medio de innovación tecnológica para realizar un seguimiento total al proceso de rehabilitación dentro y fuera del centro clínico de rehabilitación, permitiendo ser accesible para la población vulnerable que padece esta lesión, además otro objetivo primordial de la generación de este proyecto es aumentar la cobertura de servicio que ofrecen los galenos de la salud haciendo más eficiente el uso de la medicina a distancia, puesto que este sistema permitirá monitorear y controlar la ejecución de la rutina de rehabilitación del paciente, motivo por el cual la ejecución de este proyecto pretende mejorar la calidad de vida de los pacientes aportando un sistema de fácil uso en el manejo de la información de datos generados no limitando su principio de funcionamiento a otro sistema de rehabilitación. El sistema de monitoreo remoto tendrá un atractivo especial, porque se integrará en una interfaz gráfica de las plataformas de SolidWorks y LabVIEW generando una animación de movimiento del rehabilitador con la información extraída del punto en internet (IoT) al que se estará constantemente enviando datos, el cual surgirá de la visita del paciente al especialista, donde éste último será el que determine el protocolo de rehabilitación (frecuencia), de acuerdo a la afección, las mismas que podrán ser programadas por los usuarios en el sistema local del prototipo a través de la interfaz gráfica de alto nivel.

2. MARCO TEÓRICO

En la actualidad es muy amplio el número de dispositivos que son susceptibles de conexión a Internet con capacidad de reconocer su entorno, de tomar decisiones inteligentes y de proceder con base a la información que reciben, para realizar diversas aplicaciones, en varias áreas de la cotidianidad siendo así la presencia de conectividad en cada entorno.

2.1 Internet de las cosas (IoT)

Internet de las cosas (IoT) es la red de objetos físicos que se conectan a Internet usando diversas tecnologías y que tienen capacidades de conexión e interacción con el entorno, capacidades que les permiten tomar decisiones y comunicarse con el mundo. (López i Seuba, 2019, p. 21). La figura 1-2 representa la incidencia del IoT en todo el mundo.



Figura 1-2: Internet de las cosas.

Fuente: <https://n9.cl/t8eb>

El internet de las cosas (IoT) en ocasiones se lo llama “Internet de los objetos” (Evans, 2011); nos da a conocer el impacto que lleva en nuestra era el internet sobre la educación, la comunicación, las empresas y la humanidad. Es una de las mayores creaciones de toda la historia.

Los grandes avances tecnológicos de las últimas décadas en las Tecnologías de la Información y la comunicación han permitido que se tenga acceso a medios cada vez más poderosos como:

- Salud.
- Transporte.
- Agricultura.
- Servicios de electricidad.
- Fabricación.
- Compras.

Cada aplicación mencionada tiene como fin la mejora virtual mediante la aplicación e implementación de la tecnología, facilitando el uso de servicios básicos de la vida cotidiana.

Por dar un ejemplo, la salud se lo puede mejorar con los aparatos electrónicos y robóticos que ayudan a la atención y calidad de vida de los pacientes con alguna lesión.

En educación la vinculación de aulas virtuales de aprendizaje es eficiente y accesible, el servicio de bibliotecas virtuales.

2.1.2 Componentes de un sistema IoT

IoT hace referencia al conjunto de objetos físicos que tiene una inteligencia e identidad propia para integrarse e interactuar de manera independiente en Internet con otro dispositivo o usuario. (Buyya & Vahid, 2016)

IoT se ha convertido en uno de los términos más populares de los últimos años y muchas grandes empresas están apostando e invirtiendo tanto en el desarrollo de dispositivos inteligentes con múltiples aplicaciones como en el resto de tecnologías que demanda este tipo de soluciones (comunicaciones de bajo consumo, protocolos de seguridad, plataformas de gestión, etc.).

Existen tres componentes clave que habitualmente se mencionan en la arquitectura IoT.

- **Dispositivos IoT:** Dispositivos que se pueden conectar por cable o de forma inalámbrica a una red más amplia.
- **Redes:** De forma parecida a los routers domésticos, las redes o las puertas de enlace conectan varios dispositivos IoT a la nube.

La figura 4-2, muestras las articulaciones del codo.



Figura 4-2: Articulaciones húmero-cubital; humero-Radial y radio cubital proximal.

Fuente: ANGULO ET ALL, 2011

2.2.1 Fracturas de codo

La recuperación de una función satisfactoria y sin dolor depende de una reconstrucción anatómica de las superficies articulares, así como una fijación lo suficientemente estable para iniciar una rehabilitación temprana. Aunque estos objetivos son obvios para el cirujano ortopédico, son difíciles de lograr, especialmente al encontrarse con un hueso osteoporótico o multifragmentado. El área con mayor dificultad de tratar en el codo es el húmero distal, ya que su compleja anatomía lo ha convertido en uno de los retos más grandes para el cirujano. La evolución de las técnicas en el manejo de las fracturas ha facilitado enormemente la reconstrucción de esta zona; se cuenta con una gran cantidad de abordajes publicados, así como con la evolución en los implantes, especialmente diseñados para hueso de mala calidad o fracturas multifragmentadas. Es así como se ha pasado de los tratamientos conservadores a base de yesos o tracciones, normalmente con malos resultados, a técnicas con clavillos y fijadores externos o a la técnica más aceptada, que consiste en el tratamiento con doble placa. (Echevarría Zuno, 2013, p. 276), como se muestra en la figura 5-2.



Figura 5-2: Punto de venta físico

Fuente: (Slideshare, 2012)

2.2.2 La rehabilitación física

“Rehabilitación es el proceso y el resultado de rehabilitar. Este verbo se refiere a volver a habilitar, restablecer o recuperar algo. Lo físico, por su parte, se asocia a lo corporal o lo material.” (Pérez & Gardey, 2015)

La idea de rehabilitación física está vinculada al tratamiento que desarrolla una persona para recobrar la condición o el estado que perdió a causa de una enfermedad u otro tipo de trastorno de salud. La rehabilitación física, en pocas palabras, apunta a la funcionalidad corporal. Las tareas de rehabilitación pretenden que el individuo mejore su movilidad y sus habilidades físicas a partir de ejercicios, masajes y otras técnicas.

La rehabilitación tiene como finalidad lograr que el paciente realice sus funciones y actividades cotidianas mediante un conjunto de procedimientos en los que el paciente va mejorando la movilidad progresivamente e incrementando la fuerza de su articulación, músculo o tejido afectado ya sea por un accidente o por una enfermedad. Con conocimiento de que la robótica terapéutica permite acelerar el proceso de rehabilitación. (Moya & Vásquez, 2016)

La figura 6-2 muestra un ejemplo de rehabilitación asistida por equipamiento electromecánico y robótico.



Figura 6-2: Rehabilitación física con robots

Fuente: <https://n9.cl/qij9f>

2.2.3 Sistemas ciberfísicos

Los sistemas ciberfísicos abarcan varias áreas de aplicación como se aprecia en la figura 7-2

Cyber Physical System - CPS

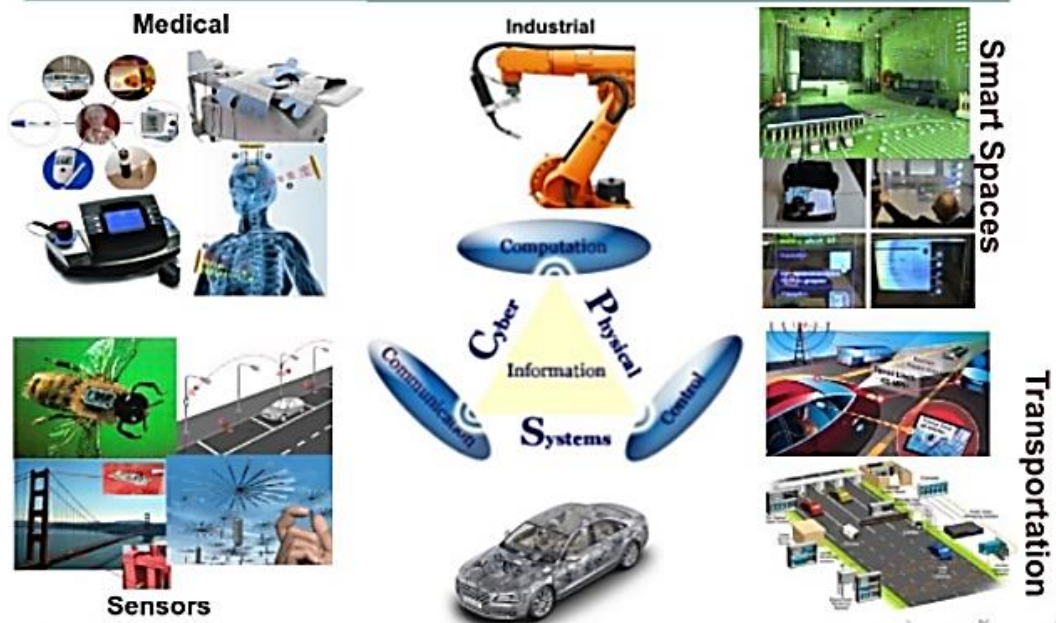


Figura 7-2: Sistema Ciberfísico

Fuente: <https://bit.ly/39vo519>

Los sistemas ciberfísicos (CPS en su acrónimo en inglés), son sistemas robóticos inteligentes vinculados a la IoT. Es una combinación de sistemas en red, robots e inteligencia artificial que interactúa con el mundo físico, presentando como características:

- La capacidad de adquirir autonomía mediante sensores y/o mediante el intercambio de datos con su entorno (interconectividad) y el análisis de dichos datos.
- La capacidad de aprender a través de la experiencia y la interacción.
- La forma del soporte físico del robot.
- La capacidad de adaptar su comportamiento y acciones al entorno. (Llaneza González, 2018, pp. 31-32)

2.3 Plataforma LabVIEW

Al lenguaje de programación que usa LabVIEW también se le llama lenguaje G. La mayoría de los lenguajes se basan en una programación imperativa, que es una sucesión de operaciones. Sin embargo, el lenguaje G no usa programación imperativa sino una ejecución basada en el flujo de datos (*dataflow*). Un programa en LabVIEW consiste básicamente en una serie de funciones unidas mediante cables. Los datos 'circulan' o 'fluyen' por los cables. Una función solo podrá ejecutarse cuando tenga disponibles todos los datos que le sirven como entradas. Esta forma de ejecutar un programa favorece el paralelismo y es apropiada para sistemas multiprocesador y multihilo. Es parecida al funcionamiento de los circuitos hardware e incluso al lenguaje VHDL. (Lajara y Pelegri, 2011, p. 25) La figura 8-2 representa el desarrollo de una interfaz gráfica en la plataforma de LabVIEW.

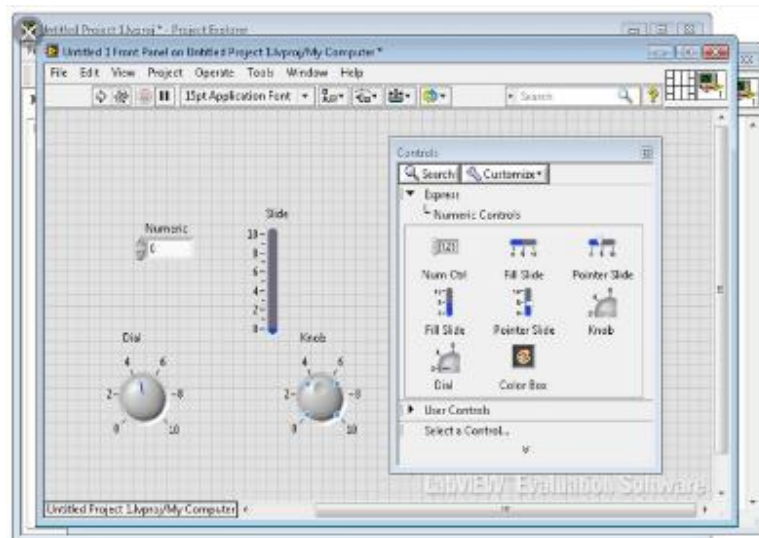


Figura 8-2: Desarrollo Interfaz Gráfica - LabVIEW

Fuente: <https://bit.ly/2ZPaqag>

2.4 SolidWorks

SolidWorks es un software CAD, se trata de un programa que permite realizar el proceso completo de diseño mecánico, desde la concepción de la idea por el diseñador a la realización de los planos técnicos necesarios para su fabricación. (Rodríguez Vidal, 2015, p. 14) La figura 9-2 representa un ejemplo de diseño mecánico de estructuras.

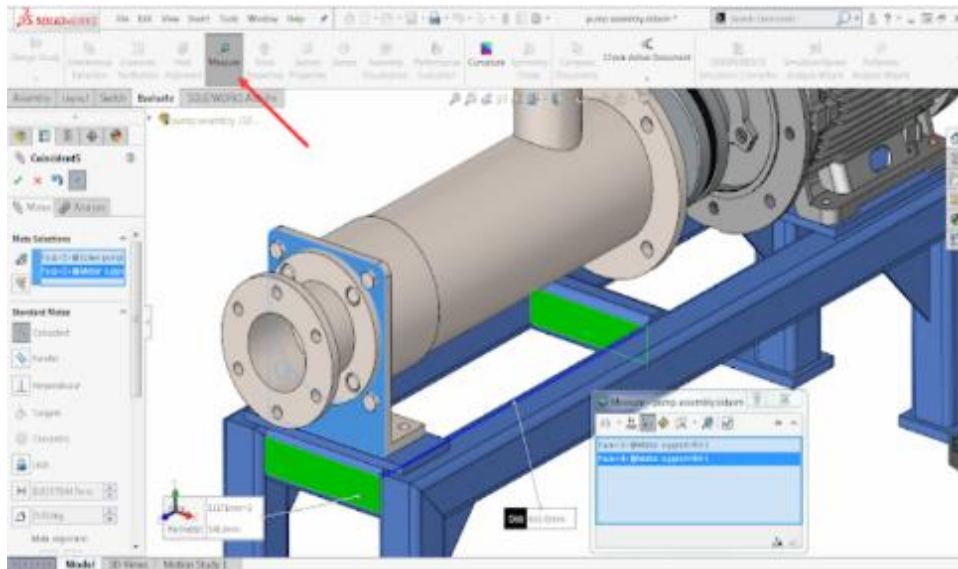


Figura 9-2: Diseño mecánico - SolidWorks.

Fuente: <https://bit.ly/2sD8k1d>

2.5 Python

La figura 10-2 muestra el logotipo del software Python, lenguaje de desarrollo de acceso libre.



Figura 10-2: Python logo.

Fuente: <https://bit.ly/2sK3Eqd>

Python es un lenguaje de "dirección" para códigos científicos escritos en otros idiomas. Sin embargo, con herramientas básicas adicionales, Python se transforma en un lenguaje de alto nivel

adecuado para el código científico y de ingeniería que a menudo es lo suficientemente rápido como para ser inmediatamente útil, pero también lo suficientemente flexible como para ser acelerado con extensiones adicionales. (Oliphant, 2007, pp. 10-20)

2.6 Tarjeta Raspberry

Raspberry Pi es un ordenador de placa reducida o (placa única) (SBC – Single Board Computer) de bajo coste (poco más de 30 Euros según modelo) desarrollado en Reino Unido por la Fundación Raspberry Pi, con el objetivo de estimular la enseñanza de ciencias de la computación en las escuelas. (López Aldea Eugenio, 2017, p. 63). En el desarrollo de esta tarjeta se han presentado varias versiones, la figura 11-2 representa el modelo Raspberry Pi3.

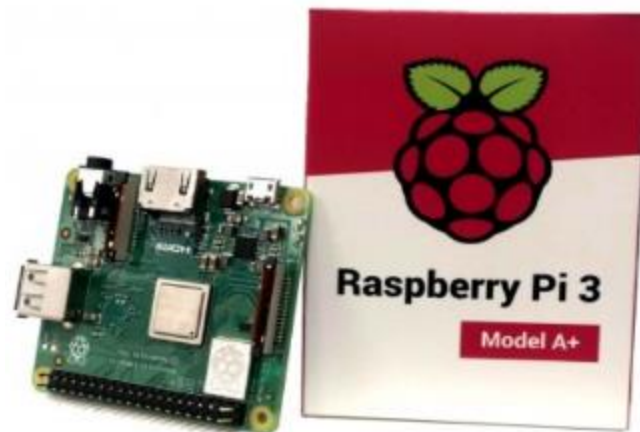


Figura 11-2: Raspberry Pi3.

Fuente: <https://bit.ly/2ZQqY1t>

2.7 Tarjeta Arduino

Arduino es una plataforma embebida cuyo elemento principal es un microcontrolador reprogramable, el mismo que maneja registros de entrada y salida permitiendo la adquisición de señales y la emisión de señales de control previo a su procesamiento mediante líneas de código programadas. Es una plataforma versátil que tiene como cualidad la fácil integración de elementos externos como sensores que pueden ser de la misma marca u otra que proporcionen señales del tipo digital y/o analógico para ser procesadas y según su evaluación por medio de interfaces de potencia poder realizar el control de actuadores sin importar la carga que manejen. (Arduino, 2020). En el mercado se encuentra disponible en varios modelos donde la selección de cualquiera de ellos dentro de una aplicación dependerá del requerimiento según el número de señales digitales o analógicas que se vaya a manejar, la figura 11-2 muestra tres de los modelos más comúnmente utilizados Arduino NANO, Arduino UNO y Arduino MEGA.



Figura 12-2: Modelos de Arduinos

Fuente: Arduino, 2020.

2.8 Motores eléctricos paso a paso

Son motores que generan movimientos discretos basados en la conversión de información digital, es decir, proporcionan desplazamientos por pasos o ángulos al cambiar el código digital proporcionado. La característica fundamental que define a estos motores es el número de pasos por vuelta o, expresado de otra manera, el ángulo por paso (CLR, 2020). La figura 13-2 representa la estructura interna del motor paso a paso.

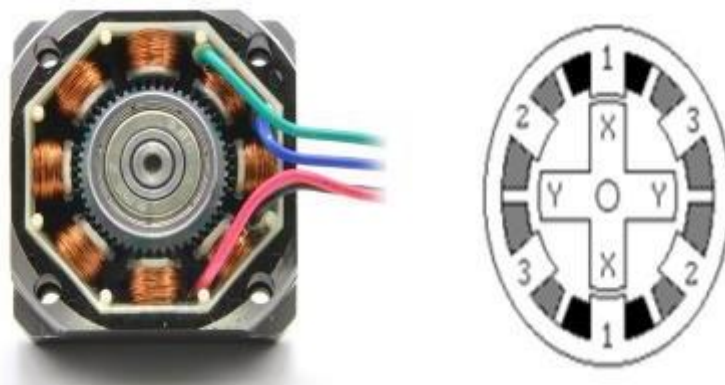


Figura 13-2: Estructura interna del motor paso a paso

Fuente: <https://url2.cl/ChEg4>

2.9 Drivers

Los drivers para motores están definidos como circuitos que permiten el manejo de motores de corriente continua. Estos controladores proporcionan la bondad de poder manejar los voltajes e intensidades suministrados al motor así como también controlar la velocidad y sentido de giro. Además, se pueden considerar como un medio de protección para salvaguardar la integridad de la electrónica de los motores limitando la corriente que circula (Hardwarelibre, 2020). La figura 14-

2 representa varios modelos de driver encontrados en el mercado, su selección depende del motor con el que se lo vaya a relacionar.



Figura 14-2: Modelos Drivers para motores paso a paso NEMA 17

Fuente: <https://url2.cl/zpvDg>

2.10 Fuente de Alimentación

La fuente de alimentación puede ser conceptualizada como un dispositivo convertidor de corriente alterna tomada de la línea de distribución domiciliaria en corriente continua o directa empleada para alimentar circuitos electrónico de equipos tales como electrodomésticos, computadoras, entre otros, proporcionan la facilidad incluso de entregar diferentes niveles de voltajes incluyen generalmente protecciones frente a eventuales inconvenientes en el suministro eléctrico, como la sobretensión. (Concepto.de, 2020). La figura 15-2 muestra la electrónica de una fuente.



Figura 15-2: Circuito electrónico de una fuente de alimentación

Fuente: <https://url2.cl/ZeCTa>

3. METODOLOGÍA

La implementación de un sistema de rehabilitación dotado de recursos tecnológicos actuales evidencia una etapa de renovación de los procesos de rehabilitación física, involucrando un trabajo en conjunto entre la parte de fisioterapia con área de la ingeniería.

Mucho se ha hablado a nivel de la industria sobre la innovación de procesos y la inserción de tecnología en los mismos, se describe actualmente como boom de la industria 4.0 donde uno de sus pilares fundamentales es el desarrollo de la tecnología IoT o internet de las cosas donde se desea lograr un mundo donde todo se encuentre conectado a la web, de éste antecedente se genera la idea de enlazar un rehabilitador electromecánico a la internet para ejecutar acciones de monitoreo y control de manera remota para un seguimiento más fino del proceso de rehabilitación física de pacientes afectados, específicamente de lesiones de codo.

Para el desarrollo de este trabajo se siguió una secuencia de pasos como se muestra en la figura 16-3 que resume la planificación de actividades para lograr alcanzar los objetivos planteados.

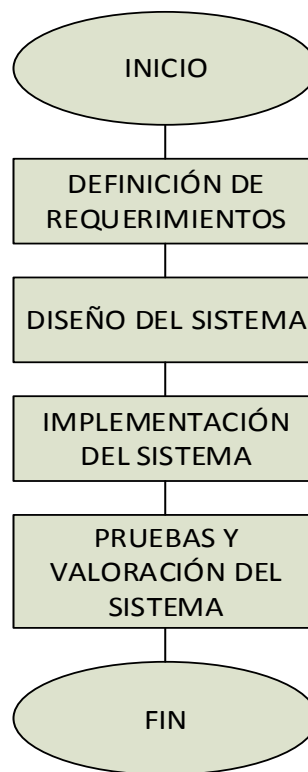


Figura 1-3: Etapas de ejecución del trabajo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Desglosando el contenido del diagrama, se manifiesta que para el desarrollo del trabajo se partió de la definición de requerimientos para el prototipo, fundamentados en dar cumplimiento a los objetivos planteados. Se propone también una etapa denominada de diseño en la que se relacionan acciones de modelado del equipo mecánico, la selección de software y hardware y su

configuración para posteriormente proceder con la etapa de implementación del proyecto y su finalización con la evaluación mediante pruebas que permitirán determinar su funcionalidad.

3.1 Determinación de requerimientos para la rehabilitación física

Para el desarrollo del trabajo se estableció por medio de la revisión bibliográfica un protocolo de rehabilitación física, donde Traumatología Hellín describe los siguientes movimientos para una rehabilitación del codo tras traumatismo.

Flexoextensión del codo: La figura 17-3 describe que “Sentado a una mesa, con el brazo apoyado sobre un cojín, estirar el codo todo lo que sea posible, intentando tocar la mesa con el dorso de la mano (A). Se mantiene en posición máxima durante 10 segundos y se descansa. Posteriormente, se dobla el codo intentando tocar el hombro con la mano (B). Mantener 10 segundos y descansar.” (Hellín, 2013)

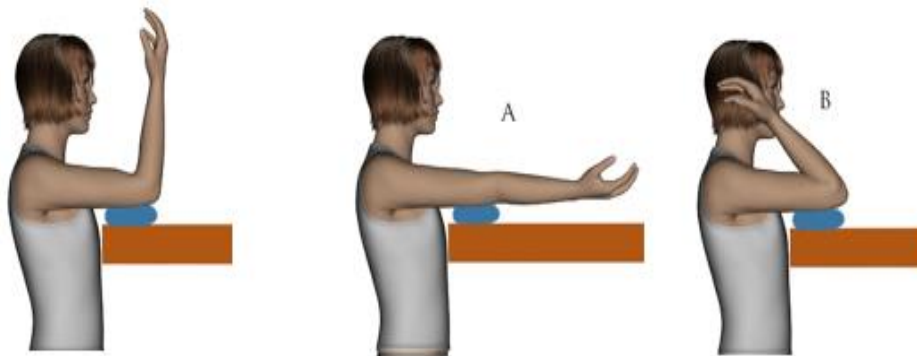


Figura 2-3: Flexoextensión del codo

Fuente: <https://n9.cl/wmz6p>

Pronosupinación: La figura 18-3 describe que “Sentado a una mesa, con el brazo apoyado sobre un cojín, se gira la mano para intentar mirar la palma (A), mantener 10 segundos y descansar. Posteriormente se gira en el otro sentido, intentando mirar el dorso de la mano (B), mantener 10 segundos y descansar.” (Hellín, 2013)

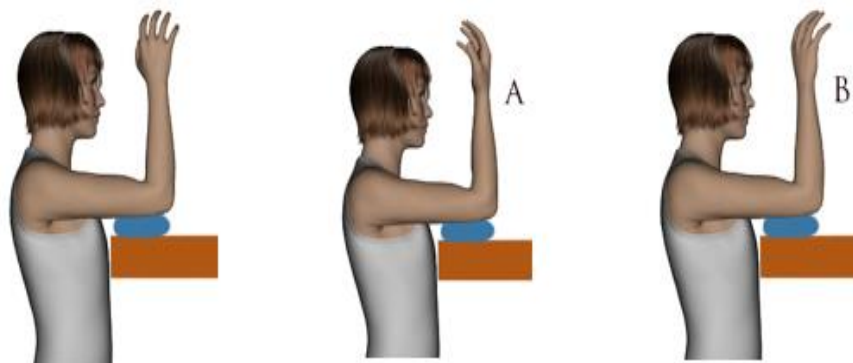


Figura 3-3: Pronosupinación del codo

Fuente: <https://url2.cl/pxcAP>

Flexoextensión de muñeca: La figura 19-3 describe que “Sentado, con el antebrazo apoyado sobre una mesa, y dejando la mano fuera, llevar la mano hacia arriba lentamente todo lo que se pueda, mantener la posición 5-10 segundos (A) y volver a la posición inicial. Posteriormente, llevar la mano hacia abajo lentamente, intentando tocar el borde de la mesa, mantener 5-10 segundos y volver a la posición inicial (B).” (Hellín, 2013)

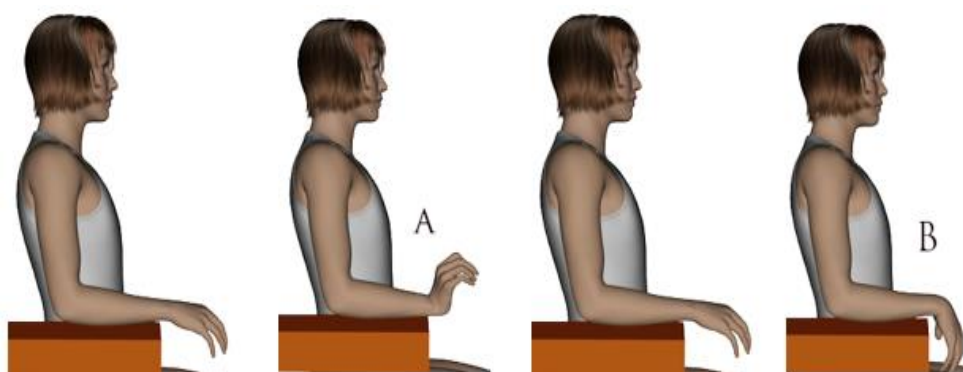


Figura 4-3: Flexoextensión muñeca

Fuente: <https://url2.cl/JZuKi>

Establecido el protocolo de rehabilitación con movimientos específicos para el proceso de recuperación del codo tras traumatismos se definió estos como requerimientos para el diseño del rehabilitador electromecánico.

3.1.1 Determinación de requerimientos de diseño mecánico

Enfocado a dar cumplimiento a la ejecución de los movimientos propuestos dentro del protocolo de rehabilitación, el desarrollo del trabajo desembocó en una nueva necesidad, la de diseñar un modelo mecánico que permita la ejecución de dichos movimientos, la determinación de una herramienta CAD (diseño asistido por computador) resultó necesaria dentro de esta actividad considerando que dicha herramienta permita la simulación y validación del diseño para en una siguiente etapa proceder a su implementación con total confianza.

3.1.2 Determinación de requerimientos del hardware

Se estableció la necesidad de determinar el hardware específico para la implementación del prototipo desde dos puntos de vista:

- Para el diseño mecánico, la utilización de actuadores eléctricos que permitan gobernar el movimiento del rehabilitador.
- Uso de un controlador y/o procesador que brinde las facilidades de albergar una aplicación informática y proporcione una interfaz de comunicación para gestionar las señales hacia los actuadores del prototipo.

3.1.3 Determinación de requerimientos del software

Definidos los actuadores para el modelo electromecánico se requirió determinar el hardware necesario para desarrollar un sistema de control que conjugado con una interfaz gráfica permitieran la manipulación, configuración y comunicación del rehabilitador.

En términos generales se planteó el desarrollo de un prototipo que disponga de un control local a través de una interfaz gráfica y mediante la tecnología IoT establecer un monitoreo remoto.

La necesidad del monitoreo remoto se complementó con la proyección animada del proceso de rehabilitación del paciente y adicionalmente una etapa de registro de información de los pacientes hacia una base de datos.

3.2 Descripción del prototipo esperado

En la figura 20-3 se muestra un esquema general del prototipo propuesto de donde se establecieron los requerimientos para su implementación.

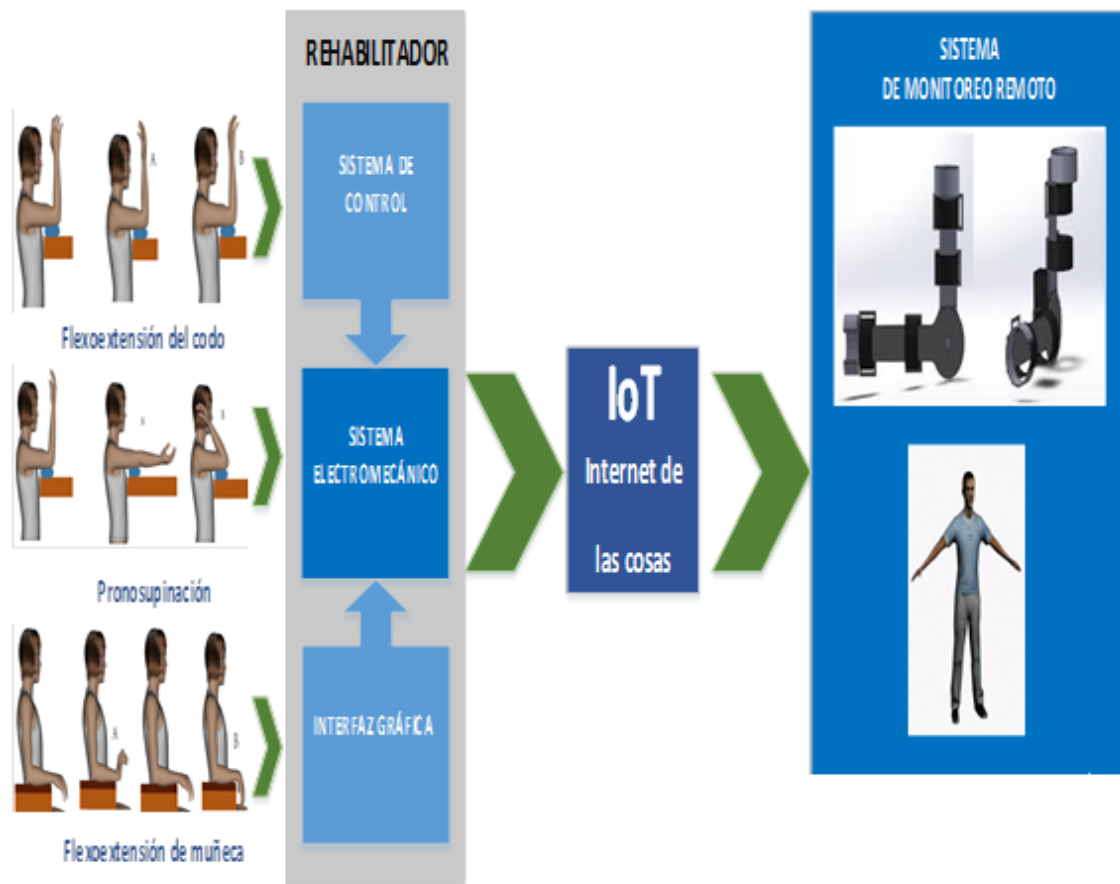


Figura 5-3: Prototipo propuesto.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

El prototipo rehabilitador dispone de:

Un equipo electromecánico que permita generar al paciente los movimientos definidos dentro del protocolo de rehabilitación de codo tras traumatismos y proporcione la facilidad de ser regulable para su adaptación a varios pacientes.

Un sistema de control que permita trabajar al rehabilitador en dos modos:

- **Modo de calibración:** Resulta necesario para posicionar los motores en sus puntos de inicio de trabajo, lo que se denomina el encerar el equipo, debido a que la contextura y situación de cada paciente es diferente.
- **Modos de maniobra:** Este modo hace referencia a la ejecución ya de actividades específicas en las que se pueda ejercitar al paciente en rangos de movimiento establecidos y en un número de ejecuciones determinadas.

Un monitoreo remoto basado en el uso de la tecnología IoT.

Un sistema que cumpla el ciclo de generación de información en el prototipo rehabilitador, escritura de la información en la nube (plataforma IoT) y su extracción en un punto remoto. El acceso remoto cuenta con un entorno amigable, es decir, con una interfaz animada que permita conocer el estado del rehabilitador.

3.3 Diseño del prototipo rehabilitador

Planteados los requerimientos del prototipo se procedió a la modelación del equipo mecánico y la determinación de los elementos software y hardware necesarios para su implementación.

3.3.1 Diseño Mecánico - Modelación CAD

Para el modelamiento del equipo rehabilitador se empleó la herramienta CAD - SolidWorks por sus prestaciones al momento de crear diseños tridimensionales y su evaluación mediante análisis estructurales.

Con la finalidad de plantear un modelo original y funcional acorde a los requerimientos planteados, se inició el diseño con un enfoque mental donde la idea de los autores poco a poco se fue plasmando en el diseño. Es así que para la mejor corrección o adaptación a los cambios que se iban generando en el modelado se optó por realizar el diseño por piezas, para su posterior ensamble. La figura 21-3 muestra algunas de las piezas consideradas auxiliares dentro del diseño del equipo por ser pequeñas y de enlace a otras piezas.

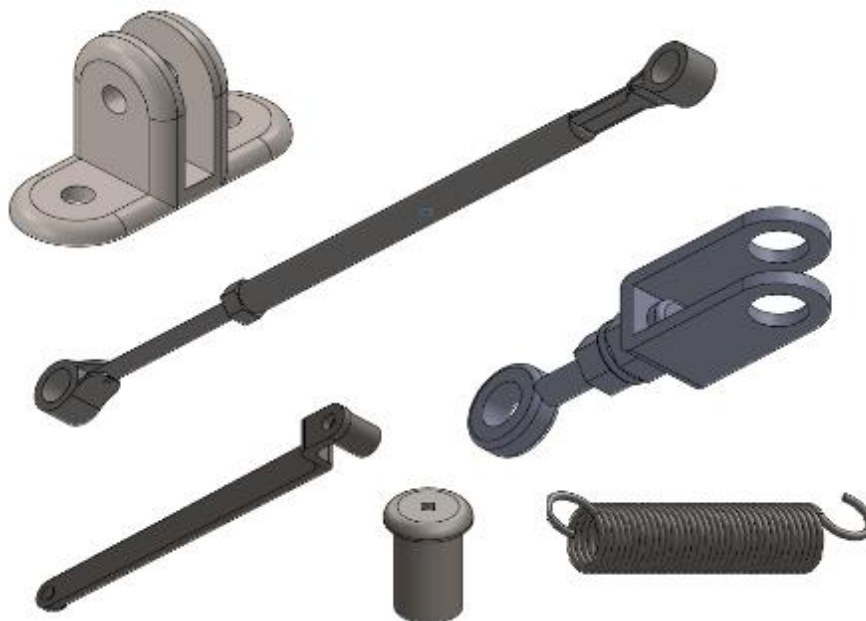


Figura 6-3: Modelación de piezas auxiliares para el rehabilitador en SolidWorks.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 22-3 muestra el diseño de piezas más grandes para en su ensamble posterior dar soporte a otros elementos.

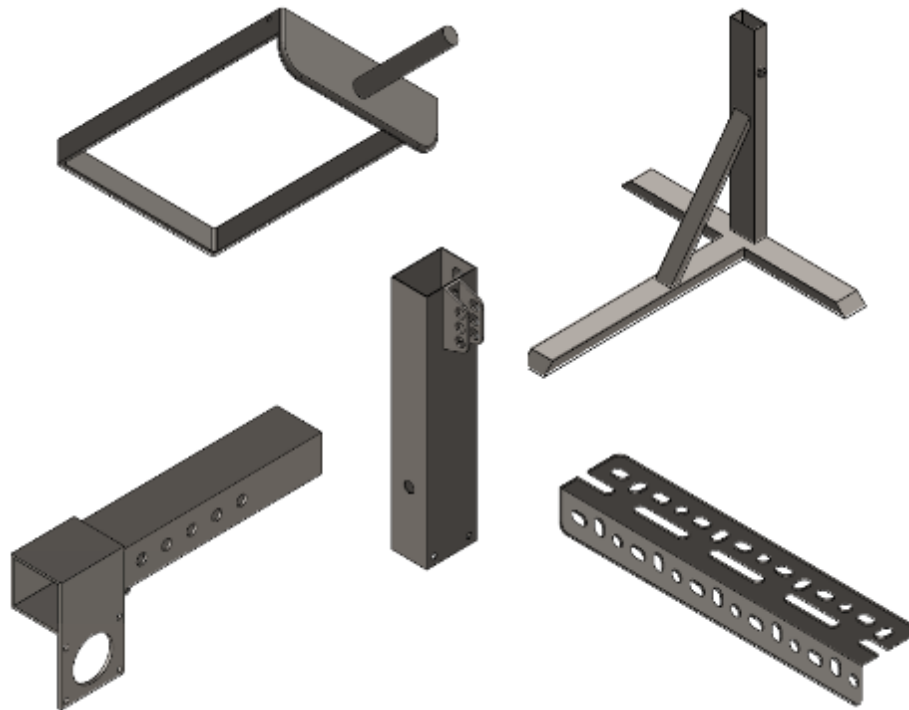


Figura 7-3: Modelación de piezas base para el rehabilitador en SolidWorks.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para el diseño de las piezas se consideró el material de acuerdo a la verificación disponible del mismo.

3.3.2 Registro de diseños – Planos

Desarrolladas todas las piezas para un fácil manejo de información al momento de la implementación se realizaron los planos de cada una de las piezas con sus respectivas medidas por medio de planos acotados, y sus vistas respectivas para una mayor perspectiva en la construcción, con el fin de descartar posibles errores en el laminado de todas las piezas (Ver ANEXO A).

En la figura 23-3 se muestra el modelado de una de las piezas que conformarían el rehabilitador electromecánico.

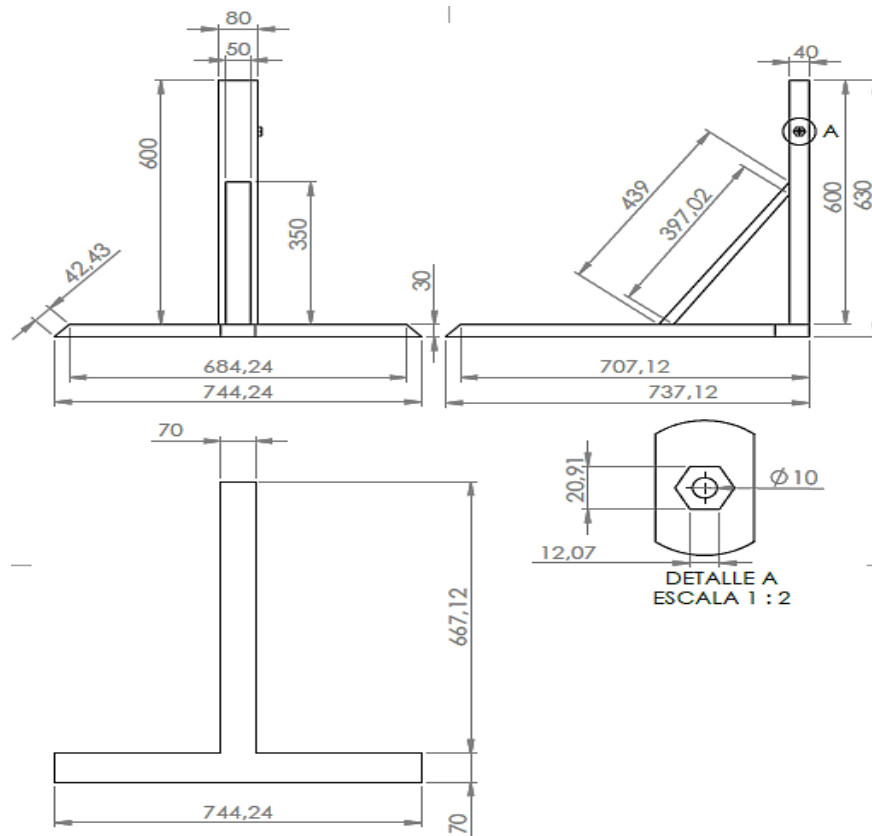


Figura 8-3: Acotación de la pieza base del prototipo rehabilitador.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.3 Ensamble del rehabilitador

Una vez laminadas las piezas que conformarían el rehabilitador se procedió a realizar el ensamble para generar un sólido tridimensional proyectado al modelo físico que se obtendría. Las láminas del ensamble se adjuntan en el ANEXO B.

La figura 24-3 muestra el diagrama explosionado del modelo en el que se permite reconocer cada una de las piezas que conforman el prototipo.

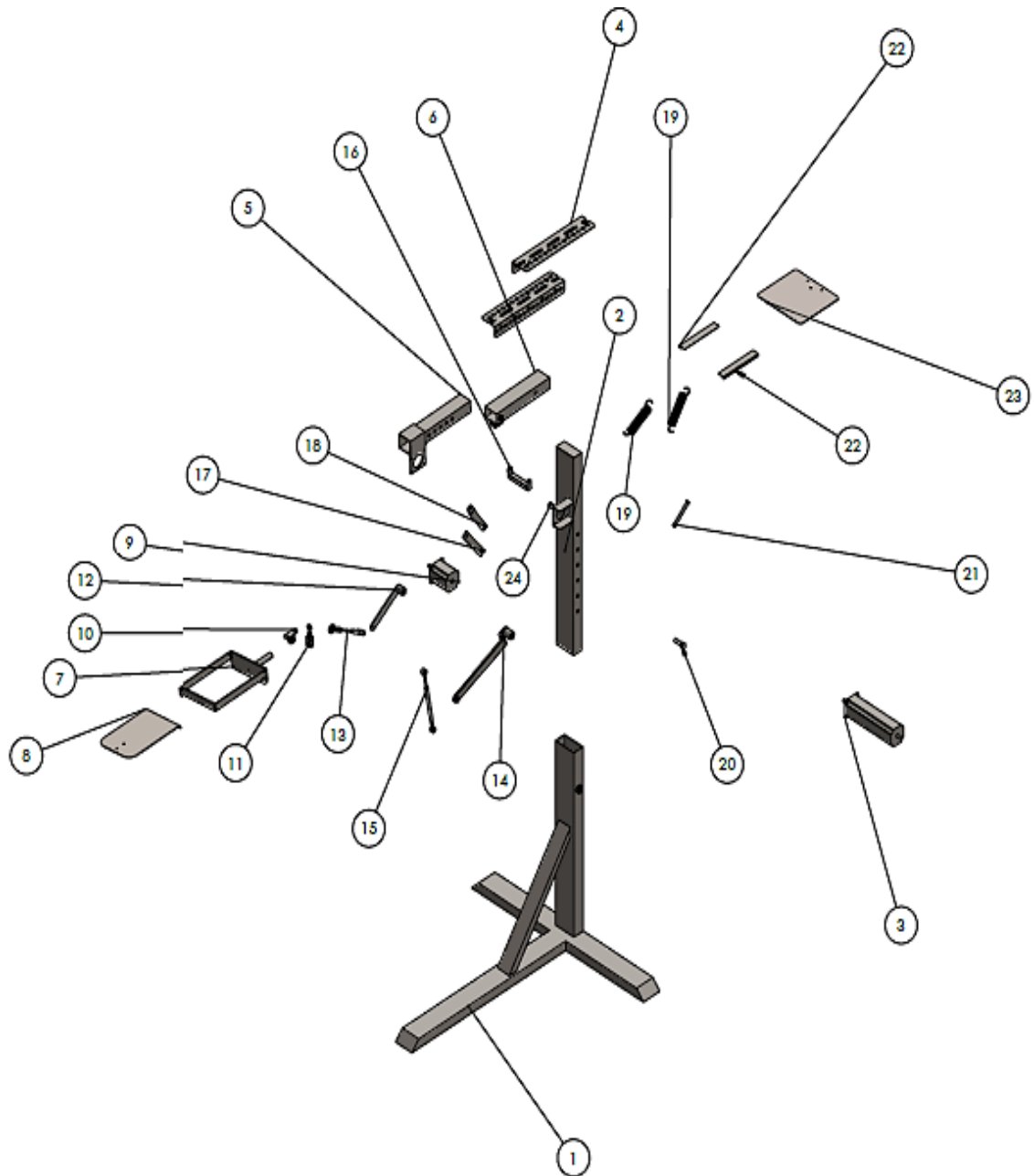


Figura 9-3: Diagrama explosionado del rehabilitador.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

En la Tabla 1-3 se muestra las piezas del prototipo rehabilitador.

Tabla 1-3: Listado de piezas en el diseño del rehabilitador (Extraída del ANEXO B)

N.º DE ELEMENTO	N.º DE PIEZA	DESCRIPCIÓN	CANTIDAD
1	base inicial		1
2	base principal		1
3	Motor antebrazo		1
4	pieza agujereada		2
5	pieza superior a mano		1
6	soporte de cilindro y resorte		1
7	base de mano		1
8	base mano		1
9	Motor mano		1
10	suj de mano		1
11	SUJECION2 MANO		1
12	brazo1 mano		1
13	brazo 2mano		1
14	PALANCA 1 ANTEBRAZO		1
15	palanca antebrazo		1
16	pieza en U soporte		1
17	soporte resorte		1
18	refuerzo a resorte		1
19	RESORTES		2
20	tornillo sop		1
21	soporte seguro		1
22	soportdelbrazo		2
23	base brazo		1
24	pasado antenbrazo		1

Fuente: SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 25-3 representa el plano global del diseño en el que se adjuntan vistas y cotas principales del prototipo para facilitar su ensamble en la etapa de implementación. La lámina completa se adjunta en el ANEXO B.

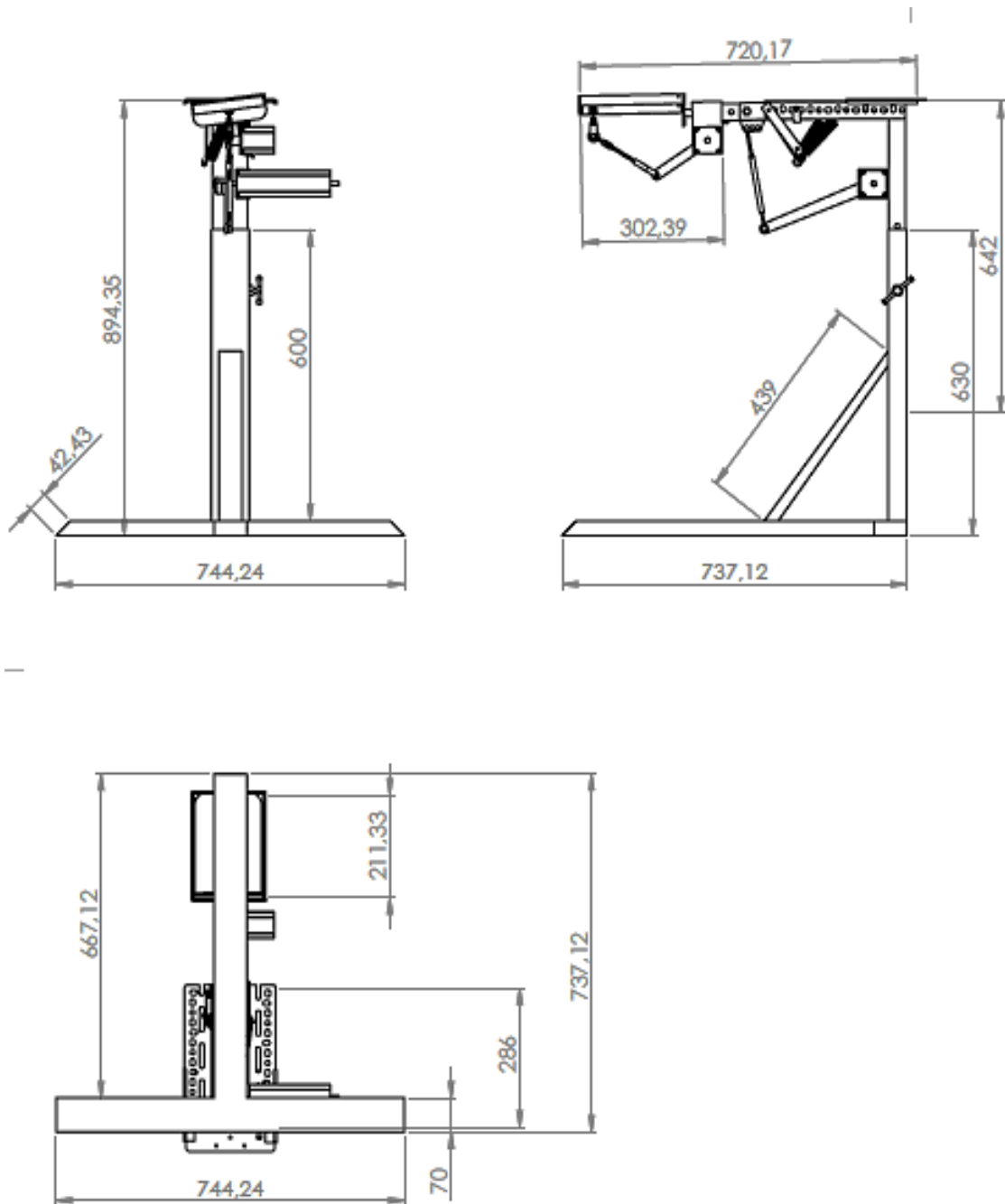


Figura 10-3: Cotas del ensamble total

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La disponibilidad de un plano bidimensional facilita la construcción de una máquina o equipo, para el prototipo se realizó un modelo tridimensional utilizando SolidWorks para poner a disposición del constructor.

La figura 26-3 representa una de las vistas del modelo tridimensional ensamblado.

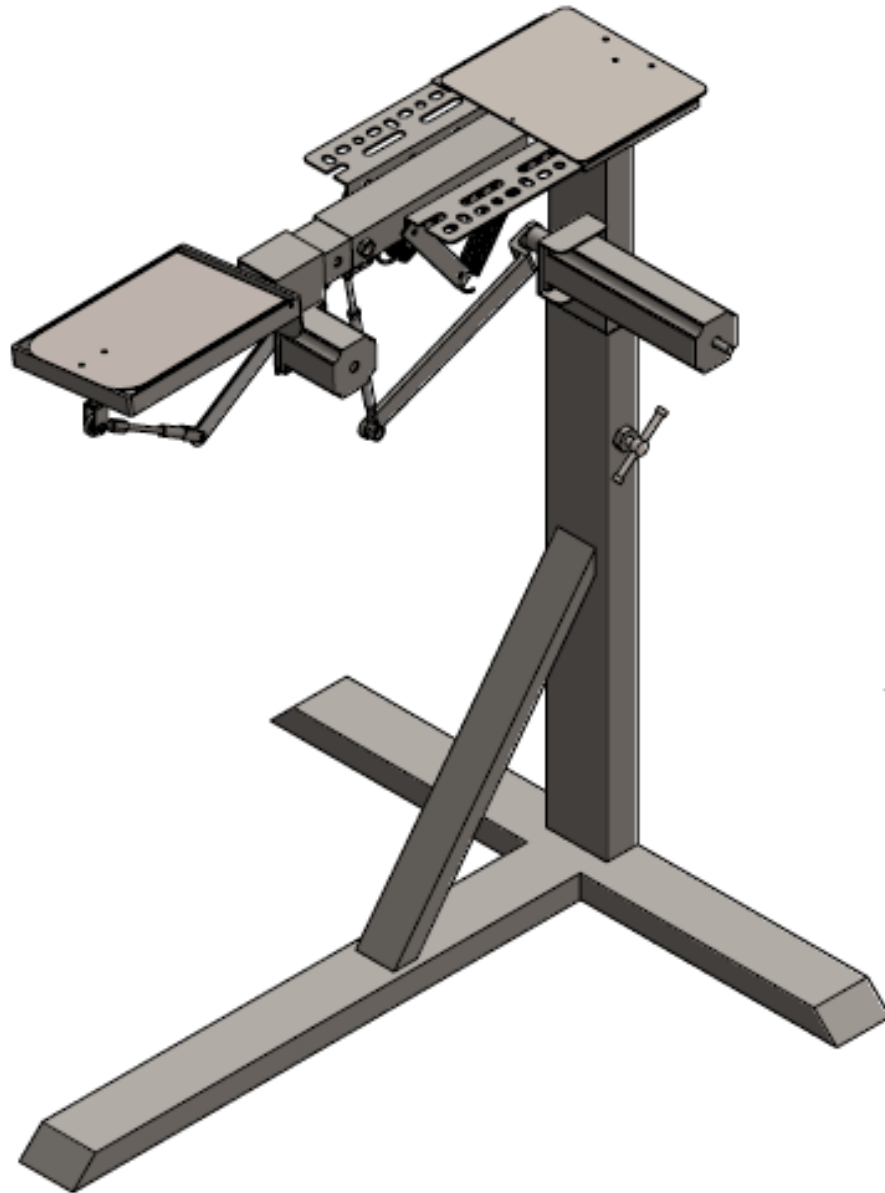


Figura 11-3: Modelo tridimensional ensamblado del rehabilitador

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.4 Análisis estático del prototipo rehabilitador

El análisis estático permite verificar si el tipo de material empleado en el diseño de las piezas es el adecuado. Los principales elementos del rehabilitador son: resortes, base principal, base del motor del antebrazo, base de alojamiento del antebrazo, soporte del cilindro y resorte, soporte para giro de la muñeca, elemento de movilidad de la mano, apoyo para rehabilitación de la mano, soporte de movimiento de mano y brazo de movilidad del antebrazo se aplicó el análisis de

esfuerzos considerando que son aquellos sometidos a mayor energía en la funcionalidad con su respectiva conclusión donde se menciona al diseño y material.

a) Resortes

La figura 27-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada resortes

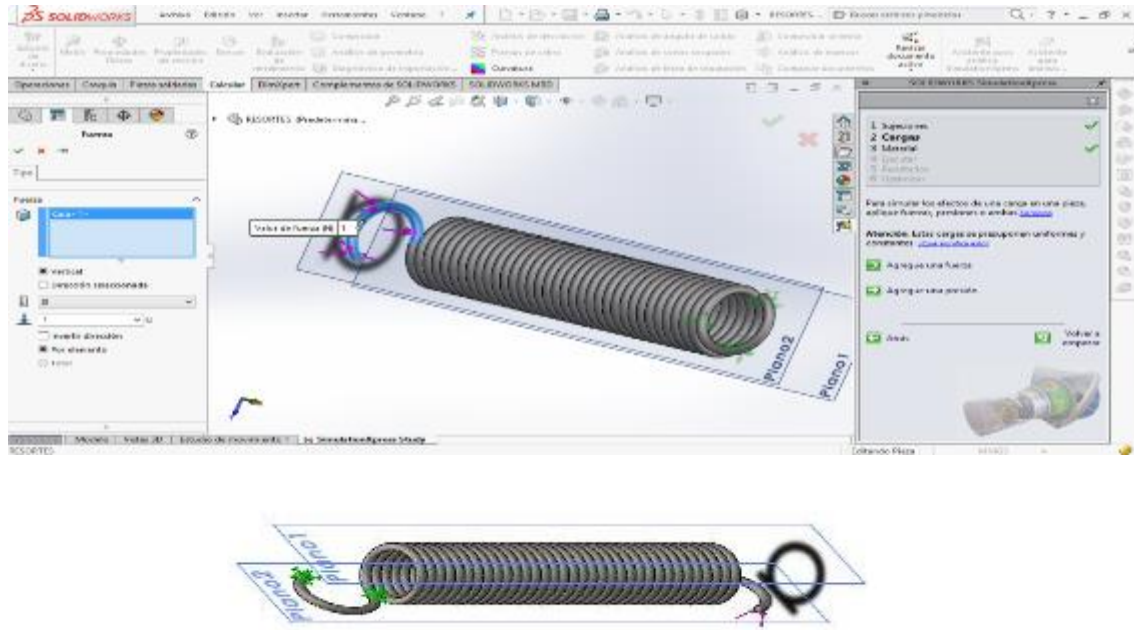



Figura 12-3: Vista de la pieza resorte en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020


En las tablas 2-3 y 3-3 se detallan las propiedades volumétricas de la pieza resorte y las propiedades del material empleado para su diseño.

Tabla 2-3: Propiedades de la pieza - Resorte

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
Barrer5 	Sólido	Masa:0.0853801 kg Volumen:1.08765e-005 m ³ Densidad:7850 kg/m ³ Peso:0.836725 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 3-3: Propiedades del material - Resortes

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error Desconocido predeterminado: Límite elástico: 250 N/mm² Límite de tracción: 400 N/mm²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

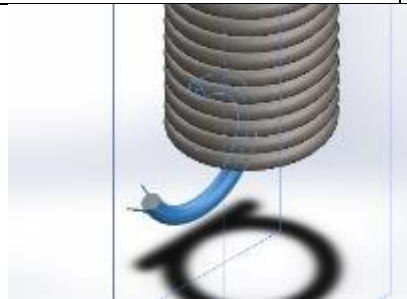
Se ubican los parámetros en SolidWorks para el análisis estático, como es el caso de los elementos de apoyo o sujeciones y las cargas que va a soportar el resorte.

Tabla 4-3: Sujeciones - Resorte

NOMBRE DE SUJECCIÓN	IMAGEN DE SUJECCIÓN	DETALLES DE SUJECCIÓN
Fijo-1		Entidades: 1 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 5-3: Cargas - Resorte

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 250 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

b) Base principal

La figura 28-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada base principal.

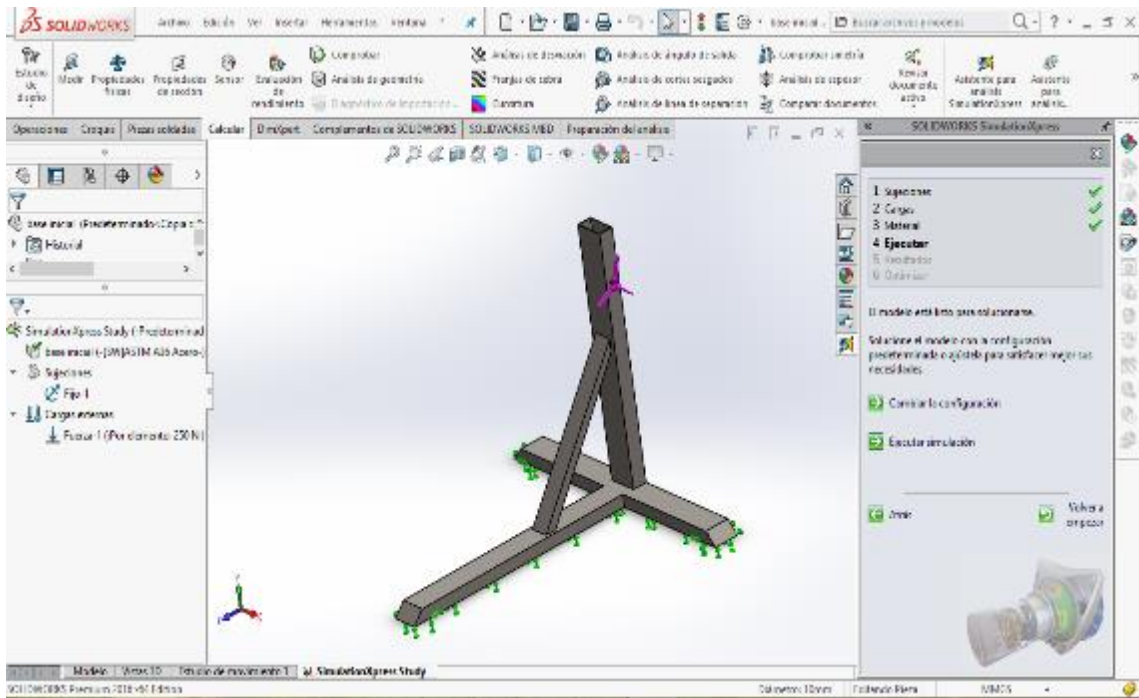


Figura 13-3: Vista de la pieza base principal en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

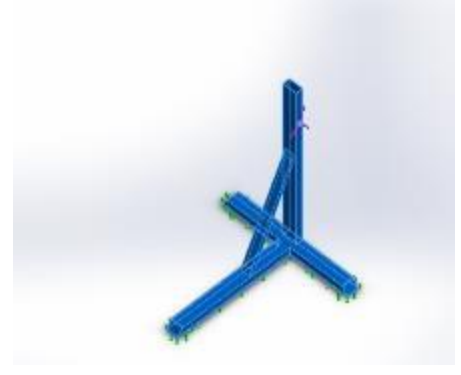
En las tablas 6-3 y 7-3 se detallan las propiedades volumétricas de la pieza base principal y las propiedades del material empleado para su diseño respectivamente.

Tabla 6-3: Propiedades de la pieza – Base principal

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
Saliente-Extruir9 	Sólido	Masa:11.7372 kg Volumen:0.00149519 m ³ Densidad:7850 kg/m ³ Peso:115.025 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

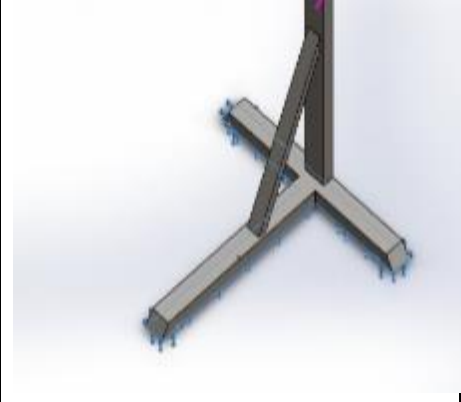
Tabla 7-3: Propiedades del material – Base principal

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error: Desconocido predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

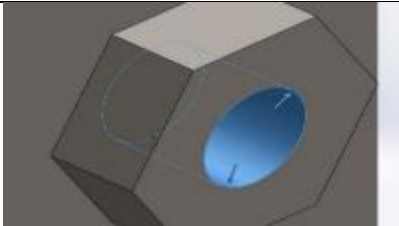
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar la base principal.

Tabla 8-3: Sujeciones - Base principal

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1		Entidades: 1 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 9-3: Cargas - Base principal

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 250 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

c) Base del motor del antebrazo

La figura 29-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada base del motor del antebrazo.

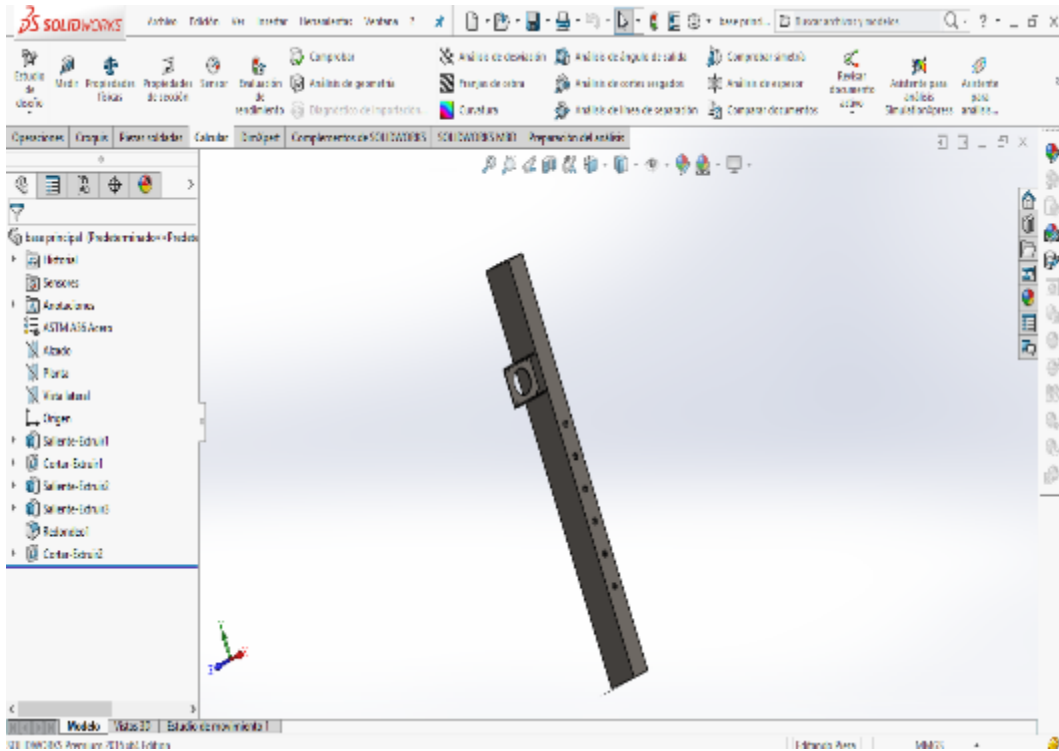
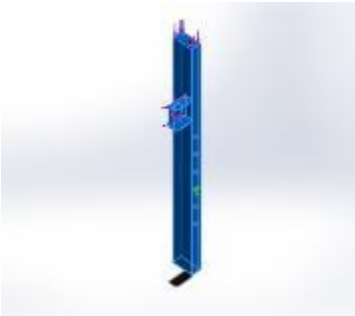


Figura 14-3: Vista de la pieza base del motor del antebrazo en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

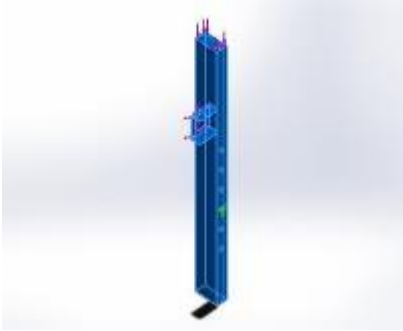
En las tablas 10-3 y 11-3 se detallan las propiedades volumétricas de la pieza base del motor del antebrazo y las propiedades del material empleado para su diseño respectivamente.

Tabla 10-3: Propiedades de la pieza – Base del motor del antebrazo

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
	Sólido	Masa:3.74181 kg Volumen:0.000476664 m ³ Densidad:7850 kg/m ³ Peso:36.6697 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 11-3: Propiedades del material – Base del motor del antebrazo

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error: Desconocido predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

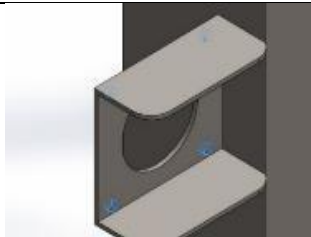
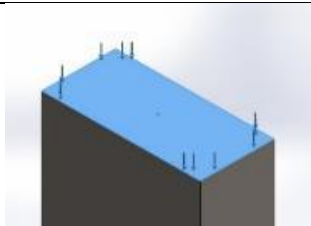
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar la base del motor del antebrazo.

Tabla 12-3: Sujeciones - Base del motor del antebrazo

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1		Entidades: 1 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 13-3: Cargas - Base del motor del antebrazo

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1		Entidades: 4 cara(s) Tipo: Aplicar fuerza normal Valor: 10 N
Fuerza-2		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 200 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

d) Base de alojamiento del antebrazo

La figura 30-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada base de alojamiento del antebrazo.

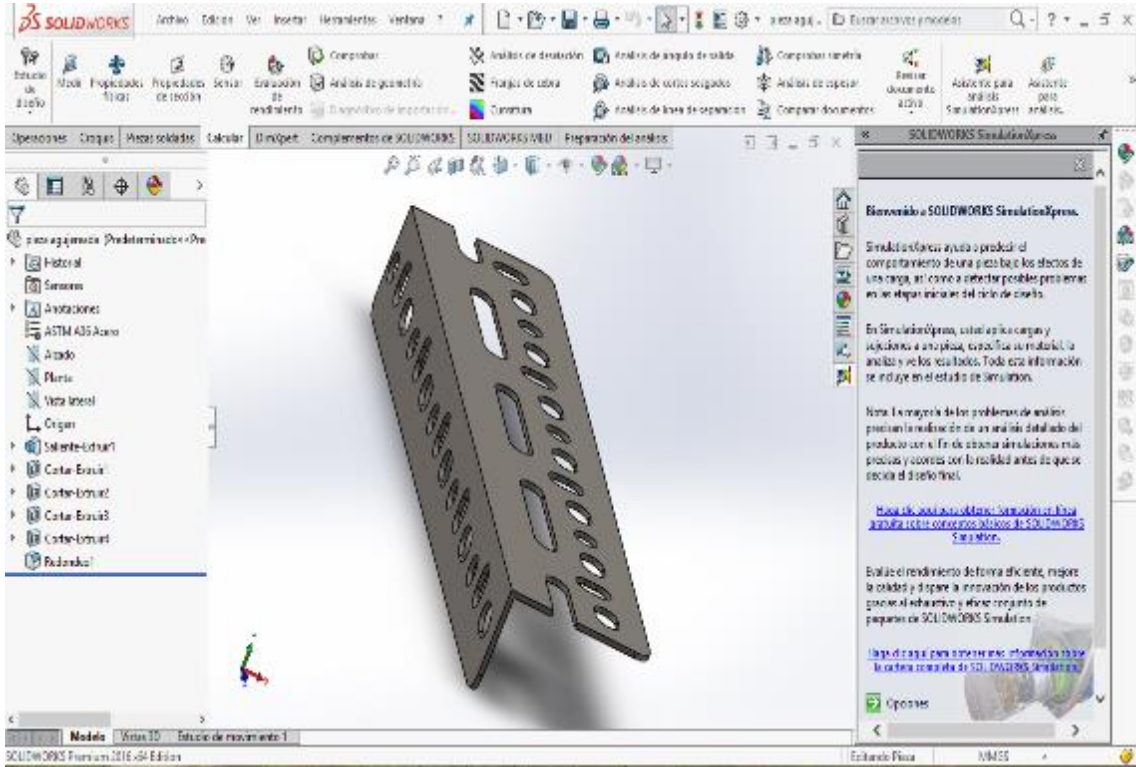



Figura 15-3: Vista de la pieza base de alojamiento del antebrazo en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020


En las tablas 14-3 y 15-3 se detallan las propiedades volumétricas de la pieza base de alojamiento del antebrazo y propiedades del material empleado para su diseño respectivamente.

Tabla 14-3: Propiedades de la pieza – Base de alojamiento del antebrazo

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
<p>Redondeo1</p> 	Sólido	<p>Masa:0.363237 kg Volumen:4.62722e-005 m³ Densidad:7850 kg/m³ Peso:3.55972 N</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 15-3: Propiedades del material – Base de alojamiento del antebrazo

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error Desconocido predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

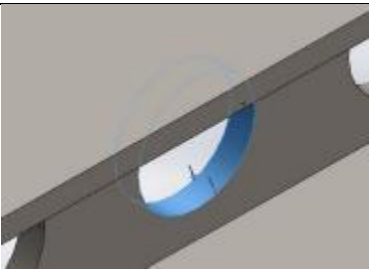

Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar la base de alojamiento del antebrazo.

Tabla 16-3: Sujeciones – Base de alojamiento del antebrazo

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1		Entidades: 1 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 17-3: Cargas – Base de alojamiento del antebrazo

Nombre de carga	Cargar imagen	Detalles de carga
Fuerza-1		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 150 N
Fuerza-2		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 50 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

e) Soporte del cilindro y resorte

La figura 31-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada soporte del cilindro y resorte.

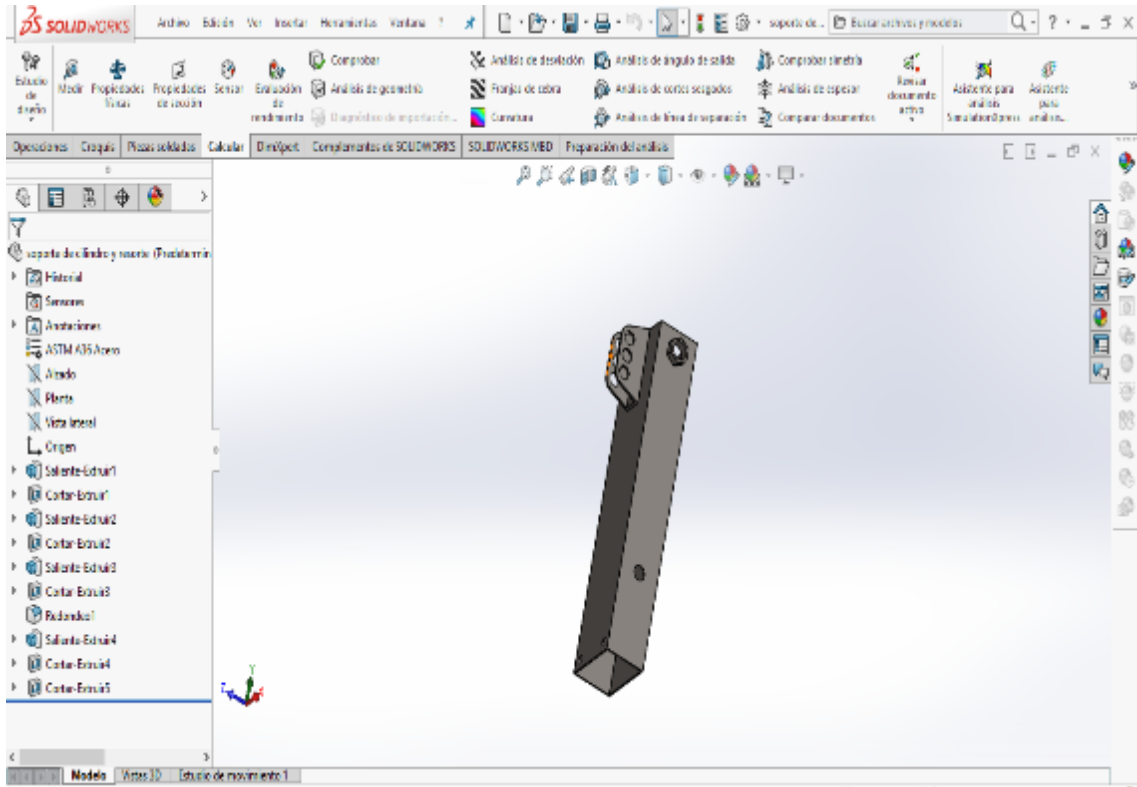



Figura 16-3: Vista de la pieza soporte del cilindro y resorte en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

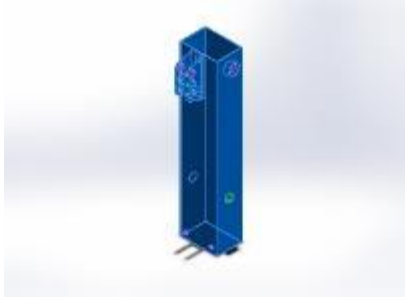
En las tablas 18-3 y 19-3 se detallan las propiedades volumétricas de la pieza soporte del cilindro, resorte y las propiedades del material empleado para su diseño respectivamente.

Tabla 18-3: Propiedades de la pieza – Soporte del cilindro y resorte

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
<p>Cortar-Extruir5</p> 	Sólido	<p>Masa:0.336376 kg Volumen:4.28505e-005 m³ Densidad:7850 kg/m³ Peso:3.29649 N</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020


Tabla 19-3: Propiedades del material – Soporte del cilindro y resorte

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error Desconocido predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

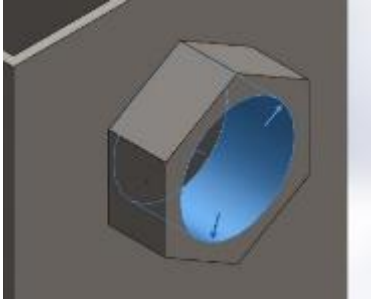
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar el soporte del cilindro y resorte.

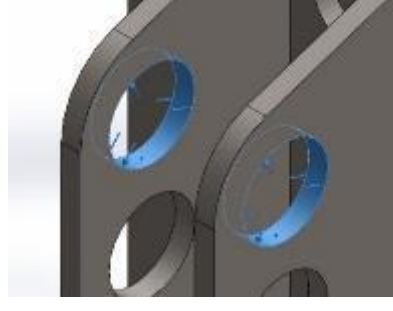
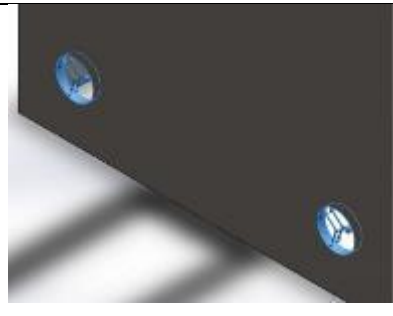
Tabla 20-3: Sujeciones – Soporte del cilindro y resorte

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1		Entidades: 2 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 21-3: Cargas – Soporte del cilindro y resorte

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 100 N

Fuerza-2		Entidades: 2 cara(s) Tipo: Aplicar fuerza normal Valor: 40 N
Fuerza-3		Entidades: 2 cara(s) Tipo: Aplicar fuerza normal Valor: 20 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

f) Soporte para el giro de la muñeca

La figura 32-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada soporte para el giro de la muñeca

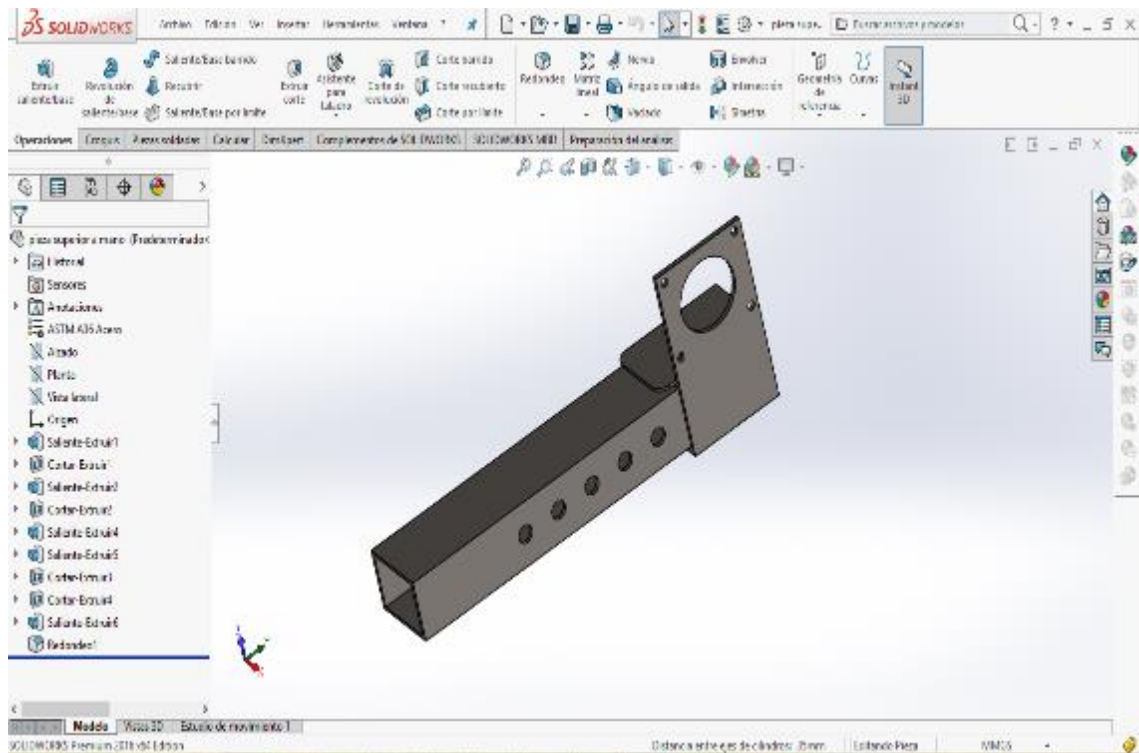
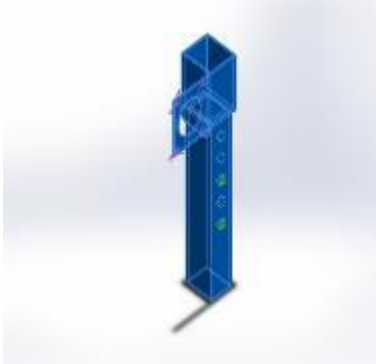


Figura 17-3: Vista de la pieza soporte para el giro de la muñeca en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

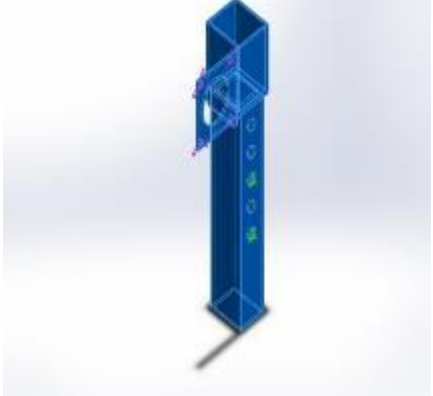
En las tablas 22-3 y 23-3 se detallan las propiedades volumétricas de la pieza soporte para el giro de la muñeca y las propiedades del material empleado para su diseño respectivamente.

Tabla 22-3: Propiedades de la pieza – Soporte para el giro de la muñeca

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
Redondeo1 	Sólido	Masa:0.848535 kg Volumen:0.000108094 m ³ Densidad:7850 kg/m ³ Peso:8.31564 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

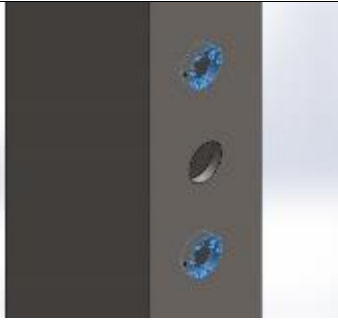
Tabla 23-3: Propiedades del material – Soporte para el giro de la muñeca

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error Desconocido predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

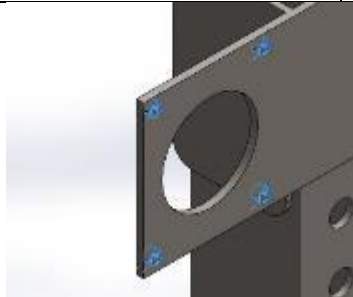
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a resistir el soporte para el giro de la muñeca.

Tabla 24-3: Sujeciones – Soporte para el giro de la muñeca

NOMBRE DE SUJECIÓN	DE	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1			Entidades: 2 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 25-3: Cargas – Soporte para el giro de la muñeca

NOMBRE DE CARGA	DE	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1			Entidades: 4 cara(s) Tipo: Aplicar fuerza normal Valor: 40 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

g) Elemento de movilidad para la mano

La figura 33-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada elemento de movilidad para la mano.

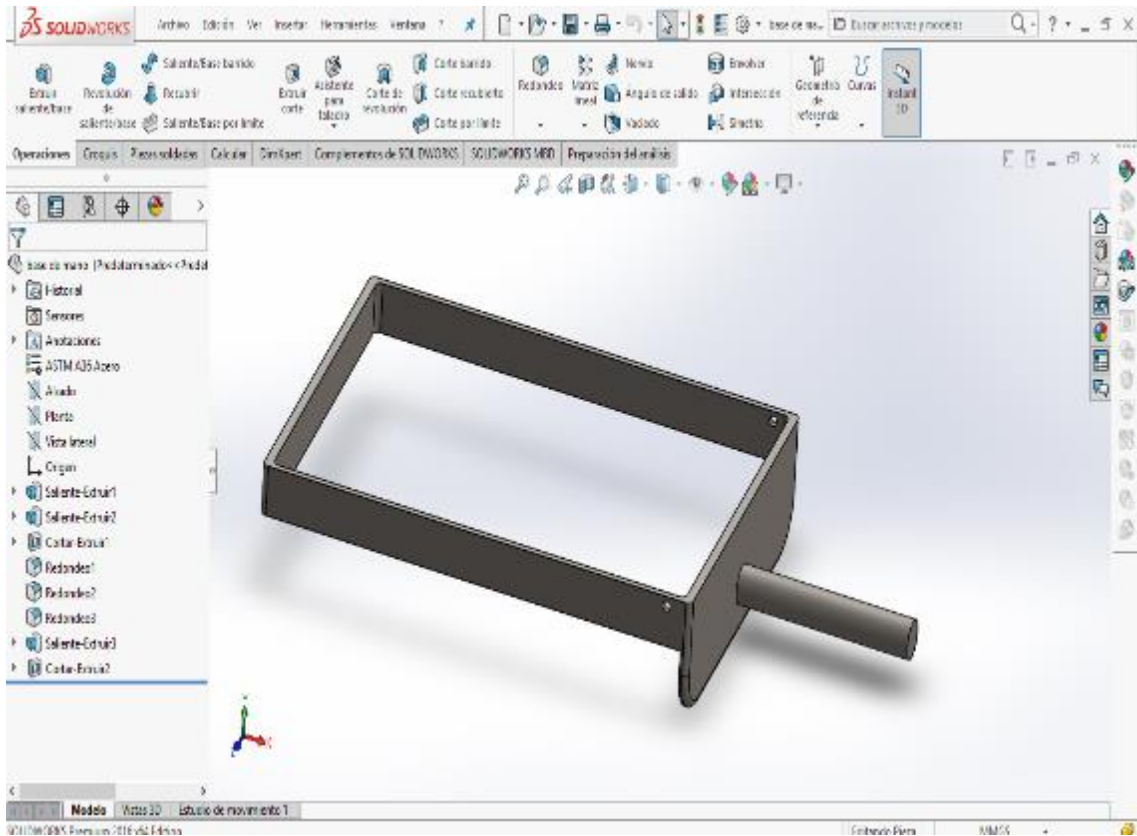



Figura 18-3: Vista de la pieza elemento de movilidad para la mano en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

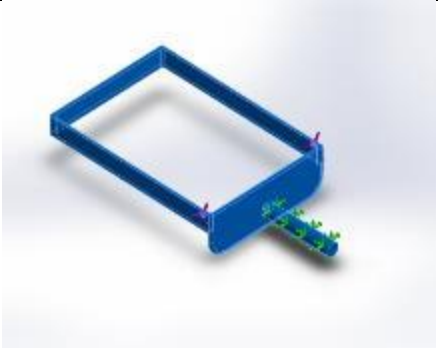
En las tablas 26-3 y 27-3 se detallan las propiedades volumétricas de la pieza elemento de movilidad para la mano y propiedades del material empleado para su diseño respectivamente.

Tabla 26-3: Propiedades de la pieza – Elemento de movilidad para la mano

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
Cortar-Extruir2 	Sólido	Masa:0.69209 kg Volumen:8.81644e-005 m ³ Densidad:7850 kg/m ³ Peso:6.78249 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

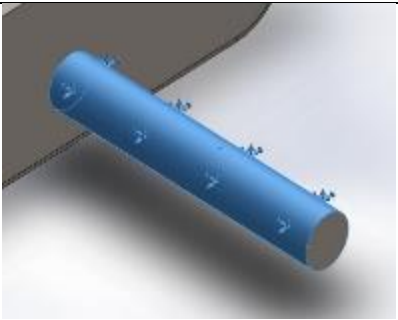
Tabla 27-3: Propiedades del material – Elemento de movilidad para la mano

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error Desconocido predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

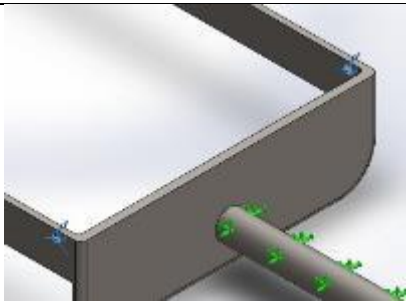
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar.

Tabla 28-3: Sujeciones – Elemento de movilidad para la mano

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1		Entidades: 1 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 29-3: Cargas – Elemento de movilidad para la mano

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1 Elemento de movilidad para la mano		Entidades: 2 cara(s) Tipo: Aplicar fuerza normal Valor: 50 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

h) Apoyo para rehabilitación de mano

La figura 34-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada apoyo para rehabilitación de mano

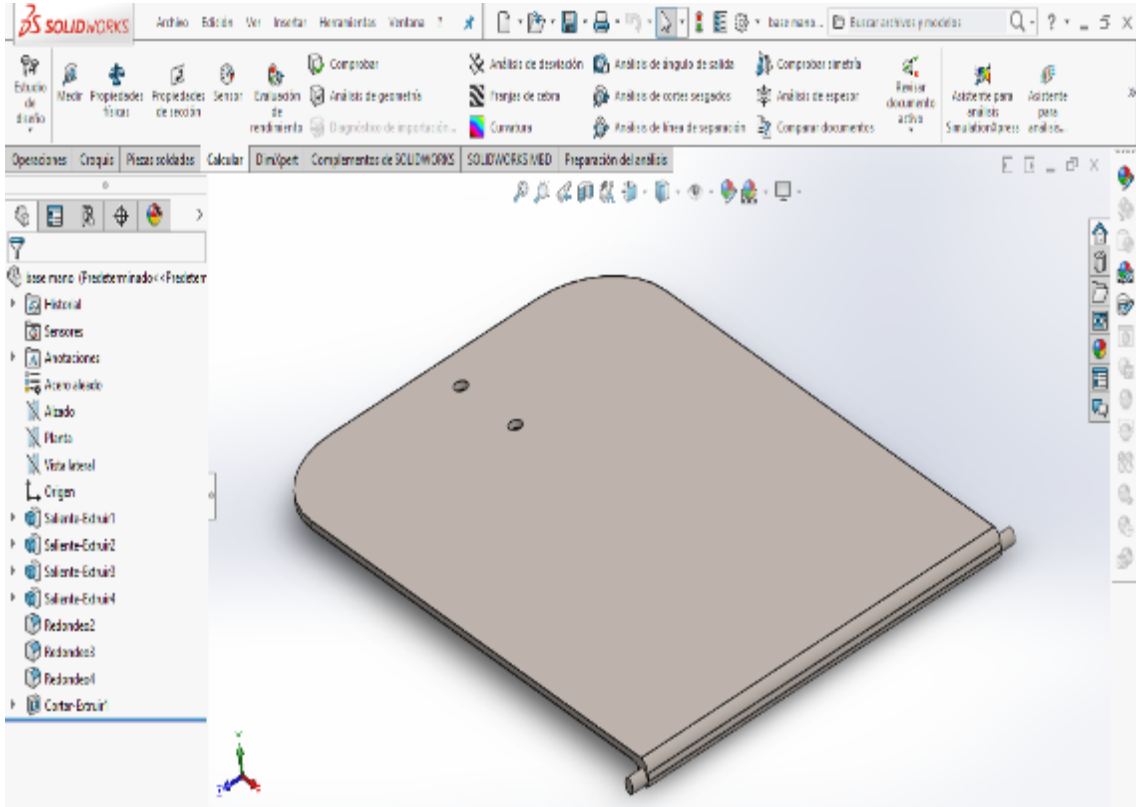



Figura 19-3: Vista de la pieza apoyo para rehabilitación de mano en SolidWorks

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

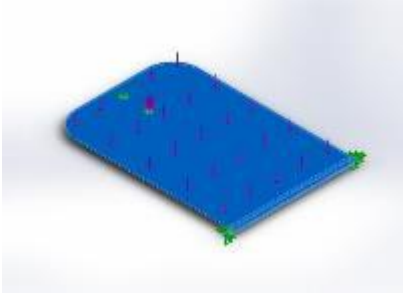
En las tablas 30-3 y 31-3 se detallan las propiedades volumétricas de la pieza apoyo para rehabilitación de mano y propiedades del material empleado para su diseño respectivamente.

Tabla 30-3: Propiedades de la pieza – Apoyo para rehabilitación de mano

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
<p>Cortar-Extruir1</p> 	Sólido	<p>Masa:0.509231 kg Volumen:6.48702e-005 m³ Densidad:7850 kg/m³ Peso:4.99046 N</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

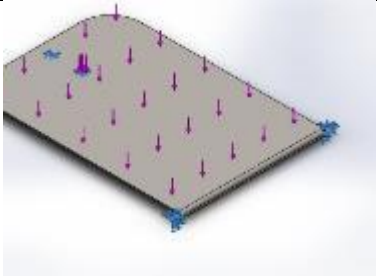
Tabla 31-3: Propiedades del material – Apoyo para rehabilitación de mano

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error: Tensión de von Mises máx. predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

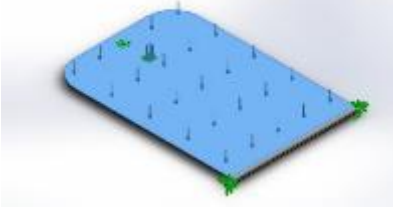
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar.

Tabla 32-3: Sujeciones – Apoyo para rehabilitación de mano

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-2		Entidades: 4 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 33-3: Cargas – Apoyo para rehabilitación de mano

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-2		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 100 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

i) Soporte de movimiento de mano

La figura 35-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada soporte de movimiento de mano.

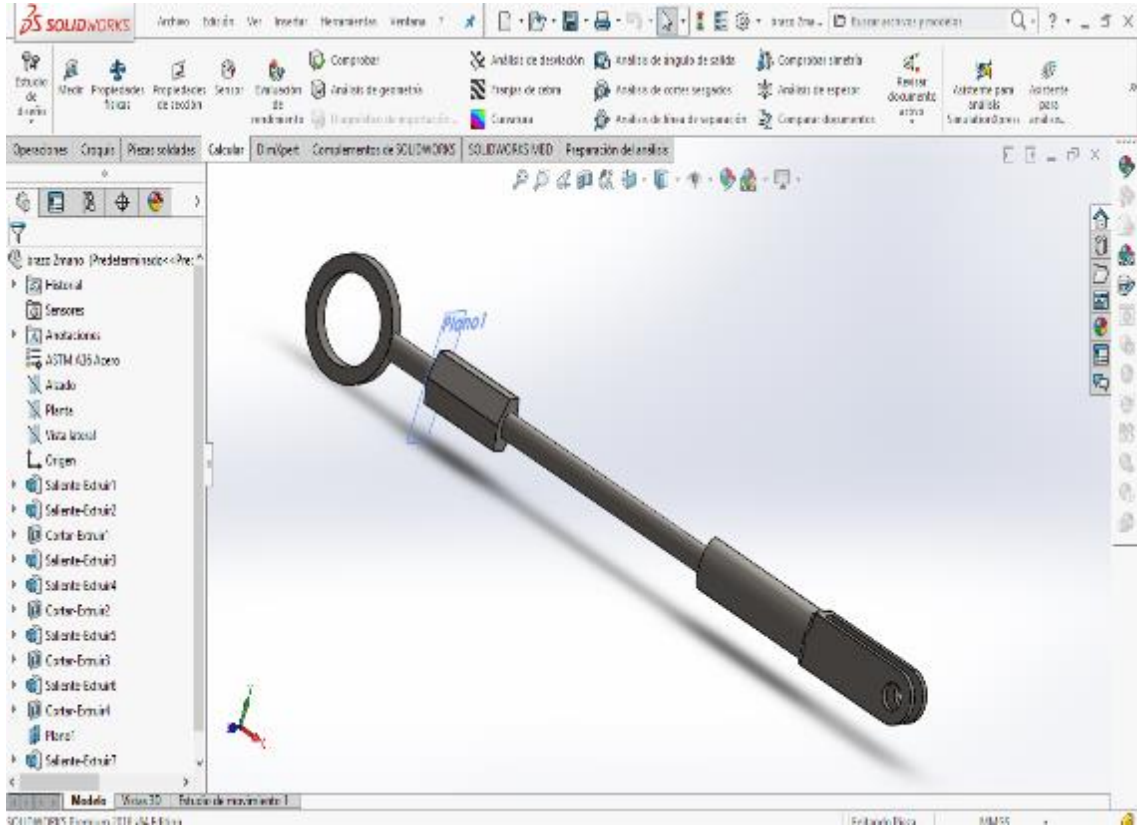



Figura 20-3: Vista de la pieza soporte de movimiento de mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020


En las tablas 34-3 y 35-3 se detallan las propiedades volumétricas de la pieza soporte de movimiento de mano y las propiedades del material empleado para su diseño respectivamente.

Tabla 34-3: Propiedades de la pieza – Soporte de movimiento de mano

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
<p>Redondeo1</p> 	Sólido	<p>Masa:0.0620012 kg Volumen:7.89824e-006 m³ Densidad:7850 kg/m³ Peso:0.607612 N</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

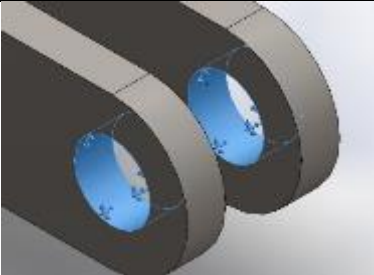
Tabla 35-3: Propiedades del material – Soporte de movimiento de mano

REFERENCIA DE MODELO	PROPIEDADES
	<p>Nombre: ASTM A36 Acero</p> <p>Tipo de modelo: Isotrópico elástico lineal</p> <p>Criterio de error Desconocido</p> <p>predeterminado:</p> <p>Límite elástico: 250 N/mm²</p> <p>Límite de tracción: 400 N/mm²</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

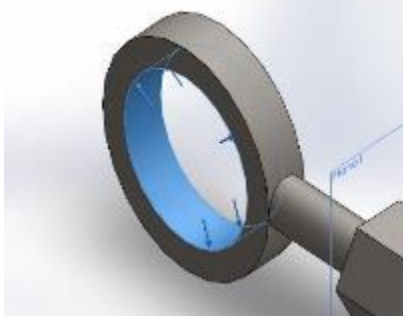
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas a la que va a soportar el soporte de movimiento de mano.

Tabla 36-3: Sujeciones – Soporte de movimiento de mano

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-1		<p>Entidades: 2 cara(s)</p> <p>Tipo: Geometría fija</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 37-3: Cargas – Soporte de movimiento de mano

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-1		<p>Entidades: 1 cara(s)</p> <p>Tipo: Aplicar fuerza normal</p> <p>Valor: 100 N</p>

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

j) Brazo de movilidad del Antebrazo

La figura 36-3 muestra el dibujo elaborado en SolidWorks de la pieza denominada brazo de movilidad del Antebrazo.

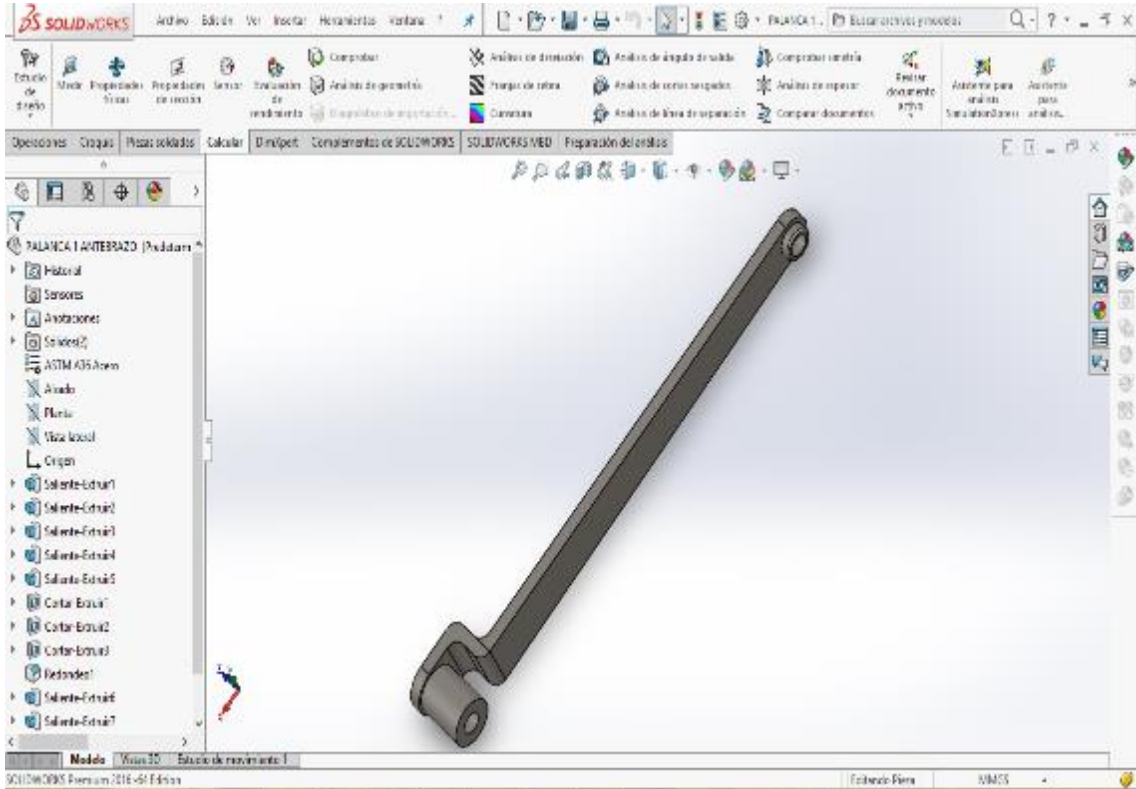



Figura 21-3: Vista de la pieza brazo de movilidad del antebrazo.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020


En las tablas 38-3 y 39-3 se detallan las propiedades volumétricas del brazo de movilidad del antebrazo y las propiedades del material empleado para su diseño respectivamente.

Tabla 38-3: Propiedades de la pieza – Brazo de movilidad del Antebrazo

NOMBRE DE DOCUMENTO Y REFERENCIA	TRATADO COMO	PROPIEDADES VOLUMÉTRICAS
Redondeo2 	Sólido	Masa:0.368781 kg Volumen:4.69785e-005 m ³ Densidad:7850 kg/m ³ Peso:3.61405 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 39-3: Propiedades del material– Brazo de movilidad del Antebrazo

REFERENCIA DE MODELO	PROPIEDADES
	Nombre: ASTM A36 Acero Tipo de modelo: Isotrópico elástico lineal Criterio de error: Tensión de von Mises máx. predeterminado: Límite elástico: 250 N/mm ² Límite de tracción: 400 N/mm ²

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

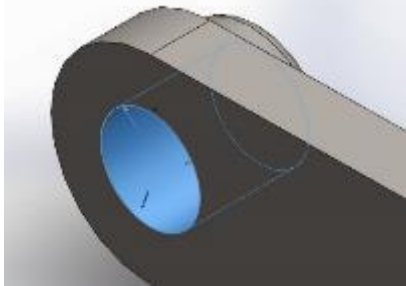
Se ubican los parámetros en SolidWorks para el análisis estático, como son los elementos de apoyo o sujeciones y las cargas que va a soportar el brazo de movilidad del antebrazo.

Tabla 40-3: Sujeciones – Brazo de movilidad del Antebrazo

NOMBRE DE SUJECIÓN	IMAGEN DE SUJECIÓN	DETALLES DE SUJECIÓN
Fijo-2		Entidades: 1 cara(s) Tipo: Geometría fija

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 41-3: Cargas – Brazo de movilidad del Antebrazo

NOMBRE DE CARGA	CARGAR IMAGEN	DETALLES DE CARGA
Fuerza-2		Entidades: 1 cara(s) Tipo: Aplicar fuerza normal Valor: 200 N

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

De esta manera se puede visualizar todos los datos sobre el material empleado en el diseño de las piezas, sus sujeciones y cargas a soportar, para posteriormente aplicando el análisis estático poder analizar los resultados del mismo y poder validar el diseño para su implementación.

3.3.5 Hardware del prototipo rehabilitador

Este apartado se lo denomina hardware a razón que se incluye información de los equipos y accesorios empleados para el control de movilidad del rehabilitador como: controladores y actuadores.

3.3.5.1 Actuadores

Acorde al diseño planteado para conseguir que el rehabilitador pueda incitar al paciente a realizar los movimientos establecidos en el protocolo de rehabilitación, se lo implementó con el uso de tres motores con un grado de libertad.

Uno de los criterios para la selección de los actuadores fue el proceso de rehabilitación que no es igual en todos los pacientes, debido a que unos necesitan un rango de movimiento mayor o menor para su recuperación.

El uso de motores eléctricos de corriente continua para este caso no es factible, porque no presentan precisión al momento del arranque y parada, por lo que se optó el uso de motores paso a paso que disponen de una excitación independiente. La alimentación del inductor se realiza a partir de una fuente de alimentación externa, lo que facilita el control de su velocidad al modificar y controlar los valores de la corriente de excitación. Se puede describir como modo de funcionamiento de estos motores que se encargan de convertir pulsos digitales en movimiento de rotación mecánica. (CLR, 2020)

“La proporción de la rotación es proporcional al número de pulsos generados, mientras que la velocidad de rotación se relaciona con la frecuencia de esos pulsos. Los impulsos, en definitiva, se definen por un ángulo predeterminado que es alimentado por un dispositivo programable”. (CLR, 2020)

Al tratarse de un equipo de rehabilitación se requiere de mucha precisión en los desplazamientos por lo que se optó por este tipo de actuadores.

La selección de los motores eléctricos se hizo en base a las fuerzas de sujeción y cargas que se emplearon en el análisis estático, no solo se consideró el peso de la estructura del equipo sino también la incidencia del peso de la extremidad del paciente.

La figura 37-3 muestra la ubicación de cada motor donde se les asigna un identificador para las posteriores citaciones de los mismos.

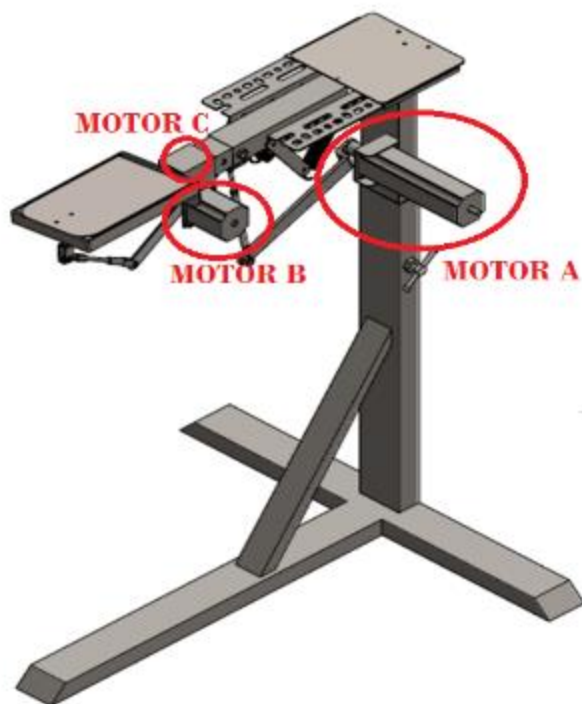


Figura 22-3: Ubicación de los motores en el rehabilitador

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La tabla 42-3 representa la descripción de la función de cada motor del rehabilitador que se seleccionó mediante catálogo.

Tabla 42-3: Selección de actuadores

MOTOR	EJECUCIÓN MOVIMIENTO	TORQUE (N-m ; Oz.in)	Ángulo de paso	MOTOR SELECCIONADO
A	FLEXOEXTENSIÓN DEL CODO	3N-m 425 Oz-in	1.8°	NEMA 23 MODEL 23HS9430B
B	FLEXOEXTENSIÓN DE LA MUÑECA	1.9N-m 269 Oz-in	1.8°	NEMA 23 MODEL 23HD76002Y-21B
C	PRONOSUPINACIÓN	0.26 N-m 36.8 Oz-in	1.8°	NEMA 17 MODEL 17HS13-0404S

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 38-3 representa el motor NEMA 23 MODEL 23HS9430B y en conjunto con la tabla 43-3 enuncian sus principales características eléctricas:



Figura 23-3: NEMA 23 MODEL 23HS9430B.

Fuente: <https://url2.cl/vCq1s>

Tabla 43-3: Características eléctricas del Motor NEMA 23 MODEL 23HS9430B.

NEMA 23 MODEL 23HS9430B.	
Tipo de Motor	BIPOLAR
Ángulo de motor	1.8°
Par de retención (Holding Torque)	3.0 Nm (425 oz/in) (30.60 Kg/cm)
Corriente nominal / fase	4.2 ^a
Resistencia de fase	0.9ohms
Tensión recomendada	3.78V
Inductancia	3.8mH ± 20%(1KHz)

Fuente: (Valderrama, Alexis, 2020)

La figura 39-3 representa el motor NEMA 23 23HD76002Y-21B y en conjunto con la tabla 44-3 enuncian sus principales características eléctricas:



Figura 24-3: NEMA 23 23HD76002Y-21B

Fuente: <https://url2.cl/XeEAE>

Tabla 44-3: Características eléctricas del Motor NEMA 23 MODEL 23HD76002Y-21B

NEMA 23 23HD76002Y-21B	
Tipo de Motor	BIPOLAR
Ángulo de motor	1.8°
Par de retención (Holding Torque)	1.9N-m (269 Oz-in)
Corriente nominal / fase	3 ^a
Resistencia de fase	1.2 ohms
Tensión recomendada	3.78V
Inductancia	3.8mH ± 20%(1KHz)

Fuente: (Valderrama, Alexis, 2020)

La figura 40-3 representa el motor NEMA 17 y en conjunto con la tabla 45-3 enuncian sus principales características eléctricas:



Figura 25-3:. NEMA 17 MODEL17HS13-0404S

Fuente: <https://url2.cl/nVKmk>

Tabla 45-3: Características eléctricas del Motor NEMA 17 MODEL17HS13-0404S

NEMA 17 MODEL17HS13-0404S	
Tipo de Motor	BIPOLAR
Ángulo de motor	1.8°
Par de retención (Holding Torque)	0.26 N-m (36.8 Oz-in)
Corriente nominal / fase	0,4 ^a
Resistencia de fase	30 ohms
Tensión recomendada	12 V
Inductancia	37mH ± 20%(1KHz)

Fuente: (Valderrama, Alexis, 2020)

En base a los motores seleccionados se procedió a la selección de los drivers para controlar los motores NEMA 23 y NEMA 17. Para ello se consideró los valores de corriente, resolución de micro pasos y sobre todo que brinden las garantías de calidad para desarrollar un funcionamiento continuo.

Para el manejo de los motores NEMA 23 se consideró el uso de los drivers DM542T por sus características de diseño.

La figura 41-3 representa el Driver DM542T y en conjunto con la tabla 46-3 enuncian sus principales características eléctricas:

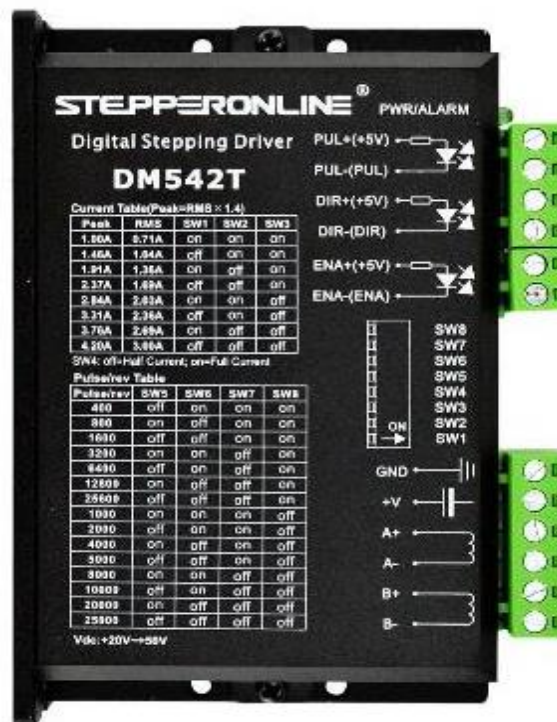


Figura 26-3: Driver DM542T

Fuente: <https://n9.cl/3x5f>

Tabla 46-3: Características eléctricas del Driver DM542T

DRIVERS DM542T	
Voltaje de entrada	18-50VDC
Corriente de entrada	< 4 ^a
Corriente de salida	1.0A ~ 42 ^a
Consumo	Consumo: 80W ; Seguros interna : 6A
Temperatura	Temperatura de trabajo -10 ~ 45°C; Stocking temperatura -40 °C ~ 70°C
Humedad	No condensación, no hay gotas de agua
Gas	Prohibición de gases combustibles y polvo conductor
Peso	200G

Fuente: (Valderrama, Alexis, 2020)

Para el manejo de los motores NEMA 17 se consideró el uso del drive BL-TB6560-V2.0 por su referencia comercial de bajo costo los cuales son muy empleados para el control de los mencionados motores.

La figura 42-3 muestra el Driver BL-TB6560-V2.0 y en conjunto con la tabla 47-3 enuncian sus principales características eléctricas:



Figura 27-3: Driver BL-TB6560-V2.0

Fuente: <https://n9.cl/s7px>

Tabla 47-3: Características del Driver BL-TB6560-V2.0

DRIVER BL-TB6560-V2.0	
Voltaje de entrada	10-35VDC
Corriente de salida	± 3 A, pico 3.5 ^a
Subdivisiones	Paso completo, medio paso, paso 1/8, paso 1/16, un máximo de 16 subdivisiones.
Dimensiones	75 x 50 x 35 mm / 2.95 x 1.96 x 1.37"

Fuente: (Valderrama, Alexis, 2020)

3.3.5.2 Tarjetas Raspberry Pi3 y Arduino UNO

Una vez seleccionados los drivers para el manejo de los motores paso a paso se seleccionó un gestor de señales de control que gestione el comportamiento de los motores a través de sus drivers.

Se planteó dentro de los requerimientos del prototipo el desarrollar una interfaz gráfica para que el usuario tenga una mejor interacción con el equipo. La tarjeta Raspberry es un minicomputador, por medio de algoritmos de control programados se vincula con la interfaz que gestiona las señales

de control emitidas hacia los drivers para el manejo de los actuadores. La figura 43-3 presenta el modelo de Raspberry Pi3 y en conjunto con la tabla 48-3 describen sus principales características.



Figura 28-3: Raspberry Pi3 Modelo B

Fuente: <https://n9.cl/3x5f>

Tabla 48-3: Características de la Raspberry Pi3

CARACTERÍSTICA	DESCRIPCIÓN
Marca	Raspberry Pi
Serie	Raspberry PI 3 Model B
Peso del producto	45,4 g
Dimensiones del producto	12,2 x 7,6 x 3,4 cm
Pilas	3 9 V (Tipo de pila necesaria)
Tipo de procesador	Core 2 Quad
Velocidad del procesador	1.20 GHz
Capacidad de la memoria RAM	1 GB
Interfaz del disco duro	ATA-4
Tipo de conectividad	WiFi
Tipo de conexión inalámbrica	802.11bgn
Número de puertos USB 2.0	4
Voltaje	5 voltios DC
Plataforma de Hardware	Linux
Sistema operativo	Linux

Fuente: (Raspberry, 2020).

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se empleó una tarjeta Arduino UNO, que conectada proporciona una comunicación serial con la Raspberry desde los algoritmos de control hacia el entorno físico que representa el prototipo electromecánico rehabilitador.

La figura 44-3 muestra la tarjeta Arduino UNO y en conjunto con la tabla 49-3 describen sus principales características.



Figura 29-3: Tarjeta Arduino UNO

Fuente: <https://n9.cl/ih0m>

Tabla 49-3: Características plataforma Arduino UNO

CARACTERÍSTICA	DESCRIPCIÓN
Microcontrolador:	ATmega328
Voltaje Operativo	5V
Voltaje de Entrada	7-12V
Voltaje de Entrada (límites)	6-20V
Pines digitales de Entrada/Salida:	14 (6 proveen salida PWM)
Corriente DC por cada Pin Entrada/Salida	40 mA
Corriente DC entregada en el Pin	3.3V: 50 mA
Memoria Flash	32 Kb de los cuales 0,5 KB usados por el bootloader
Memoria SRAM	2KB
Memoria EEPROM	1KB
Velocidad de Reloj	16 MHz
Forma de alimentación	Fuente externa o por USB
Interfaz de comunicación	USB/Serial

Fuente: (Sabika, 2010).

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.5.3 Fuente de alimentación

La fuente de alimentación, seleccionada es de 24VCC a 10A considerando la demanda que representa la sumatoria de consumos individuales de los actuadores expuestos en las tablas del

ítem 3.3.5.1. Analizando el funcionamiento del prototipo no siempre los tres motores estarán a plena carga, debido a que un solo motor gobierna uno de los movimientos requeridos para ejecución del protocolo de rehabilitación.

La figura 45-3 presenta la fuente empleada para alimentación de los motores, que es una fuente de alimentación profesional CNC de conmutación de 250 W 24 V 10 A que se utiliza a menudo en aplicaciones industriales que relacionan motores de pasos o servos.



Figura 30-3: Fuente de Alimentación - Motores

Fuente: <https://n9.cl/ih0m>

3.3.5.4 Diseño - Conexiones de hardware

Como parte de la etapa de diseño se planteó el desarrollo del diagrama de conexiones de los elementos hardware (Ver ANEXO C). En esta sección se parte por evidenciar la comunicación serial entre el Arduino UNO y la Raspberry Pi3, como se puede observar en la figura 46-3 que la comunicación se la ejecuta a través del puerto USB para el flujo bidireccional de información.

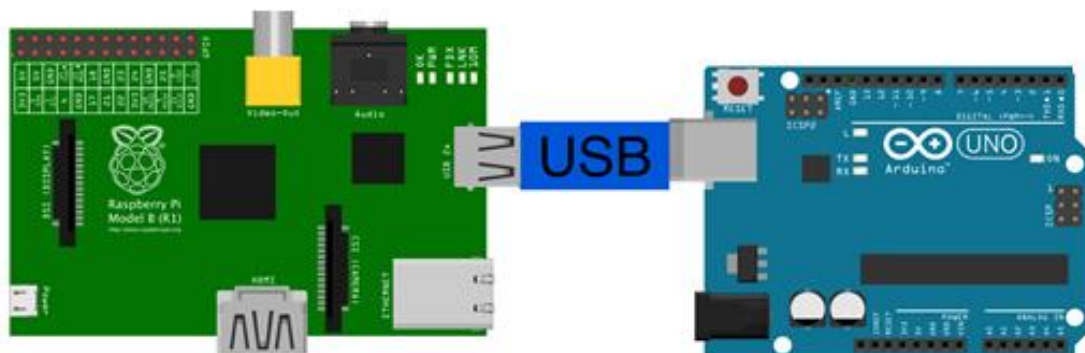


Figura 31-3: Diagrama de Conexión de la Raspberry Pi3 con el Arduino UNO

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La plataforma Arduino además de actuar como un enlace entre la parte abstracta del equipo, es decir los algoritmos de control y la parte física del rehabilitador electromecánico, actúa también como un elemento de protección de la Raspberry. La minicomputadora realiza la gestión de señales de control a través de sus terminales GPIO, pero resulta una desventaja exponer los terminales directamente al entorno físico, por lo que al Arduino se lo puede considerar como una interfaz de potencia en este caso.

Las figuras 47-3 y 48-3 muestran la conexión de la tarjeta Arduino con los drivers de los motores paso a paso. Se realizó un diagrama con recursos de la plataforma de dibujo electrónico Fritzing con la finalidad de facilitar el proceso de implementación.

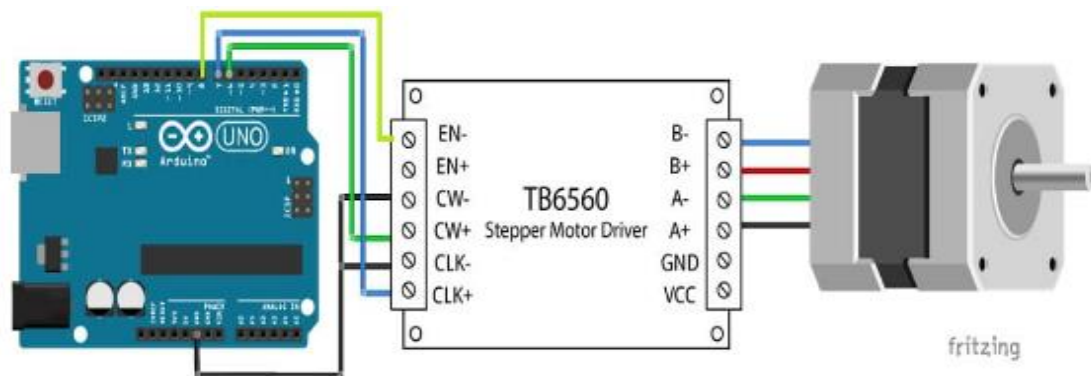


Figura 32-3: Diagrama de Conexión Arduino - Driver BL-TB6560-V2.0 – NEMA 17

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

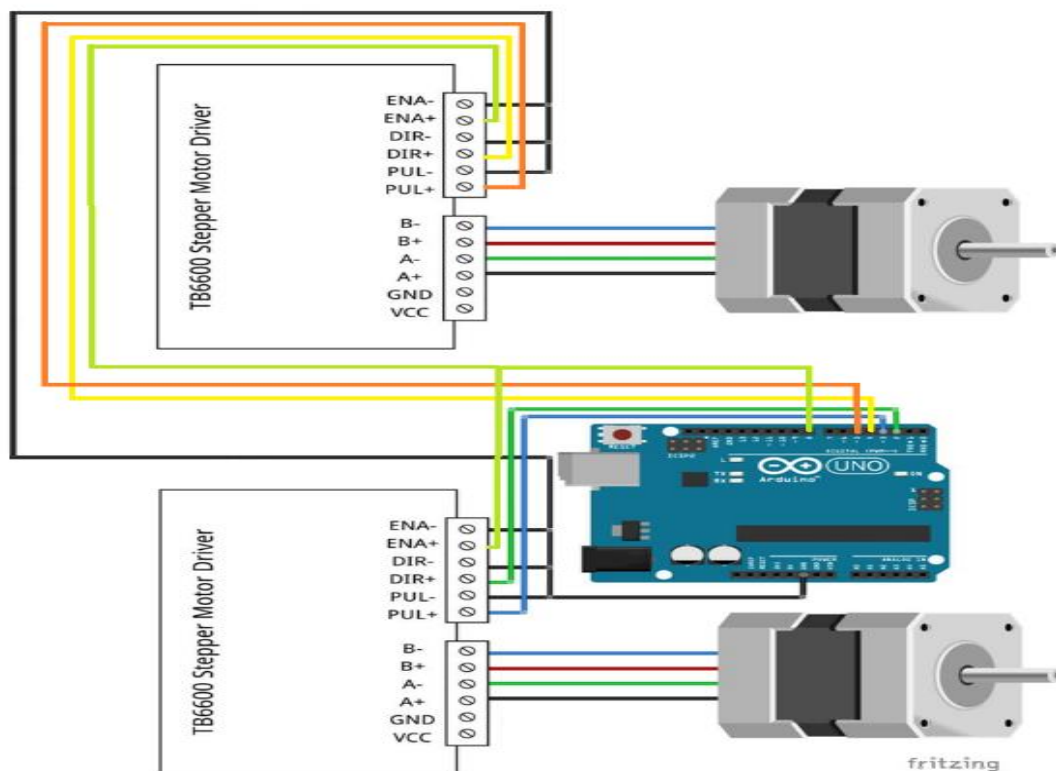


Figura 33-3: Diagrama de Conexión Arduino - Driver DM542T – NEMA 23

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 49-3 muestra el diagrama de conexiones.

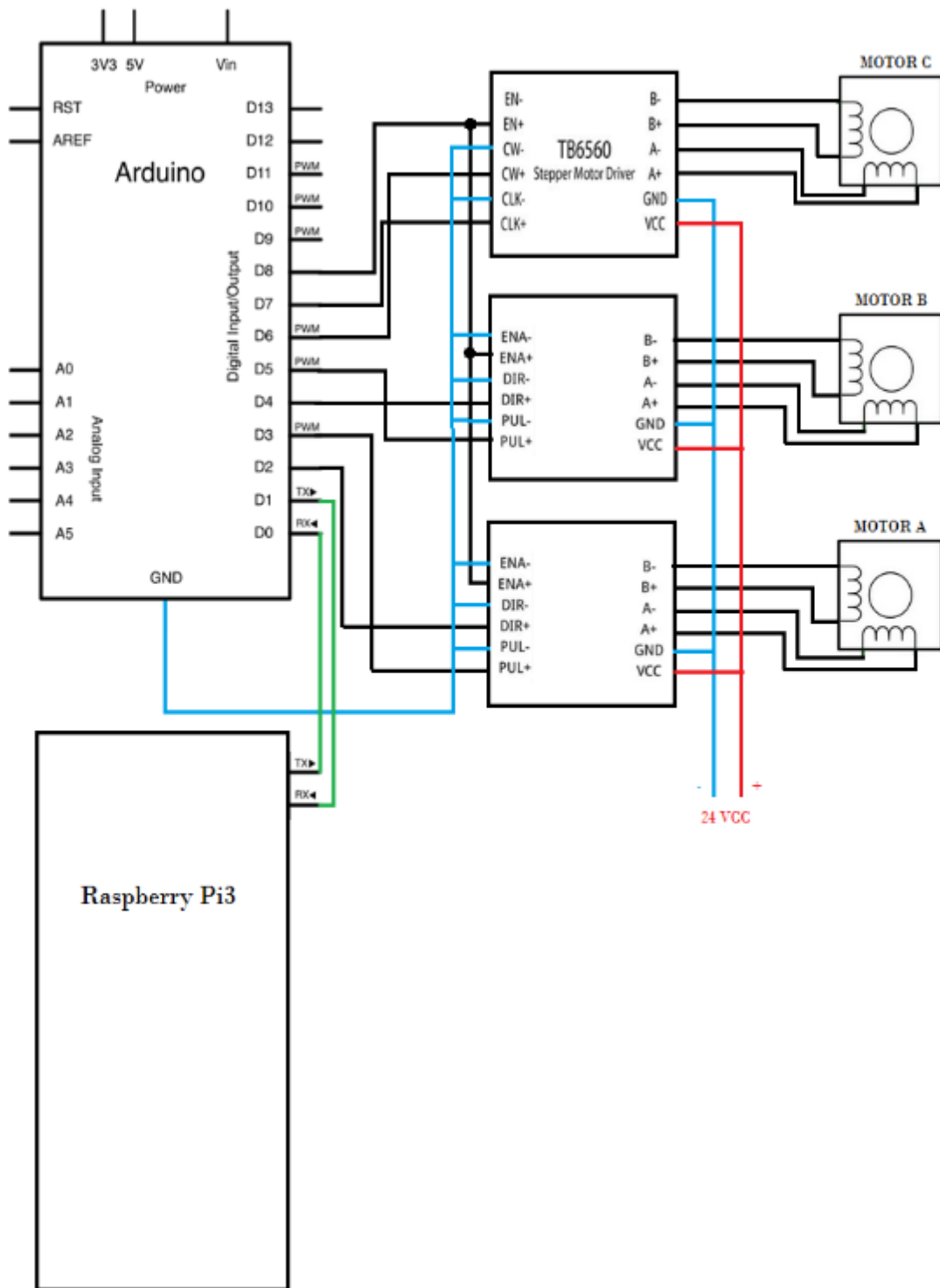


Figura 34-3: Diagrama eléctrico del prototipo

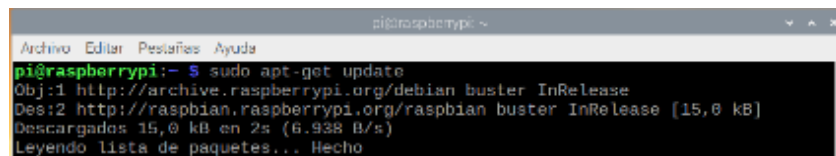
Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.6 Software del prototipo rehabilitador

En esta sección denominada software se describe la secuencia de programación ejecutada bajo las plataformas Python, LabVIEW y SolidWorks empleadas para definir acciones de configuración del hardware, enlaces de conectividad, definición de protocolos y formas de comunicación empleadas en el desarrollo del prototipo.

3.3.6.1 Carga del IDE de Arduino en la Raspberry

Para la interacción y programación del microcontrolador albergado en la plataforma embebida Arduino se procedió a la instalación del paquete del IDE de Arduino a la Raspberry Pi3, como se muestra en la figura 50-3, mediante “*sudo apt-get update*”, empleado para actualizar la lista de paquetes disponibles del sistema operativo Linux Raspbian.

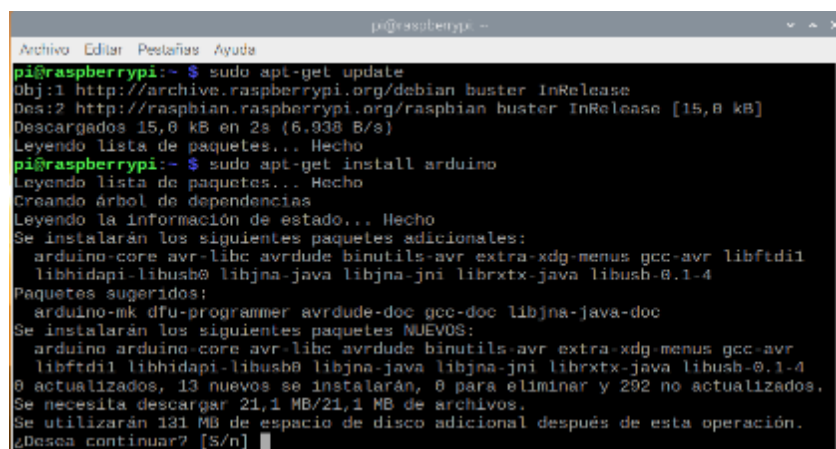


```
pi@raspberrypi:~$ sudo apt-get update
Obj:1 http://archive.raspberrypi.org/debian buster InRelease
Des:2 http://raspbian.raspberrypi.org/raspbian buster InRelease [15,0 kB]
Descargados 15,0 kB en 2s (6.938 B/s)
Leyendo lista de paquetes... Hecho
```

Figura 35-3: Actualización paquetes en la Raspberry.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Posterior a la actualización de paquetes se escribió el comando “*sudo apt-get install arduino*” como se muestra en la figura 51-3. Comando que sirve para descargar e instalar el paquete de Arduino del servidor de Linux que incluirá el IDE y el conjunto de drivers para las diferentes presentaciones de la tarjeta.



```
pi@raspberrypi:~$ sudo apt-get install arduino
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  arduino-core avr-libc avrdude binutils-avr extra-xdg-menus gcc-avr libftdi1
  libhidapi-libusb0 libjna-java libjna-jni librxtx-java libusb-0.1-4
Paquetes sugeridos:
  arduino-mk dfu-programmer avrdude-doc gcc-doc libjna-java-doc
Se instalarán los siguientes paquetes NUEVOS:
  arduino-core avr-libc avrdude binutils-avr extra-xdg-menus gcc-avr
  libftdi1 libhidapi-libusb0 libjna-java libjna-jni librxtx-java libusb-0.1-4
0 actualizados, 13 nuevos se instalarán, 0 para eliminar y 292 no actualizados.
Se necesita descargar 21,1 MB/21,1 MB de archivos.
Se utilizarán 131 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

Figura 36-3: Instalación paquete de Arduino.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Una vez instalado el paquete de Arduino se procedió a asegurar que el usuario “pi” identificado para la Raspberry disponga de los permisos necesarios para acceder al puerto serie. Como se observa en la figura 52-3. Se añadió el usuario “pi” a la terminal “tty” del grupo “dialout” por medio de la ejecución de los comandos “*sudo usermod -a -G tty pi*” y “*sudo usermod -a -G dialout pi*”. Acciones que deberán efectuarse previo a la ejecución del IDE Arduino para activar el puerto serie en el menú de Arduino.

```
pi@raspberrypi:~ $ sudo usermod -a -G tty pi
pi@raspberrypi:~ $ sudo usermod -a -G dialout pi
```

Figura 37-3: Activación del puerto serial.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

A modo de test, se ejecutó el comando “*ls /dev/tty**”, como se observa en la figura 53-3. Se despliega una lista con el siguiente texto “*/dev/ttyAMA0*” siendo esto el indicador que ha sido instalado con éxito el puerto serie sin que la placa de Arduino esté conectada.

```
pi@raspberrypi:~ $ ls /dev/tty*
/dev/tty      /dev/tty19  /dev/tty3   /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty8     /dev/tty2   /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty1     /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty10    /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty11    /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty12    /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty13    /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty14    /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58
/dev/tty15    /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59
/dev/tty16    /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty17    /dev/tty28  /dev/tty39  /dev/tty5   /dev/tty60
/dev/tty18    /dev/tty29  /dev/tty4   /dev/tty50  /dev/tty61
pi@raspberrypi:~ $
```

Figura 38-3: Instalación paquete de Arduino.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.6.2 Creación de la cuenta y registro del dispositivo en Remot3 IT

Para acceder al uso de la plataforma de soporte IoT se procedió a acceder a la página de remot3.it y crear una cuenta como se muestra en la figura 54-3.

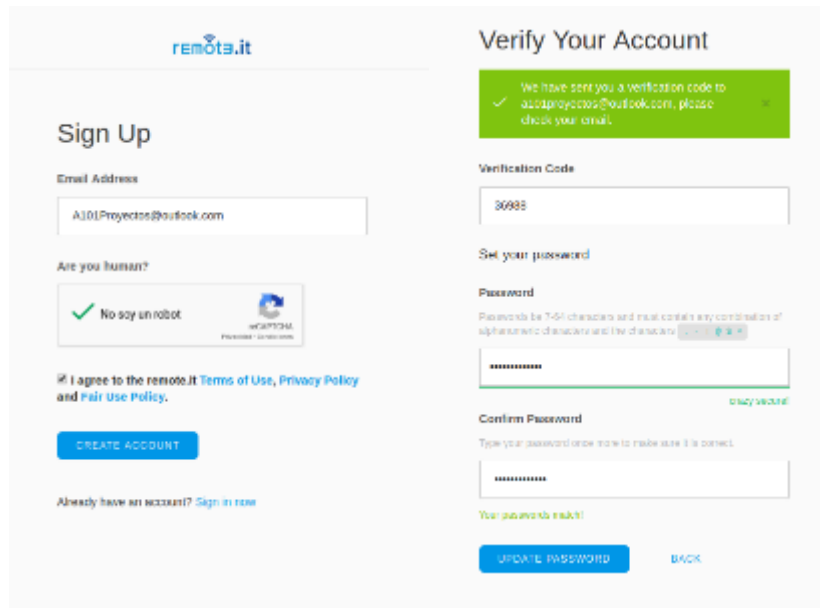


Figura 39-3: Creación de cuenta en la plataforma remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Una vez creada la cuenta, la plataforma permite el registro de los dispositivos que se desea anclar, como se observa en la figura 52-3. Para la tarjeta Raspberry Pi3 se añadió desde el terminal mediante comandos específicos, posterior a la selección en la plataforma Add Device.

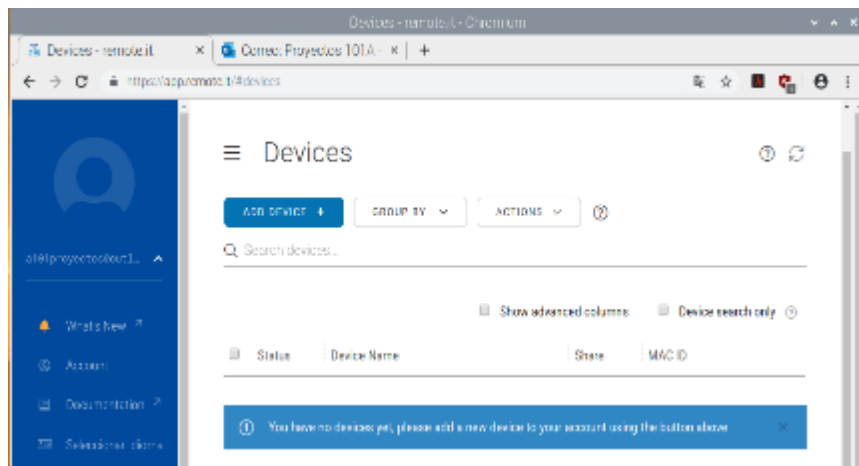


Figura 40-3: Inserción de un nuevo dispositivo en la plataforma remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

En la plataforma se debe especificar el tipo de dispositivo que se desea adicionar. En este caso una Raspberry modelo Pi3, como se muestra en la figura 56-3, luego se selecciona los comandos que deben ejecutarse paralelamente en la Raspberry para su registro.

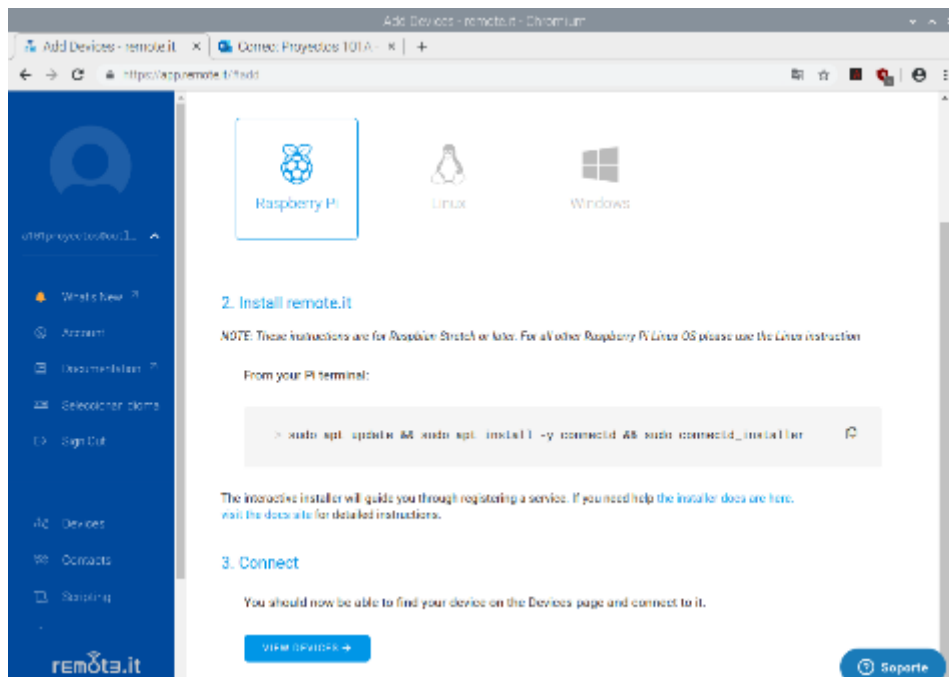


Figura 41-3: Definición del tipo de dispositivo a agregarse en la plataforma remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

En la figura 57-3 se muestra la ejecución del comando **“sudo apt update”** empleado para la actualización de la lista de paquetes disponibles del sistema operativo Linux Raspbian para el terminal de la Raspberry.

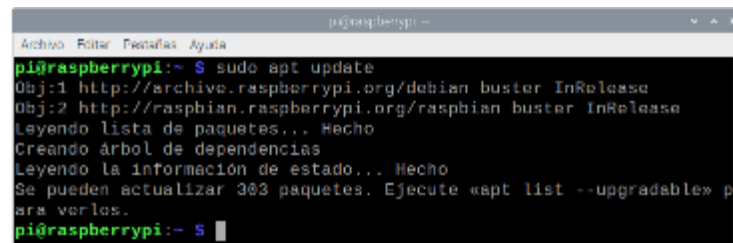


Figura 42-3: Actualización de paquetes en S.O Linux Raspbian.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Una vez actualizados los paquetes del sistema operativo se procedió a la descarga e instalación del paquete de Remot3 it mediante el comando **“sudo apt install -y connectd”** como se observa en la figura 58-3.

```
pi@raspberrypi:~$ sudo apt install -y connectd
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  connectd
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 303 no actualizados.
Se necesita descargar 102 kB de archivos.
Se utilizarán 0 B de espacio de disco adicional después de esta operación.
Des:1 http://archive.raspberrypi.org/debian buster/main armhf connectd armhf 2.3.17 [102 kB]
Descargados 102 kB en 3s (37,8 kB/s)
Seleccionando el paquete connectd previamente no seleccionado.
(Leyendo la base de datos ... 157159 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../connectd 2.3.17 armhf.deb ...
Desempaquetando connectd (2.3.17) ...
Progreso: [ 26%] [#####.....]
```

Figura 43-3: Instalación del paquete de Remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para la configuración del dispositivo se ejecuta el comando “*sudo connectd_installer*”, para relacionar la Raspberry con la plataforma IoT. En la figura 59-3 se despliega una lista de opciones de las cuales se eligió *Sign in to your existing remote.it account* para enlazar el dispositivo a la cuenta previamente creada en remot3 it.

```
pi@raspberrypi:~$ sudo connectd_installer
remote.it connection installer Version: 2.3.17 lib v2.1.17
Modified: April 15, 2019 (library) April 17, 2019
Build date: Thu Apr 18 18:21:59 PDT 2019
curl with SSL support installed.

Checking the daemon for compatibility...

Using architecture arm-linux-pi...

Checking your network for compatibility...

Network connection OK...
Your network is compatible with remote.it services.

***** Sign In Menu *****

  1) Sign in to your existing remote.it account
  2) Request a code for a new remote.it account
  3) Enter a verification code received in e-mail
  4) Exit

*****

Choose a menu selection (1 - 4):
```

Figura 44-3: Formas de enlazar el dispositivo a la plataforma remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Posteriormente se ejecutó un proceso de autenticación mediante el ingreso de usuario y contraseña de la cuenta de remot3 it generada, como se muestra en la figura 60-3.

```
***** Sign In Menu *****
1) Sign in to your existing remote.it account
2) Request a code for a new remote.it account
3) Enter a verification code received in e-mail
4) Exit
*****
Choose a menu selection (1 - 4):
1
Please enter your remote.it Username (e-mail address):
A101Proyectos@outlook.com
Please enter your remote.it password:
```

Figura 45-3: Ingreso a la plataforma remot3 it desde la Raspberry – Autenticación.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para añadir el nuevo dispositivo a la plataforma IoT se debe ingresar un nombre para identificación del dispositivo en la cuenta de remote3 it, “Raspberry Pi”, como se muestra en la figura 61-3.

```
Enter a name for your device (e.g. my_Pi_001).
The Device Name identifies your device in the remote.it portal.
Your services will be grouped under the Device Name.
Only letters, numbers, underscore, space and dash are allowed.
Raspberry Pi
```

Figura 46-3: Identificación del dispositivo a la plataforma IoT.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Establecido el nombre del dispositivo se despliega un nuevo menú como se observa en la figura 62-3 del cual se seleccionó la opción *Attach/reinstall a remote.it Service to an application*, que permite el levantamiento de un servicio.

```
----- Main Menu -----
1) Attach/reinstall a remote.it Service to an application
2) Attach/reinstall a remote.it Service to a LAN application
3) Remove a remote.it Service from an application
4) Remove all remote.it Services, then exit
5) Exit
-----
'application' is any TCP service (e.g. SSH, VNC, HTTP, etc.)
running on this device. 'LAN application' is a TCP service
running on another device on this LAN.
-----
Choose a menu selection (1 - 5):
```

Figura 47-3: Levantamiento de un servicio en remot3 it

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Sobre la activación del servicio ejecutada previamente se procedió a dar de alta el protocolo TCP, como se observa en la figura 63-3 se seleccionó la opción número 10, *Custom (TCP)*.

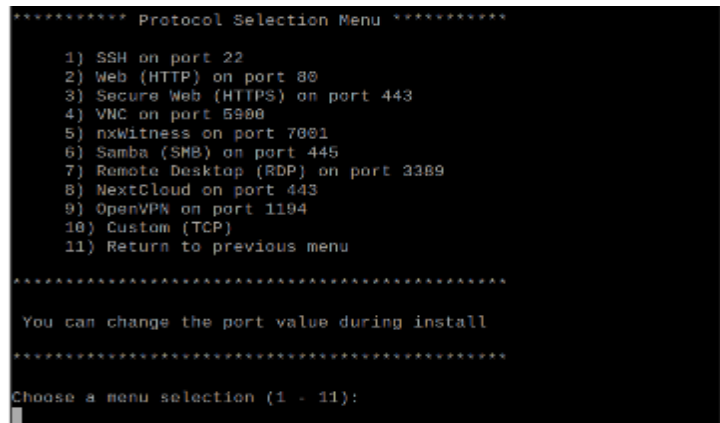


Figura 48-3: Definición del servicio

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Definido el tipo de servicio que se activó se debe asignar una identificación por medio de un nombre como se observa en la figura 64-3. Siendo este el último paso dentro de la configuración del dispositivo para enlazarlo a la cuenta de remot3 it se procede al cierre del terminal.

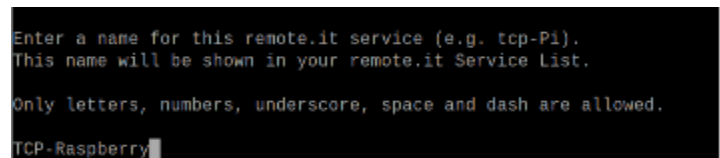


Figura 49-3.: Asignación de nombre al servicio activado

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para verificar la efectividad del proceso de registro del dispositivo en la cuenta de remot3 it se regresó al entorno de la plataforma IoT donde aparece registrado el dispositivo. La figura 65-3 muestra el proceso de registro y se verifica que el dispositivo de este trabajo se ha agregado correctamente.

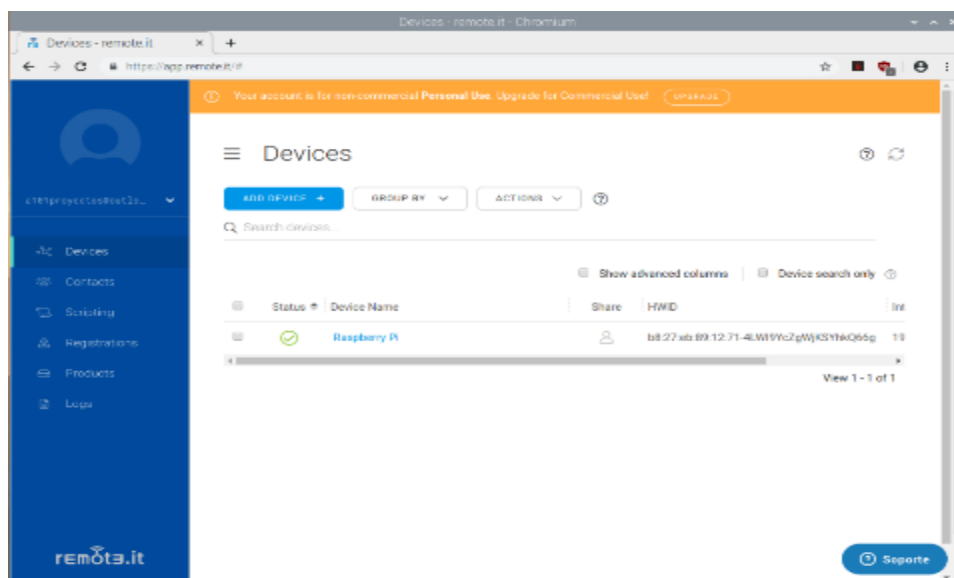


Figura 50-3: Verificación del dispositivo agregado a la cuenta de remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.6.3 Programación Microcontrolador

a) Definición de variables

El programa en el IDE de Arduino contiene las variables requeridas para la recepción y envío de datos así como la declaración de los pines del Arduino para la gestión de señales de control y el manejo de cada uno de los actuadores del rehabilitador, lo descrito se lo observa en la figura 66-3.

```
//definiendo variables y/o constantes //variables para el envio de angulos //Tiempo de calibrado
int angulo=0; //Tiempo de calibrado
String angle=""; int t=6000;
String str1=""; String angulo_str="";
String str2=""; String informacion=""; //Tiempo de Maniobra
String str3=""; String maniobra=""; //Maniobra A
//Variables para comparacion de sentencias //definiendo pines para el control de los motores a pasos
int x=0; //Motor A int t1=6000;
int y=1; int PUL_A=3; //Pin para la señal de pulso //Maniobra B
int z=1; int DIR_A=2; //define Direction pin int t2=6000;
int val=0; //Motor B
int direc=0; int PUL_B=5; //Pin para la señal de pulso //Maniobra C
int giro=0; int DIR_B=4; //define Direction pin int t3=6000;
int pasos=0; //Motor C
int repet=0; int PUL_C=7; //Pin para la señal de pulso
int cont=1; int DIR_C=6; //define Direction pin
int a=1; //
int m=0; int EN_ABC=8; //define Enable Pin de los 3 Motores a paso
int p=0; //
int q=0; int boton=10; //Pin detener el Motor a paso
int carrera=12; //Fin Fin de carrera
//
```

Figura 51-3: Declaración de variables en el IDE de Arduino

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

b) Inicio y declaración de funcionalidad de recursos del prototipo

En el bloque de la función `setup()`, se especifica la velocidad de comunicación con la que se relacionará el microcontrolador con la Raspberry. Se consideró el valor de 9600 que debe ser el mismo que se declara en la configuración del puerto de la Raspberry, como se observa en la figura 67-3.

```

void setup()
{
  Serial.begin(9600);

  //Motor A
  pinMode(PUL_A, OUTPUT);
  pinMode (DIR_A, OUTPUT);

  //Motor B
  pinMode(PUL_B, OUTPUT);
  pinMode (DIR_B, OUTPUT);

  //Motor C
  pinMode(PUL_C, OUTPUT);
  pinMode (DIR_C, OUTPUT);

  //Motores ABC
  pinMode (EN_ABC, OUTPUT);
  digitalWrite(EN_ABC,LOW);

  //Boton y final de carrera
  pinMode (boton, INPUT);
  pinMode (fcarrera, INPUT);
}

```

Figura 52-3: Función setup()

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

c) Adquisición y procesamiento de información del puerto serial.

La programación para la recepción de datos, se realiza carácter por carácter con una trama que contiene los valores de: sentido de giro, rango de movimiento (ángulo) y número de repeticiones. Para la decodificación de la información en el microcontrolador se utilizó el algoritmo que se muestra en la figura 68-3. Se lee y almacena el carácter que llega hasta encontrar un caracter “/” asignado en la programación de la Raspberry para separar los datos de cada una las variables, esta información se almacena en las variables str1, str2 y str3 respectivamente.

```

void loop()
{
  ////////////////////////////////////////////////////
  if (Serial.available())
  {
    char c = Serial.read();

    ////////////////////////////////////////////////////deteccion de separad
    if ((c=='/') && (y==1))
    {
      z=2;
      x=1;
    }
    if ((c=='/') && (y==2))
    {
      z=4;
      x=2;
    }
    if ((c=='/') && (y==3))
    {
      z=7;
      x=0;
      y=1;
    }

    //////////////////////////////////////////////////// adición de caract ////////////////////////////////////////////////////salto de lectura
    if(z==1)
    {
      str1 = str1+c;
      direc=c;
    }
    if(z==3)
    {
      str2 = str2+c;
    }
    if (z==5)
    {
      str3 = str3+c;
    }

    if(x==1)
    {
      z=3;
      y=2;
    }
    if(x==2)
    {
      z=5;
      y=3;
    }
  }
}

```

Figura 53-3: Decodificación de la información adquirida

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

d) Asignación de acciones a los actuadores.

Una vez adquirida y decodificada la información del puerto serial se procedió a programar las acciones para cada uno de los tres motores paso a paso que incluye el prototipo rehabilitador electromecánico. Dentro del funcionar del equipo hay varias acciones diferenciadas que relacionan el sentido de giro, rangos de alcance entre otros parámetros por lo que se consideró incluir en la programación un menú por medio de la estructura CASE para una mejor manipulación como se puede observar en la figura 66-3. El microcontrolador de acuerdo a la opción reconocida en la estructura CASE ejecuta el llamado de una función específica optimizando la respuesta.

```
if(z==7)
{
    val=str1.toInt();           //direccion de str a Int
    angle=str2;                //ángulo en str
    giro=str3.toInt();         //ángulo de str a Int
    repet=str3.toInt();        //Número de repeticiones para la maniobra de str a int
    pasos=map(giro,0,360,0,3200); //Conversion de angulos a pasos

    switch(direc)
    {
        case '1':
            pulsos_A();
            break;

        case '2':
            pulsos_A();
            break;

        case '3':
            pulsos_B();
            break;

        case '4':
            pulsos_B();
            break;

        case '5':
            pulsos_C();
            break;

        case '6':
            pulsos_C();
            break;

        case '7':
            pulsos_AA();
            break;

        case '8':
            pulsos_BB();
            break;

        case '9':
            pulsos_CC();
            break;
    }

    //////////reestablece la lectura
    z=1;
    //////////borra datos almacenados
    str1="";
    str2="";
    str3="";
}
//////////
}
//////////
}
```

Figura 54-3: Selección de acciones – Estructura CASE

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La tabla 50-3 presenta el listado de funciones declaradas dentro de la estructura CASE que se rigen a las acciones de calibración y maniobra de los actuadores eléctricos que gobiernan el rehabilitador. Estas acciones son controladas por el o los usuarios del equipo.

Tabla 50-3: Procesos del CASE

Calibración	Case		Maniobra	Case
pulsos_A()	1	(Retroceso)	pulsos_AA()	7
	2	(Avance)		
pulsos_B()	3	(Retroceso)	pulsos_BB()	8
	4	(Avance)		
pulsos_C()	5	(Retroceso)	pulsos_CC()	9
	6	(Avance)		

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Según el caso seleccionado sea maniobra o calibración, el motor a paso se moverá en un sentido según los grados ingresados por el usuario, de esta forma en la figura 70-3 se muestran las tres funciones que gobernarán la etapa de calibración del rehabilitador.

```

void pulsos_A()
{
    delay(500);

    if (val==1)
    {
        digitalWrite(DIR_A,LOW);
    }
    if (val==2)
    {
        digitalWrite(DIR_A,HIGH);
    }
    for (int i=0; i<pasos; i++)
    {
        digitalWrite(PUL_A,HIGH);
        delayMicroseconds(t);
        digitalWrite(PUL_A,LOW);
        delayMicroseconds(t);
    }
}

void pulsos_B()
{
    delay(500);

    if (val==3)
    {
        digitalWrite(DIR_B,LOW);
    }
    if (val==4)
    {
        digitalWrite(DIR_B,HIGH);
    }
    for (int i=0; i<pasos; i++)
    {
        digitalWrite(PUL_B,HIGH);
        delayMicroseconds(t);
        digitalWrite(PUL_B,LOW);
        delayMicroseconds(t);
    }
}

void pulsos_C()
{
    delay(500);

    if (val==5)
    {
        digitalWrite(DIR_C,LOW);
    }
    if (val==6)
    {
        digitalWrite(DIR_C,HIGH);
    }
    for (int i=0; i<pasos; i++)
    {
        digitalWrite(PUL_C,HIGH);
        delayMicroseconds(t);
        digitalWrite(PUL_C,LOW);
        delayMicroseconds(t);
    }
}

```

Figura 55-3: Funciones para calibración

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

El modo de maniobra es la opción que se ofrece a los tres motores para ejecución de los movimientos de la rehabilitación, la figura 71-3 muestra el código de la maniobra del motor A del prototipo rehabilitador, que ejecuta su ciclo de movimiento según el ángulo definido un determinado número de repeticiones que son ingresadas por el usuario o hasta que se detecte una señal de Pare/Final de carrera.

```

//////////////////////////////////Motor A

void pulsos_AA()
{
  delay(500);
  cont=1;
  a=1;
  maniobra="/A:";
  informacion="";
  p=0;
  q=0;
  while (cont<=repet)
  {
    ////////////////////////////////////
    for (int b=1; b<=2; b++)
    {
      ////////////////////////////////////
      if (b==1)
      {
        digitalWrite(DIR_A,LOW);
        int i=1;
        while ((i<=pasos)&&(p==0)&&(q==0))
        {
          if (digitalRead(boton))
          {
            p=1;
            break;
          }
        }

        if (digitalRead(fcarrera))
        {
          q=1;
          break;
        }
        else
        {
          m=i;
          imprimir();
          digitalWrite(PUL_A,HIGH);
          delayMicroseconds(t1);
          digitalWrite(PUL_A,LOW);
          delayMicroseconds(t1);
          i=i+1;
        }
      }
    }
    ////////////////////////////////////

    if (b==2)
    {
      digitalWrite(DIR_A,HIGH);
      int i=pasos;
      while ((i>=1)&&(p==0)&&(q==0))
      {
        if ( digitalRead(boton))
        {
          p=1;
          break;
        }
        if (digitalRead(fcarrera))
        {
          q=1;
          break;
        }
        else
        {
          m=i;
          imprimir();
          digitalWrite(PUL_A,HIGH);
          delayMicroseconds(t1);
          digitalWrite(PUL_A,LOW);
          delayMicroseconds(t1);
          i=i-1;
        }
      }
      ////////////////////////////////////
      delay(1000);
    }
    ////////////////////////////////////
    if (p==1)
    {
      Serial.println("STOP");
      break;
    }
    ////////////////////////////////////
    if (q==1)
    {
      Serial.println("CALIBRE-NUEVAMENTE");
      break;
    }
    ////////////////////////////////////
    cont = cont+1;
    ////////////////////////////////////
  }
  maniobra="";
  informacion="";
}

```

Figura 56-3: Sentencias para ejecución de la maniobra del motor A

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

El código del microcontrolador contiene una etapa de impresión de datos. Los valores de ángulos son enviados cada cierto tiempo por el puerto serial para el respectivo monitoreo de los actuadores. La figura 72-3 representa la codificación para la impresión de los datos para el monitoreo de los actuadores.

```

//IMPRESION DE DATOS

void imprimir()
{
  //////////////// Impresion angulo inicial
  if (m==1)
  {
    angulo =0;
    angulo_str= String (angulo);
    informacion=maniobra + angulo_str+"";
    Serial.println(informacion);
  }

  if (a==89)
  {
    //////////////// conversion de pasos a angulos

    angulo=map(m,0,3200,0,360);

    //////////////// impresion cada cierto paso

    angulo_str = String (angulo);
    informacion=maniobra + angulo_str+"";
    Serial.println(informacion);
    a=1;
  }
  a=a+1;

  ////////////////Impresion angulo final

  if (m==pasos)
  {
    delay(1);
    informacion=maniobra + angle+"";
    Serial.println(informacion);
    a=1;
    delay(1);
  }

  ////////////////
}

```

Figura 57-3: Impresión de datos para monitoreo de actuadores

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.6.4 Programación en Python

a) Instalación de librerías para desarrollo de la interfaz gráfica y comunicación

Para el desarrollo de la interfaz gráfica de la aplicación informática implementada sobre la Raspberry Pi3, para el manejo y configuración del prototipo rehabilitador electromecánico se empleó el lenguaje de programación Python, por características y versatilidad. Es considerado como de multiparadigma debido a que brinda soporte para programación orientada a objetos, programación imperativa y, en menor medida, programación funcional. (Oliphant, 2007, pp. 10-20)

El desarrollo de la interfaz gráfica para la aplicación informática tiene como objetivo que el usuario del equipo pueda manipularlo sin tener ningún conocimiento técnico. Se definen los recursos necesarios dentro de la programación para poder comunicar a la Raspberry PI3 con el medio externo.

La figura 73-3 representa el encabezado de la programación empleada en Python desarrollada en la Raspberry PI3 donde se puede observar que inicialmente se definen las librerías necesarias:

- **“import serial”, “import time”**: Importan el módulo de recurso para permitir el flujo de información a través de comunicación serial.
- **“import Tkinter”, “import tkMessageBox”**: Importa el módulo y recursos para el desarrollo de la interfaz gráfica.
- **“import socket”**: Importa el módulo para la conexión al acceso remoto.
- **“import threading”**: Importa el módulo para trabajar por hilos, lo que representa la ejecución de bloques de programas por secuencias ordenadas.

```
1 #!/usr/bin/python
2 import serial
3 import time
4 #importar el modulo para la interfaz grafica
5 import Tkinter as tk
6 from Tkinter import *
7 import tkMessageBox
8 #importar el modulo para la conexion
9 import socket
10 #importar el modulo de hilos
11 import threading
```

Figura 58-3: Importación de módulos.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Importados los módulos para el desarrollo del programa se procedió a la realización de dos actividades evidenciadas en la figura 74-3, como:

- Habilitar el puerto serial y establecer la velocidad de comunicación que se va a emplear con el elemento de enlace hacia el dispositivo electromecánico.
- La declaración de variables para definir acciones en el algoritmo para el control de procesos, la comparación de sentencias y la gestión de información con la plataforma remot3it. Las sentencias implementadas para la definición de las variables en Python, no solicita establecer el tipo de dato, se asigna las ubicaciones para el almacenamiento de información.

```
13 serial_Arduino=serial.Serial('/dev/ttyACM0',9600)
14 serial_Arduino.flushInput()
15
16 calibrar=1
17 manobra=1
18 datos=""
19 cant=0
20 signal=0
21 env1=0
22 env2=0
23 x=0
24 y=0
25 contador=0
26 a=0
27
```

Figura 59-3: Declaración de variables y habilitación del puerto serial.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

b) Programación modos de funcionamiento del prototipo

En la programación del algoritmo se procede a definir el código para ejecutar los modos de funcionamiento del prototipo rehabilitador.

Para la etapa de calibración del prototipo como se muestra en las figuras 75-3, 76-3 y 77-3 se generaron dos funciones por cada motor, estos bloques de programación son para las funciones de avance y retroceso de los motores de paso que permiten en cierto modo encerrar el prototipo creando puntos de partida para la ejecución de las rutinas de los motores establecidas en una siguiente etapa.

```
47 def avanzar_1 ():
48     if calibrar==0:
49         if maniobra==1:
50             global giro
51             giro="2"
52             global angulo
53             angulo=etiquetas11.get()
54             global repeticion
55             repeticion="0"
56             validar1()
57         else:
58             tkinterMessageBox.showwarning("Atencion", "Bloquear la Maniobra")
59     else:
60         print ("Data no enviado a calibracion")
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Figura 60-3: Modo Calibración motor A.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Mediante una sentencia de comparación se evalúa el modo de funcionamiento del prototipo rehabilitador, se considera que si está operando en el modo de calibración no puede funcionar en modo maniobra, esto se implementó como medida de seguridad acompañado de un mensaje de alerta, “Atención”, “Bloquear Maniobra”.

```
77 def avanzar_2 ():
78     if calibrar==0:
79         if maniobra==1:
80             global giro
81             giro="4"
82             global angulo
83             angulo=etiquetas22.get()
84             global repeticion
85             repeticion="0"
86             validar1()
87         else:
88             tkinterMessageBox.showwarning("Atencion", "Bloquear la Maniobra")
89     else:
90         print ("Data no enviado a calibracion")
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

Figura 61-3: Modo Calibración motor B.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

En los bloques de programación expuestos resulta importante la sentencia **“angulo=etiqueta22.get()”**, que representa el enlace de la codificación con la interfaz gráfica de la que se obtiene la información para establecer un desplazamiento angular en el motor a partir de la interacción del usuario.

```
92 def retroceder_3 ():
93     if calibrar==0:
94         if maniobra==1:
95             global giro
96             giro="5"
97             global angulo
98             angulo=etiqueta33.get()
99             global repeticion
100            repeticion="0"
101            validar1()
102        else:
103            tkinter.showwarning("Atencion","Bloquear la Maniobra")
104    else:
105        print ("Datos no enviados a calibracion")
106
107 def avanzar_3 ():
108     if calibrar==0:
109         if maniobra==1:
110             global giro
111             giro="8"
112             global angulo
113             angulo=etiqueta33.get()
114             global repeticion
115             repeticion="0"
116             validar1()
117         else:
118             tkinter.showwarning("Atencion","Bloquear la Maniobra")
119     else:
120        print ("Datos no enviados a calibracion")
121
```

Figura 62-3: Modo Calibración motor C.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Del mismo modo que las funciones de avance/retroceso en el modo de calibración se procedió a realizar las funciones necesarias para la ejecución de las maniobras con su respectiva evaluación de bloqueo como se muestra en la figura 78-3. En esta sección se enlaza la programación a la interfaz gráfica para obtener datos específicos para el proceso de rehabilitación como: el número de repeticiones para el proceso de desplazamiento angular de los motores, con el fin de generar e inducir los movimientos planteados dentro del protocolo de rehabilitación del paciente para su recuperación tras una afección del codo.

```

123 def ejercitarA():
124     if maniobra==0:
125         if calibrar==1 :
126             global giro
127             giro="7"
128             global angulo
129             angulo=etiqueta181.get()
130             global repeticion
131             repeticion=etiqueta104.get()
132             validar2()
133         else:
134             tkinterMessageBox.showwarning("Atencion","Bloquear el Calibrado")
135     else:
136         print ("Data no enviado a maniobra")
137
138 def ejercitarB():
139     if maniobra==0:
140         if calibrar==1 :
141             global giro
142             giro="8"
143             global angulo
144             angulo=etiqueta182.get()
145             global repeticion
146             repeticion=etiqueta105.get()
147             validar2()
148         else:
149             tkinterMessageBox.showwarning("Atencion","Bloquear el Calibrado")
150     else:
151         print ("Data no enviado a maniobra")
152
153
154 def ejercitarC():
155     if maniobra==0:
156         if calibrar==1 :
157             global giro
158             giro="9"
159             global angulo
160             angulo=etiqueta183.get()
161             global repeticion
162             repeticion=etiqueta106.get()
163             validar2()
164         else:
165             tkinterMessageBox.showwarning("Atencion","Bloquear el Calibrado")
166     else:
167         print ("Data no enviado a maniobra")
168

```

Figura 63-3: Modo Maniobra – Motores A, B, C.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

c) Controles en el ingreso de información

Para verificar la veracidad de la información ingresada a través de la interfaz gráfica, se generan funciones de validación como medios para conocer si el dato que ha sido ingresado a través de pantalla corresponde a un dígito o no, de esta manera se determina que la información a ser enviada a los procesos de calibración o maniobra sean válidos para fijar los desplazamientos angulares en los motores e igualmente establecer el número de repeticiones de un movimiento específico.

```

170 def validar1():
171     global val
172     val=angulo.isdigit()
173     if val==1:
174         limite1()
175     else:
176         tkinterMessageBox.showwarning("Cuidado","Valores Incorrectos : \n Ingrese angulos \n")
177
178 def validar2():
179     global val
180     val=angulo.isdigit()
181     global val2
182     val2=repeticion.isdigit()
183     if val==1 and val2==1:
184         limite2()
185     else:
186         tkinterMessageBox.showwarning("Cuidado","Valores Incorrectos : \nIngrese angulos/repeticiones\n")

```

Figura 64-3: Validación de información.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Como medida de prevención por mal uso del prototipo sea voluntaria o involuntariamente se creó funciones de control para que cuando sean ingresados los ángulos de desplazamiento angular de los motores o repeticiones se verifique que sean valores aceptables dentro de los rangos de calibración permitidos, como se observa en la figura 80-3.

```

189 def límite1():
190     global angle
191     angle= int(angulo)
192     if angle>=0 and angle<=25:
193         mover1()
194     else:
195         tkinter.messagebox.showwarning("ÁNGULO", " Ingrese valores \n del 0 al 25 ")
197
198 def límite2():
199     global angle
200     angle= int(angulo)
201     global repetition
202     repetition= int(repetition)
203     if angle>=0 and angle<=180:
204         if repetition>=1 and repetition<=5:
205             mover2()
206     else:
207         tkinter.messagebox.showwarning("REPETICION", " Ingrese valores \n del 1 al 5 ")
208
209     tkinter.messagebox.showwarning("ÁNGULO", " Ingrese valores \n del 0 al 180 ")
210

```

Figura 65-3: Validación de información “ángulo”, “repeticiones”.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

d) Impresión de información en el Puerto Serial

Se considera información válida aquella ingresada a través de la interfaz gráfica, se define una función exclusiva en la que se concatena y codifica toda la información sobre el giro, ángulo y repeticiones en una sola variable de tipo *cadena* para ser escrita en el puerto serial y direccionada hacia el microcontrolador con información únicamente de la sección de calibración. Se emplea en este caso un carácter “/” como separador entre cada dato concatenado para la fácil decodificación en el destino, como se muestra en la figura 81-3.

```

212 def mover1():
213     global cadena
214     cadena=giro+"/"+angulo+"/"+repeticion="/"
215     print (cadena)
216     comando_utf = cadena.encode()
217     serial_arduino.write(comando_utf)
218     time.sleep(0.1)

```

Figura 66-3: Impresión en el Puerto Serial - Información solo para calibración

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

De la misma manera, se creó una función que envía los datos de maniobra y genera una llamada a un hilo de programación el cual se efectuará en segundo plano.

```

221 def mover2():
222     global x
223     x=1
224     global cadena
225     cadena=giro+"/"+angulo+"/"+repeticion+"/"+
226     print (cadena)
227     comando_utf = cadena.encode()
228     serial.Arduino.write(comando_utf)
229     time.sleep(0.1)
230     global a
231     if a==0 :
232         a=1
233         serial.Arduino.flushInput()
234         exitir2= threading.Thread(name="hilo_3", target=enviar2)
235         exitir2.start()

```

Figura 67-3: Impresión en el Puerto Serial - Información solo para maniobra

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 83-3 muestra la programación para la creación del hilo, el cual se encarga de la lectura continua del puerto serial para la recolección y envío de datos hacia la red para el respectivo monitoreo a través de la plataforma IoT.

```

238 def enviar2():
239     global env2
240     global envi
241     global datos
242     global x
243     global contador
244     global informacion
245     while env2==0:
246         try:
247             print(contador)
248             contador=contador+1
249             time.sleep(0.1)
250             while (serial.Arduino.inWaiting()>0):
251                 contador=0
252                 print(contador)
253                 informacion= serial.Arduino.readline()
254                 dt=informacion.rstrip('\n')
255                 dt_utf = dt.decode('utf-8')
256                 datos=dt_utf
257                 print (datos)
258                 @tiqueta78.config.text=datos
259                 enviar()
260                 if contador==100:
261                     @tiqueta78.config.text="Datos"
262                     reenviar()
263                 if contador==110:
264                     contador=101
265             except:
266                 print("No hubo datos")
267
268 def enviar():
269     try:
270         socket.setdefaulttimeout(1)
271     except:
272         @tiqueta58.config.text="Desconectado"
273         print ("Data No Enviada")
274
275 def reenviar():
276     global envi
277     global x
278     x=0
279     print ("Abrir envío")
280     #hilo de envío de datos(segundo plano)
281     exitir = threading.Thread(name="hilo 2", target=enviar2)
282     exitir.start()

```

Figura 68-3: Hilo de programación – Recolección y Gestión de información

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

e) *Conexión a la Plataforma IoT – Remot3 it*

Las siguientes líneas de código de la figura 84-3, evalúan la contraseña que se ingresa a través de la pantalla, en el caso de ser correcta se efectúa la llamada hacia la conexión con la plataforma Remot3-it.

```

298 def password():
299     if etiqueta44.get()=="internet":
300         call_me()
301     else:
302         tkMessageBox.showwarning("Cuidado","Password Incorrecto")
303
304 def call_me():
305     answer=tkMessageBox.askquestion("Conexion"," Desea continuar, presione 'Si' ")
306     if answer=="yes":
307         #hilo de conexion
308         conectar= threading.Thread(name="hilo_1", target=accept)
309         conectar.start()
310
311

```

Figura 69-3: Bloque de autenticación y conexión a la plataforma IoT

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Después se crea el hilo de programación que efectúa la conexión a Remot3 it para el respectivo envío de datos. Esta función es llamada a otro hilo de programación para ejecutarlo continuamente y evitar el corte de la comunicación con la plataforma

La ejecución de este hilo mediante líneas de programación se muestra en la figura 85-3

```

385 #Conexion: Hilo1 (Programacion paralela)
386 def accept():
387     global cont
388     global signal
389     global env
390     if cont==0:
391         cont=1
392         signal=1
393         env=0
394     global socket s
395     socket s = socket.socket()
396     global host
397     host = '
398     global port
399     port = 9999
400     global backlog
401     backlog = 5
402     socket s.bind ((host,port))
403     socket s.listen(backlog)
404     print ("ESPERANDO UNA CONEXION ... :|")
405     socket s, (host,port) = socket s.accept()
406     #hilo de envio de datos(segundo plano)
407     envtir = threading.Thread(name="hilo_2", target=enviar1)
408     envtir.start()
409     print ("CONEXION ESTABLECIDA ..... :|")
410     etiqueta59.config(text="Conectado")
411     else :
412         tkMessageBox.showwarning("Cuidado","Conexion Realizada")
413
414 def enviar1():
415     global env1
416     global informacion
417     global cont
418     global x
419     while (env1==0):
420         try:
421             informacion="LEYENDO"
422             socket s.send(informacion)
423             print('Dato enviado:LEYENDO')
424             time.sleep(1)
425             if (x==1):
426                 env1=1
427         except:
428             time.sleep(1)
429             print('Dato no enviado:LEYENDO')
430             etiqueta58.config(text="Desconectado")
431             env1=1
432             cont=0
433
434

```

Figura 70-3: Envío de datos a la plataforma IoT – Remot3 it.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

f) Controles para la ejecución de los procesos

La línea de programación de la figura 86-3 muestra el código que permite determinar las funciones de bloqueo/Desbloqueo de los procesos de calibración y maniobra. Es útil para controlar los posibles errores en la manipulación de la interfaz gráfica. Las acciones de estos bloques de

función vienen conjugadas con la impresión de mensajes de texto que orienten al uso correcto del prototipo.

```
356 def Habilitar1():
357     global calibrar
358     if calibrar==1 :
359         answer=tkMessageBox.askquestion("Desbloquear"," Desea continuar, presione 'SI' ")
360         if answer=='yes' :
361             etiqueta5.config(text="Desbloqueado")
362             etiqueta5.config(bg="light cyan")
363             BT_Habilitar1.config(text="Bloquear")
364             BT_Habilitar1.config(bg="orchid1")
365             calibrar=0
366         else:
367             answer=tkMessageBox.askquestion("Bloquear"," Desea continuar, presione 'SI' ")
368             if answer=='yes' :
369                 etiqueta5.config(text="Bloqueado")
370                 etiqueta5.config(bg="thistle1")
371                 BT_Habilitar1.config(text="Desbloquear")
372                 BT_Habilitar1.config(bg="cyan2")
373                 calibrar=1
374
375
376 def Habilitar2():
377     global maniobra
378     if maniobra==1 :
379         answer=tkMessageBox.askquestion("Desbloquear"," Desea continuar, presione 'SI' ")
380         if answer=='yes' :
381             etiqueta6.config(text="Desbloqueado")
382             etiqueta6.config(bg="light cyan")
383             BT_Habilitar2.config(text="Bloquear")
384             BT_Habilitar2.config(bg="orchid1")
385             maniobra=0
386         else:
387             answer=tkMessageBox.askquestion("Bloquear"," Desea continuar, presione 'SI' ")
388             if answer=='yes' :
389                 etiqueta6.config(text="Bloqueado")
390                 etiqueta6.config(bg="thistle1")
391                 BT_Habilitar2.config(text="Desbloquear")
392                 BT_Habilitar2.config(bg="cyan2")
393                 maniobra=1
394
```

Figura 71-3: Funciones de bloqueo/desbloqueo de los procesos de calibración y maniobra.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

g) *Generación de código para la creación de la Interfaz Gráfica*

Se determina las dimensiones deseadas para trabajar sobre una pantalla específica. La figura 87-3 muestra la configuración empleada para la pantalla de la interfaz gráfica del proyecto de dimensiones 1280x600, además que se le agrega un elemento gráfico para mejorar su estética.

```
397 #####
398
399 root = Tk()
400 root.title("Interfaz")
401 root.minsize(1280,600)
402 root.resizable(width=False, height=False)
403 #Color de fondo pantalla
404 #root.config(bg="aquamarine3")
405
406 img = PhotoImage(file="FOND0.png")
407 widget = Label(root, image=img)
408 widget.place(x=0,y=0,relwidth=1.0,relheight=1.0)
409
```

Figura 72-3: Creación de pantalla - medidas.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Todos los recursos gráficos empleados para el desarrollo de la interfaz gráfica como: botones y cuadros de texto se los organiza mediante la creación de etiquetas que permiten una buena organización dentro del algoritmo desarrollado para el correcto flujo de la información.

En la figura 88-3 se puede observar la sección del programa en la que se incluyen las sentencias necesarias para la declaración de etiquetas, a continuación, se describen las más relevantes:

img2=PhotoImage(file='x1.png'): La clase PhotoImage es usada para mostrar imágenes sean estas en escala de grises o RGB montadas sobre etiquetas o botones.

img2=img2.subsample(1,1): La clase subsample permite muestrear a la imagen y generar una escala.

widget2=label(root, image=img2): En esta línea de programación es un proceso de actualización de la imagen sobre la etiqueta asignada sobre el widget que en este caso maneja la forma de una caja de texto para un encabezado en la interfaz gráfica, puede presentarse también como un desplegable o una casilla de verificación.

widget2.grid(column=2, row=0): En esta línea de código se está asignando dentro de la caja del widget sobre qué posición se está trabajando en términos de columnas y filas.

```
418 #####Inagenes de apoyo
419
420 #####Horizontal
421
422 img2 = PhotoImage(file="x1.png")
423 img2 = img2.subsample(1,1)
424 widget2 = Label(root, image=img2)
425 widget2.grid(column=2, row=0)
426
427 img3 = PhotoImage(file="x2.png")
428 img3 = img3.subsample(1,1)
429 widget3 = Label(root, image=img3,bg="snow2")
430 widget3.grid(column=5, row=0)
431
432 img4 = PhotoImage(file="x3.png")
433 img4 = img4.subsample(1,1)
434 widget4 = Label(root, image=img4,bg="snow2")
435 widget4.grid(column=7, row=0)
436
437 img5 = PhotoImage(file="x4.png")
438 img5 = img5.subsample(1,1)
439 widget5 = Label(root, image=img5,bg="snow2")
440 widget5.grid(column=9, row=0)
441
442 img6 = PhotoImage(file="x5.png")
443 img6 = img6.subsample(1,1)
444 widget6 = Label(root, image=img6,bg="snow2")
445 widget6.grid(column=11, row=0)
446
447 img7 = PhotoImage(file="x6.png")
448 img7 = img7.subsample(1,1)
449 widget7 = Label(root, image=img7,bg="snow2")
450 widget7.grid(column=13, row=1)
451
452 #####Vertical
453
454 img8 = PhotoImage(file="y1.png")
455 img8 = img8.subsample(1,1)
456 widget8 = Label(root, image=img8,bg="gray68")
457 widget8.grid(column=8, row=0)
458
459 img9 = PhotoImage(file="y2.png")
460 img9 = img9.subsample(1,1)
461 widget9 = Label(root, image=img9,bg="gray74")
462 widget9.grid(column=8, row=2)
463
464 img10 = PhotoImage(file="y3.png")
465 img10 = img10.subsample(1,1)
466 widget10 = Label(root, image=img10,bg="gray78")
467 widget10.grid(column=8, row=5)
468
469 img11 = PhotoImage(file="y4.png")
470 img11 = img11.subsample(1,1)
471 widget11 = Label(root, image=img11,bg="gray85")
472 widget11.grid(column=8, row=7)
```

Figura 73-3: Asignación etiquetas a recursos de la interfaz gráfica.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

El bloque descrito anteriormente se lo determinó como referencia en el programa como imágenes de apoyo pues representa parte del diseño estético de la interfaz gráfica.

La figura 89-3 describe el proceso de ingreso de etiquetas y variables de entrada de datos por medio de la pantalla o interfaz gráfica para establecer acciones del modo de calibración de los motores de paso del rehabilitador.

```

475 #####TITULO#####
476
477 etiqueta01 = Label(root, text="Calibrar",width="14", relief=RAISED,bg="goldenrod1", pady=5)
478 etiqueta01.grid(column=1, row=4)
479
480 ##### Motor A
481 var1 = StringVar()
482 etiqueta1 = Label(root, textvariable=var1,width="14", relief=SOLID,bg="bisque", pady=5)
483 var1.set("Motor A")
484 etiqueta1.grid(column=1, row=6)
485
486 var11 = StringVar()
487 etiqueta11 = Entry(root, textvariable=var11 ,width="18")
488 etiqueta11.grid(column=6, row=6)
489 ##### Motor B
490 var2 = StringVar()
491 etiqueta2 = Label(root, textvariable=var2,width="14", relief=SOLID,bg="bisque", pady=5)
492 var2.set("Motor B")
493 etiqueta2.grid(column=1, row=8)
494
495 var22 = StringVar()
496 etiqueta22 = Entry(root, textvariable=var22 ,width="18")
497 etiqueta22.grid(column=6, row=8)
498 ##### Motor C
499 var3 = StringVar()
500 etiqueta3 = Label(root, textvariable=var3,width="14", relief=SOLID,bg="bisque", pady=5)
501 var3.set("Motor C")
502 etiqueta3.grid(column=1, row=10)
503
504 var33 = StringVar()
505 etiqueta33 = Entry(root, textvariable=var33 ,width="18")
506 etiqueta33.grid(column=6, row=10)
507 #####
508 #####
509 #####
510 #####
511 #####

```

Figura 74-3: Asignación etiquetas y variables en la interfaz gráfica.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 90-3 muestra una sección del bloque de programación empleado para la creación de etiquetas para la identificación de valores para la contraseña, bloqueo / desbloqueo y ángulo.

```

513 #####Ingreso de Password
514
515 var4 = StringVar()
516 etiqueta4 = Label(root, textvariable=var4, width="18", relief=SOLID,bg="bisque", pady=5)
517 var4.set("Password")
518 etiqueta4.grid(column=10, row=1)
519
520 var44 = StringVar()
521 etiqueta44 = Entry(root, textvariable=var44 ,width="13",show="*")
522 etiqueta44.grid(column=12, row=1)
523
524 #####Etiqueta Bloqueo Calibrado
525
526 etiqueta5 = Label(root, text="Bloqueado",width="13", relief=FLAT,bg="thistle1", pady=5)
527 etiqueta5.grid(column=4, row=4)
528
529 #####Etiqueta Bloqueo Maniobra
530
531 etiqueta6 = Label(root, text="Bloqueado",width="13", relief=FLAT,bg="thistle1", pady=5)
532 etiqueta6.grid(column=12, row=3)
533
534 #####Etiqueta Angulo de calibrado
535
536 etiqueta7 = Label(root, text="Angulo",width="18", relief=SOLID,bg="khaki1", pady=5)
537 etiqueta7.grid(column=6, row=4)
538
539 #####Etiqueta Angulo de Maniobra
540
541 etiqueta8 = Label(root, text="Angulo",width="13", relief=SOLID,bg="khaki1", pady=5)
542 etiqueta8.grid(column=12, row=4)
543
544 #####

```

Figura 75-3: Asignación de etiquetas de valores de ingreso en la interfaz gráfica.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

En la figura 91-3, se muestra bloques de sentencias de un proceso de creación de etiquetas con el fin de representar variables de ingreso para datos del tipo de maniobra y repeticiones.

```

546 #####TITULO2#####
547
548 etiqueta82 = Label(root, text="Maniobra",width="14", relief=GROOVE,bg="goldenrod1", pady=5)
549 etiqueta82.grid(column=8, row=3)
550
551 #Maniobra Motor A
552
553 var18 = StringVar()
554 etiqueta18 = Label(root, textvariable=var18,width="14", relief=SOLID,bg="bisque", pady=5)
555 var18.set("Maniobra A")
556 etiqueta18.grid(column=8, row=5)
557
558 var181 = StringVar()
559 etiqueta181 = Entry(root, textvariable=var181,width="13")
560 etiqueta181.grid(column=12, row=6)
561
562 #Maniobra Motor B
563
564 var28 = StringVar()
565 etiqueta28 = Label(root, textvariable=var28,width="14", relief=SOLID,bg="bisque", pady=5)
566 var28.set("Maniobra B")
567 etiqueta28.grid(column=8, row=8)
568
569 var182 = StringVar()
570 etiqueta182 = Entry(root, textvariable=var182,width="13")
571 etiqueta182.grid(column=12, row=8)
572
573 #Maniobra Motor C
574
575 var38 = StringVar()
576 etiqueta38 = Label(root, textvariable=var38,width="14", relief=SOLID,bg="bisque", pady=5)
577 var38.set("Maniobra C")
578 etiqueta38.grid(column=8, row=10)
579
580 var183 = StringVar()
581 etiqueta183 = Entry(root, textvariable=var183,width="13")
582 etiqueta183.grid(column=12, row=10)
583
584 #####Numero de repeticiones para la maniobra#####
585 #####TITULO 3 #####
586
587
588 var48 = StringVar()
589 etiqueta48 = Label(root, textvariable=var48,width="13", relief=SOLID,bg="khaki1", pady=5)
590 var48.set("Repeticiones")
591 etiqueta48.grid(column=14, row=4)
592

```

Figura 76-3: Asignación de etiquetas a valores de ingreso en la interfaz gráfica - Maniobra.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para verificación de la comunicación y flujo de información a la plataforma IoT se declaró ciertos recursos que se pueden mostrar en la interfaz gráfica. La figura 92-3, muestra la sección del código empleado para creación del recurso.

```

609 #####Etiqueta de aviso de conexión#####
610
611 etiqueta50 = Label(root, text="Desconectado",width="13", relief=FLAT,bg="lightpink1", pady=5)
612 etiqueta50.grid(column=14, row=1)
613
614 #####
615 #####Etiqueta de Muestreo de datos#####
616
617
618
619 etiqueta60 = Label(root, text="Información",width="14", relief=SOLID,bg="bisque", pady=5)
620 etiqueta60.grid(column=8, row=4)
621
622 etiqueta70 = Label(root, text="Datos",width="18", relief=SOLID,bg="light yellow", pady=5)
623 etiqueta70.grid(column=10, row=4)
624
625 #####

```

Figura 77-3: Asignación etiquetas a valores de ingreso en la interfaz gráfica - Conectividad.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

En la interfaz gráfica se insertaron botones para emplearlos como señales pulsantes para el control de avance y retroceso al momento de calibrar los motores de paso del rehabilitador, la figura 93-3 muestra la codificación para la creación de los botones.

MA_retroceder=Botton(root, text="Retroceder", bg='DarkSeaGreen1), activebackground='green, relief = GROVEE, width='10', command=avanzar_1): Esta línea de código representa el modelo estructural empleado para la creación de todos los botones en el que se especifica: nombre, asignaciones de color, tamaño y procedimiento o función para ser ejecutado.

```

820 #####
829
830 ## Botones para el calibrado#####
831
832 ##### Motor A
833
834 #boton Retroceder
835 MA_retroceder = Button(root,
836 text=" Retroceder ",
837 bg='DarkSeaGreen1',
838 activebackground= 'green yellow',
839 relief=GROOVE,
840 width="10",
841 command=retroceder_1)
842 MA_retroceder.grid(column=1,row=6)
843 #boton Avanzar
844 MA_avanzar = Button(root,
845 text=" Avanzar ",
846 bg='DarkSeaGreen1',
847 activebackground= 'green yellow',
848 relief=GROOVE,
849 width="10",
850 command=avanzar_1)
851 MA_avanzar.grid(column=3,row=6)
852 #####
853 ##### Motor B
854 #boton Retroceder
855 MB_retroceder = Button(root,
856 text=" Retroceder ",
857 bg='DarkSeaGreen1',
858 activebackground= 'green yellow',
859 relief=GROOVE,
860 width="10",
861 command=retroceder_2)
862 MB_retroceder.grid(column=4,row=6)
863 #boton Avanzar
864 MB_avanzar = Button(root,
865 text=" Avanzar ",
866 bg='DarkSeaGreen1',
867 activebackground= 'green yellow',
868 relief=GROOVE,
869 width="10",
870 command=avanzar_2)
871 MB_avanzar.grid(column=3,row=7)
872 #####
873 ##### Motor C
874 #boton Retroceder
875 MC_retroceder = Button(root,
876 text=" Retroceder ",
877 bg='DarkSeaGreen1',
878 activebackground= 'green yellow',
879 relief=GROOVE,
880 width="10",
881 command=retroceder_3)
882 MC_retroceder.grid(column=4,row=7)
883 #boton Avanzar
884 MC_avanzar = Button(root,
885 text=" Avanzar ",
886 bg='DarkSeaGreen1',
887 activebackground= 'green yellow',
888 relief=GROOVE,
889 width="10",
890 command=avanzar_3)
891 MC_avanzar.grid(column=3,row=8)
892 #####
893 #####
894 #####
895 #####

```

Figura 78-3: Creación botones para el manejo de los motores en el proceso de calibración.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se creó un botón para iniciar la conexión y enviar datos a la plataforma IoT – Remot3 it empleando el mismo principio de creación de los botones anteriores como se puede observar en la figura 94-3.

```

797
798 #boton Conectar
799 Bconec = Button(root,
800                 text=" Conectar ",
801                 bg='turquoise',
802                 activebackground= 'olive#1',
803                 relief=GRAVITY,
804                 width='12',
805                 command=password)
806 Bconec.grid(column=0, row=1)

```

Figura 79-3: Creación de botón para inicio de conexión a la plataforma IoT

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Y finalmente se declara una función mostrada en la figura 95-3 que cierre el bucle de la interfaz gráfica, además de una condición que cancela la comunicación con plataforma Remot3 it y también finaliza la comunicación del puerto serial.

socket_s.close(): Cancela la comunicación con la plataforma IoT.

serial_Arduino.close(): Finaliza la comunicación del puerto Serial.

```

769 root.mainloop()
770
771 #####
772 env1=1
773 env2=1
774 if signal==1:
775     print ("DESCONECTANDO....")
776     global socket_s
777     socket_s.close()
778     print ("DESCONECTADO")
779     serial_Arduino.close()
780
781     print ("Proceso Finalizado")
782
783
784 #####
785 #####

```

Figura 80-3: Función que permite cerrar el bucle de la interfaz gráfica.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.7 Enlace LabVIEW y la plataforma Remote it

El punto de partida para trabajar en LabVIEW es la creación de un nuevo proyecto direccionado a un espacio de memoria específico donde se almacenarán todos los recursos a emplearse para la implementación del sistema de monitoreo remoto. La figura 96-3 muestra el proceso de creación de una nueva hoja de instrumentos virtuales o VI en la que se desarrolló el algoritmo para la vinculación de LabVIEW con la plataforma IoT.

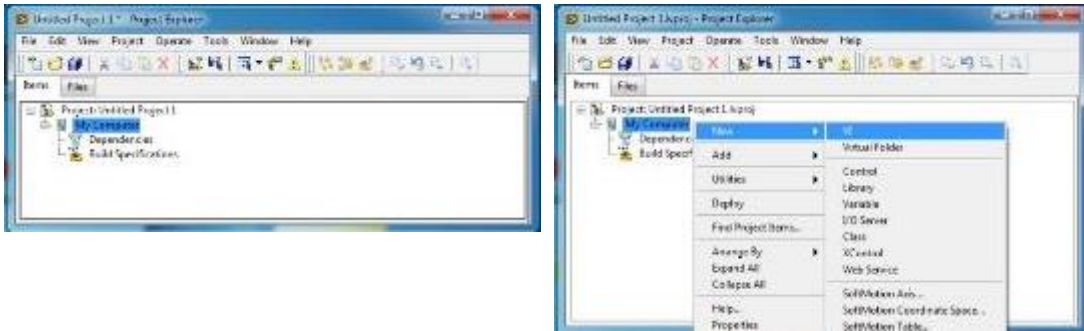


Figura 81-3: Creación Proyecto y VI.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para el enlace de LabVIEW hacia Remote it se emplea el protocolo de comunicación TCP, abriendo la posibilidad de un monitoreo remoto desde cualquier lugar del mundo donde tanto el equipo rehabilitador y de monitoreo estén conectados a la internet. La figura 97-3 muestra el conjunto de bloques de programación empleados para cumplir con este propósito. Se puede observar que se apertura una conexión de red TCP asignando una dirección y un puerto remoto como atributos del bloque *TCP Open Connection Function* que paralelamente se relaciona con la función *TCP Close Connection* para cerrar la conexión. Además, se maneja un ciclo de repetición *While* para que el bloque *TCP Read Function* reciba un conjunto de bytes de la conexión de red TCP y proporcione información en su salida.

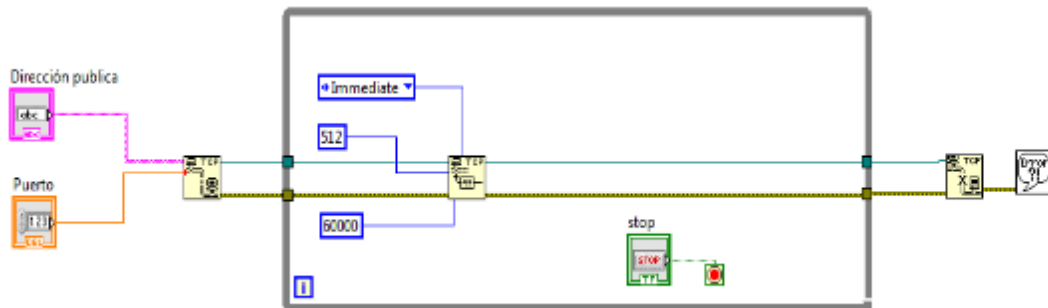


Figura 82-3: Bloque de programación para conexión TCP

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Paralelamente al bloque de programación se desarrolló una interfaz gráfica como se puede observar en la figura 98-3 con la finalidad de facilitar el ingreso de datos como la dirección TCP y el puerto requeridos para ejecutar la comunicación TCP y verificar el enlace con la Raspberry PI a través de la plataforma IoT y visualizar la información que ésta emita.

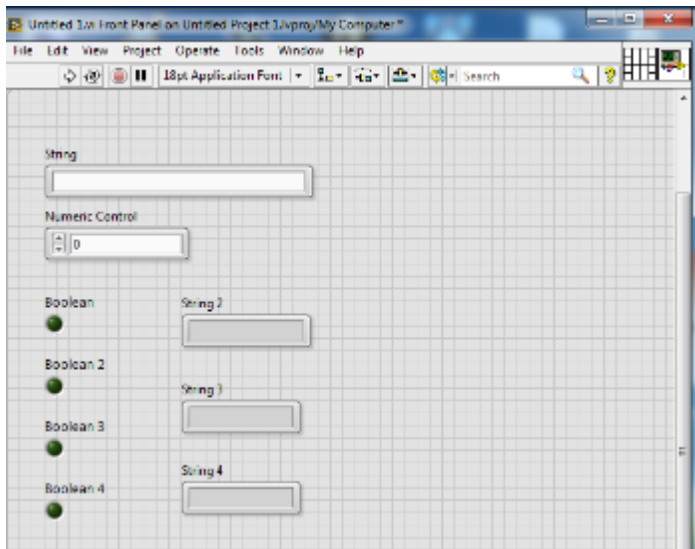


Figura 83-3: Interfaz gráfica para verificación de conexión de LabVIEW con el rehabilitador

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 99-3 representa el bloque de programación cuya función es la de identificar, procesar y distribuir la trama recibida con *Match Pattern Function* para ser direccionada a una estructura CASE que contiene acciones específicas para cada situación de información recibida.

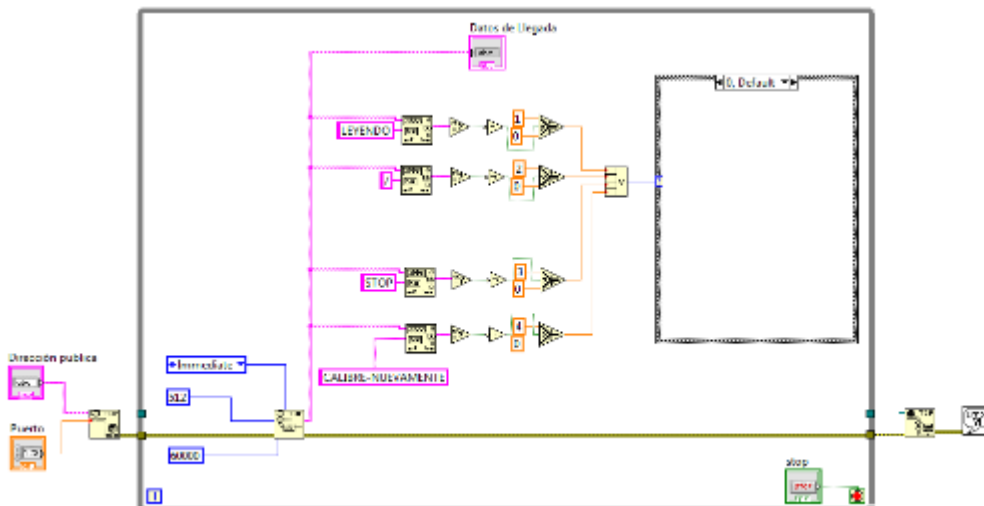


Figura 84-3: Boque de programación para el procesamiento de la cadena de texto recibida.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se puede observar en la figura 100-3 el contenido de la estructura CASE que conjuga los recursos para la verificación de la información recibida e interacción con los indicadores para el estado de la conexión.

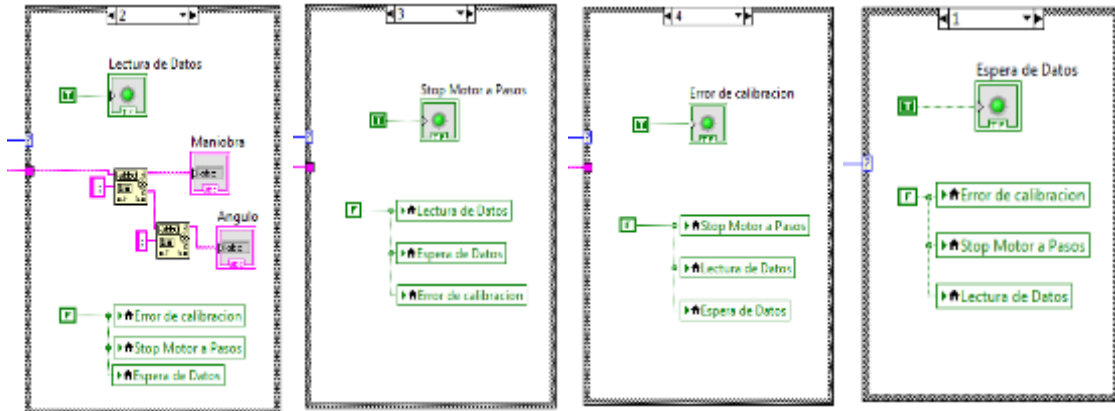


Figura 85-3: Sentencias de ejecución para cada espacio de la estructura CASE

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

a) *Configuración de parámetros para enlace de LabVIEW y Remote it*

Para ejecutar el enlace de las plataformas LabVIEW y Remote it se requiere la dirección TCP, la misma que se la puede obtener ingresando a la plataforma IoT, seleccionando el dispositivo creado en este caso la Raspberry Pi y el servicio TCP Raspberry, como se observa en la figura 101-3.

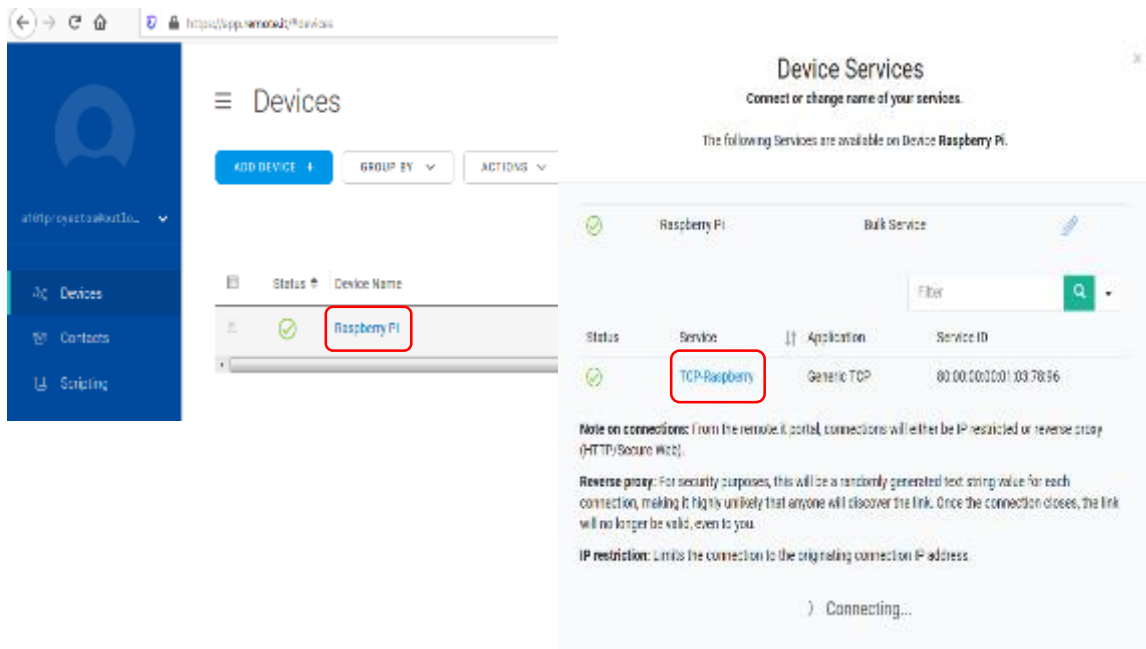


Figura 86-3: Ingreso a la información del servicio TCP en Remote it

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Al ingresar al servicio TCP en la plataforma Remote it se accedió a la información de la conexión, dichos datos son necesarios para la configuración de la dirección TCP y del puerto en la interfaz de Labview, se lo puede observar en la figura 102-3.

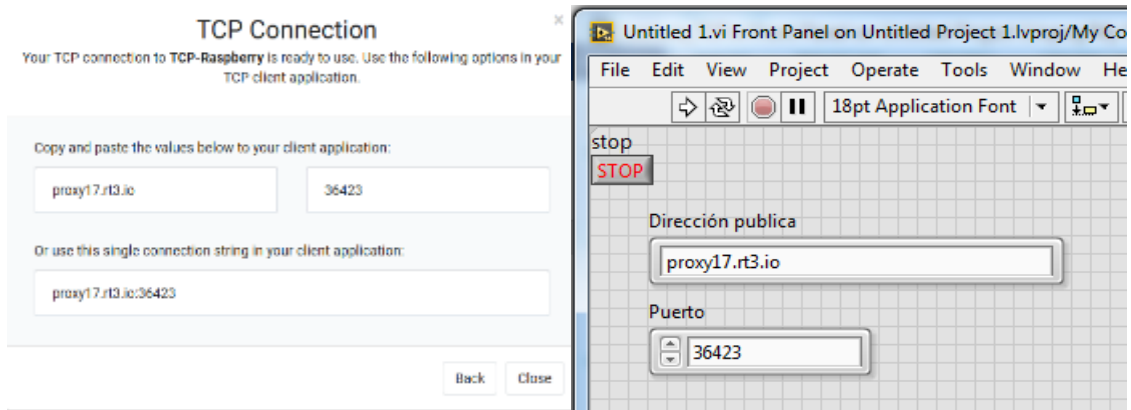


Figura 87-3: Obtención, carga de dirección TCP y puerto

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.8 Enlace de LabVIEW con SolidWorks

El propósito de este apartado es describir el trabajo realizado para la configuración desarrollada con el fin de enlazar un modelo de brazo virtual modelado en SolidWorks como se muestra en la figura 103-3 con LabVIEW. El fin con el que se desarrolló este modelo de brazo fue para que el monitoreo del funcionamiento del rehabilitador fuera animado en el sentido de dar seguimiento a los movimientos que efectúa el paciente en su proceso de rehabilitación.



Figura 88-3: Modelo 3D de brazo

virtual modelado en SolidWorks.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se posee un modelo virtual con 3 movimientos, por lo que se requirió de tres motores rotacionales, los mismos que están relacionados a los motores empleados en las articulaciones del rehabilitador para la ejecución de los movimientos requeridos.

El proceso de enlace entre las plataformas mencionadas se lo logró con la ejecución de dos etapas:

- Configuraciones en SolidWorks
- Configuraciones en LabVIEW

3.3.8.1 Configuraciones en SolidWorks

En SolidWorks se debe activar dentro de los complementos de la plataforma las opciones “SolidWorks Motion” y “SolidWorks Simulation”, de esta manera se habilita la opción para realizar un “Estudio de Movimiento” y “Análisis de movimiento”.

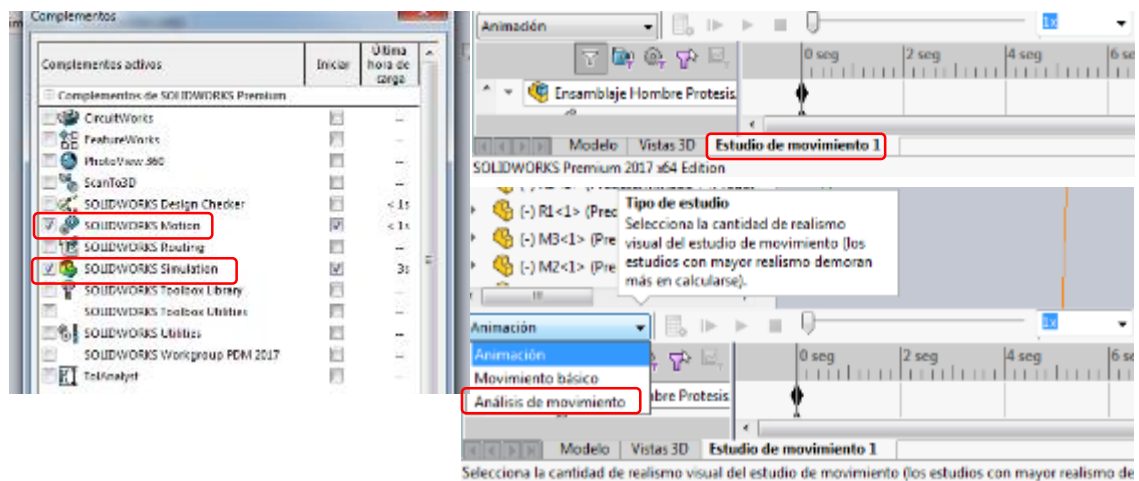


Figura 89-3: Habilitación complementos en SolidWorks.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Como recurso para la vinculación de SolidWorks con LabVIEW se crearon espacios de memoria denominados “Motor Rotatorio” relacionados directamente con las piezas del modelo 3D del brazo que ejecuta la animación del movimiento. Dentro de los parámetros de configuración de los motores rotatorios se seleccionó la cara donde se ubicará el motor rotatorio y a su vez se estableció su dirección de giro, como se muestra en la figura 105-3.

En la configuración de SolidWorks se debe seleccionar el recurso “Distancia” que habilita las características de ejecución de “Movimiento Absoluto” del módulo “Motion” de LabVIEW. Se ingresó valores aleatorios en SolidWorks para poder visualizar la simulación gracias al estudio de movimiento. Por ejemplo “90” grados y un estudio en tiempos de 0 a 10 segundos. Nota: Los

valores no repercutirán en la simulación del sistema, LabVIEW emitirá sus valores para la determinación del giro.

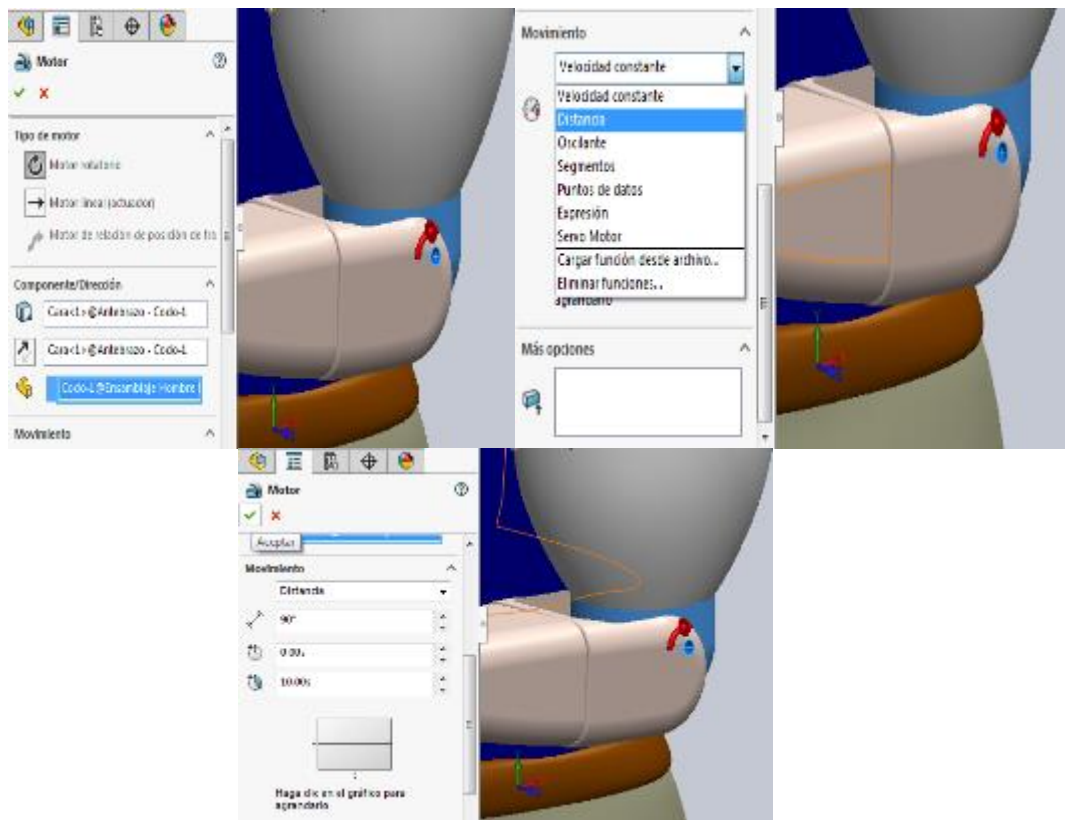


Figura 90-3: Asignación de un Motor Rotatorio

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.3.8.2 Configuraciones en LabVIEW

Para dar inicio a la vinculación de SolidWorks a LabVIEW se procedió a crear dentro del proyecto del prototipo un recurso nuevo del tipo “SolidWorks Assembly”, y se importó el modelo del brazo 3D, posterior se fijan parámetros relacionados con la forma de interpretación y actualización de información entre las plataformas, como se observa en la figura 106-3 en el espacio de configuración denominado ”Maximum step size” se fijó que el intercambio se realice en un tiempo de 0,01 a 0,001segundos, proceso requerido para que el software calcule las lecturas de simulación del ensamblaje entre 1×10^{-8} a 0,01segundos.

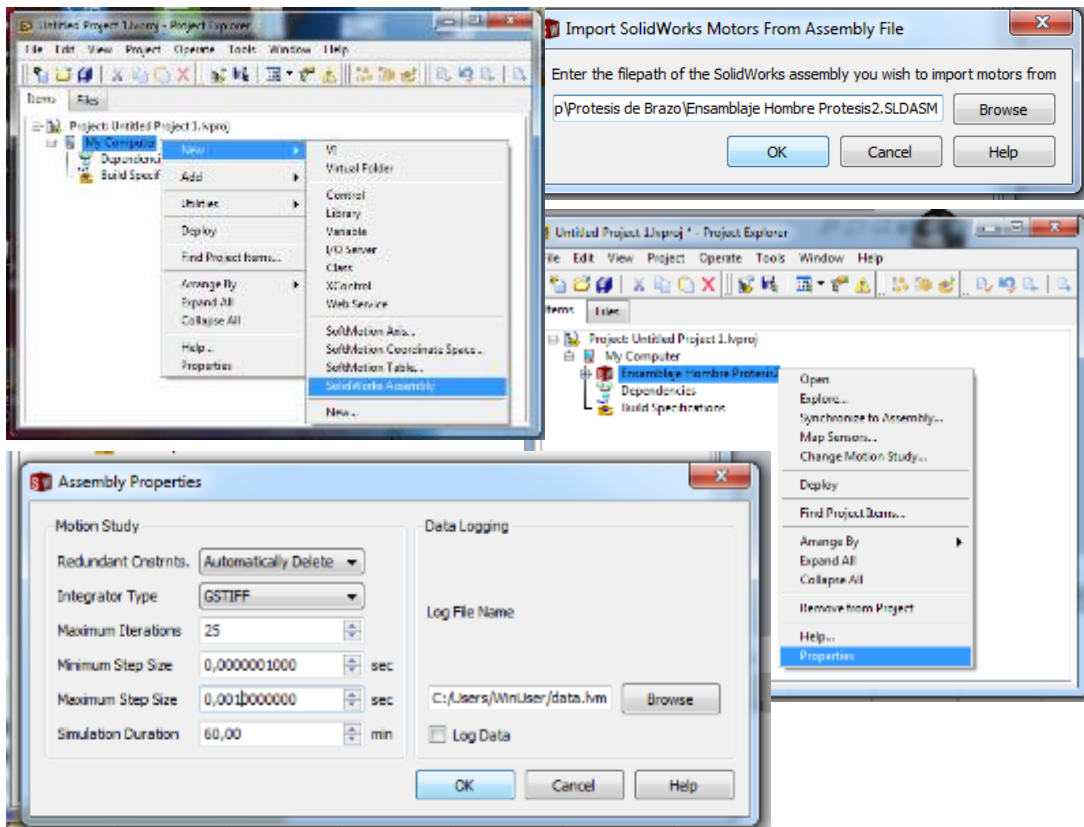


Figura 91-3: Configuración de recursos para el enlace de las plataformas

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Como se puede observar en la figura 107-3, se efectuaron ciertas acciones, entre ellas se describe que en la raíz del proyecto de LabVIEW se crearon tres nuevos elementos del tipo SoftMotion Axis, recursos sobre los cuales se montaron las variables de SolidWorks correspondientes a los motores rotatorios. En cada recurso Axis se conmutó en su hoja de propiedades la unidad de transición al modo activo (Opción “Enable Drive on Transition to Active Mode”), de esta manera se permite el tráfico de información bidireccional entre las dos plataformas.

Para dejar establecida la comunicación entre LabVIEW y SolidWorks se debió habilitar el Scan Engine, recurso que permite un acceso eficiente de un solo punto a conjuntos de canales de datos, como los canales de E / S. Usando un escaneo que almacena datos en un mapa de memoria global que actualiza todos los valores a una velocidad única, conocida como el período de escaneo. (LabVIEW, 2020)

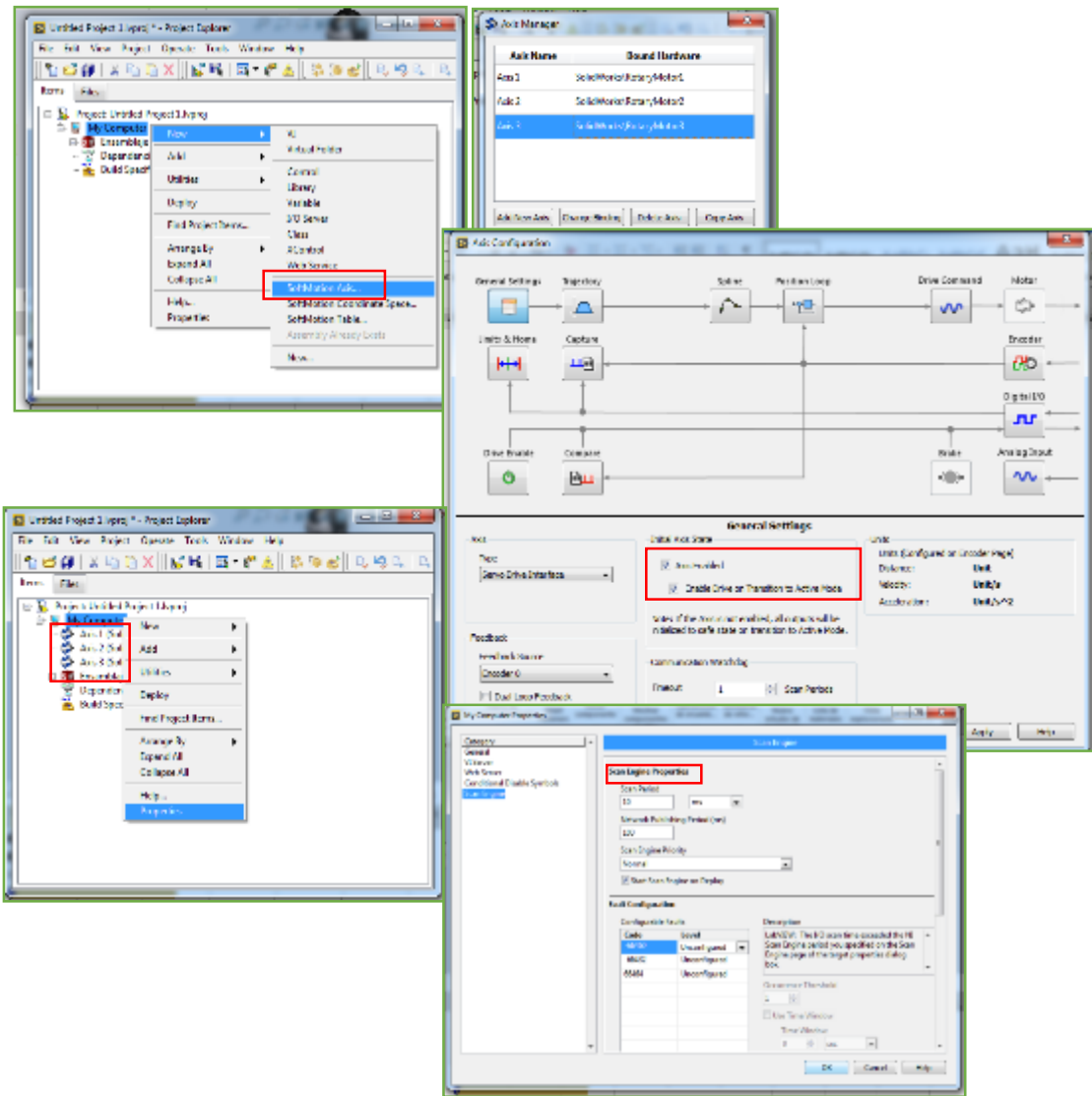


Figura 92-3: Configuración flujo de información de LabVIEW hacia el ensamble

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.4 Implementación del prototipo rehabilitador

3.4.1 Construcción del prototipo electromecánico

Validado el diseño y la resistencia del material empleado para cada pieza del rehabilitador se procede a su construcción, la mencionada validación se la expone en el apartado de análisis de resultados, la figura 108-3 permite observar la construcción de cada una de las piezas para su posterior ensamble.

Adicionalmente se observa que previo al ensamble del equipo se pintó las piezas para ofrecer un equipo estéticamente aceptable.



Figura 93-3: Piezas construidas

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Posterior a la construcción de las piezas se ensambló el sistema mecánico siguiendo la planimetría obtenida en el diseño, en la figura 109-3 se puede observar como el prototipo iba tomando forma.



Figura 94-3:. Ensamble mecánico de las piezas

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Ensamblado el sistema mecánico se insertaron los actuadores eléctricos que gobernarán los movimientos del rehabilitador de acuerdo al diseño como puede observarse en la figura 110-3.

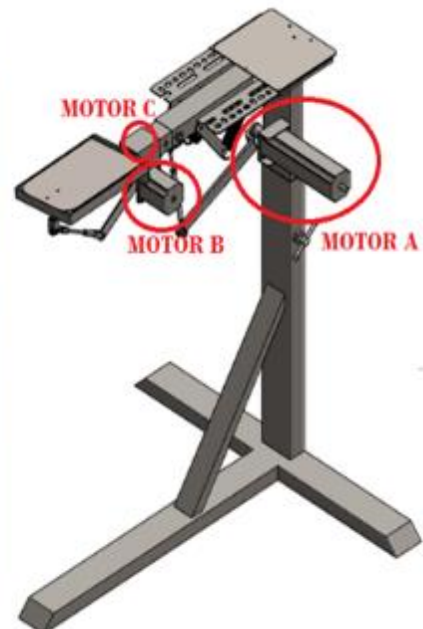


Figura 95-3: Acople de actuadores eléctricos

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.4.2 Implementación circuito eléctrico y electrónico

Una vez construido el equipo electromecánico se procedió a la implementación del circuito eléctrico y electrónico.

En la figura 111-3 se muestra el cableado de los controladores y motores de paso donde se conecta la alimentación del circuito y se polariza cada par de bobinas de los motores. También se realiza la conexión de los pines habilitadores del controlador (Enables).

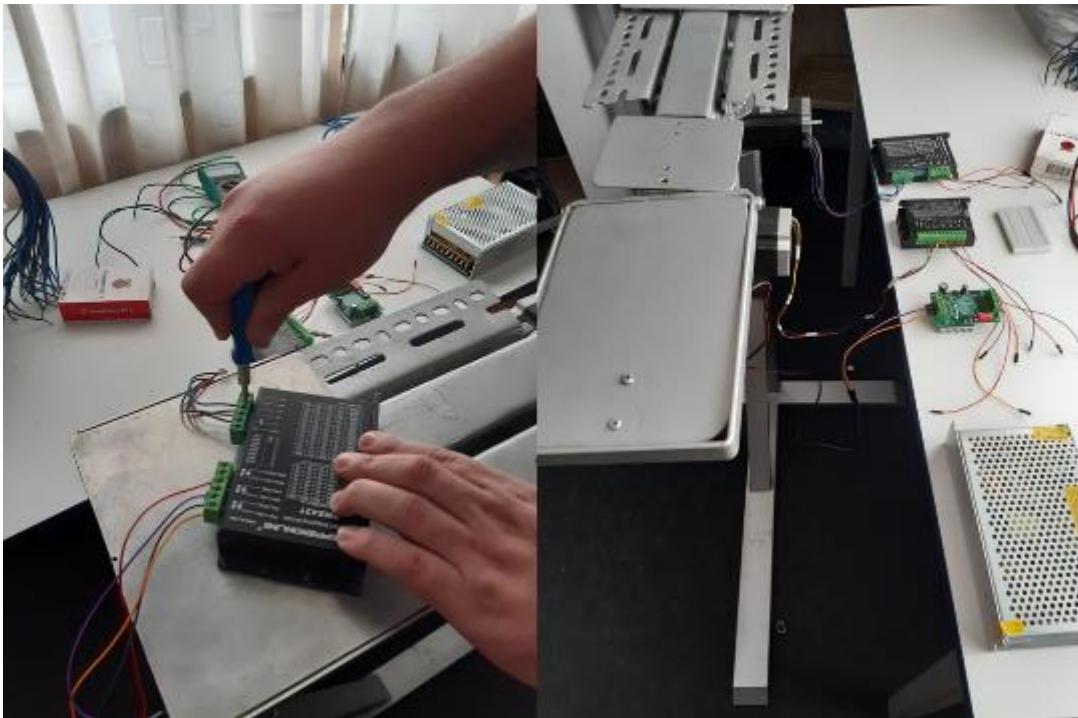


Figura 96-3: Instalación de drives y motores de paso

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 112-3 permite observar la integración de los dispositivos encargados de la gestión de las señales de control, en este caso partiendo de la Raspberry que mediante los algoritmos programados y cargados en la aplicación informática gestionará las señales hacia el microcontrolador de acuerdo a las instrucciones seteadas a través de la interfaz gráfica. La tarjeta Arduino por su lado actúa como pasarela adquiriendo la información de la Raspberry por medio de la comunicación serial para decodificarla y expresarla en términos de dirección y número de pasos que deberán recorrer los motores para la ejecución de sus rutinas en el prototipo rehabilitador electromecánico.

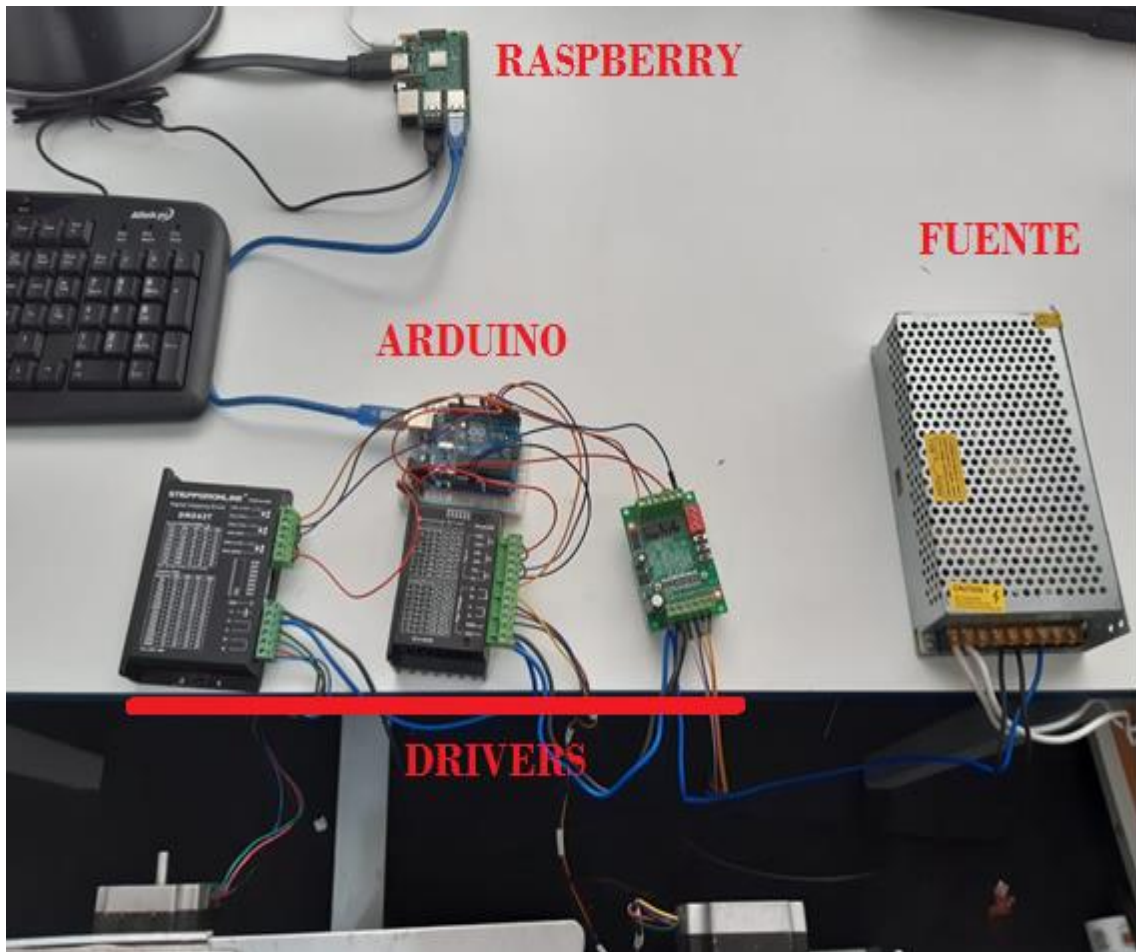


Figura 97-3: Instalación de dispositivos encargados de la gestión de las señales de control

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.5 Análisis y Resultados

En este apartado se plantea las acciones efectuadas en la construcción del prototipo rehabilitador, partiendo del estudio de los resultados obtenidos en el diseño de la sección del análisis estático y posterior a esto la construcción en sí del equipo electromecánico para mediante pruebas determinar la funcionalidad del prototipo.

3.5.1 Resultados del Análisis Estático

Como se pudo observar en la parte del diseño se estableció el material, las sujeciones y cargas a las que estarían expuestas ciertas piezas dentro de la estructura del prototipo rehabilitador desarrollado, en esta sección se analizan los resultados.

Tabla 51-3: Resultado Stress del resorte

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	0 N/mm ² (MPa) Nodo: 39	451.135 N/mm ² (MPa) Nodo: 31686

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

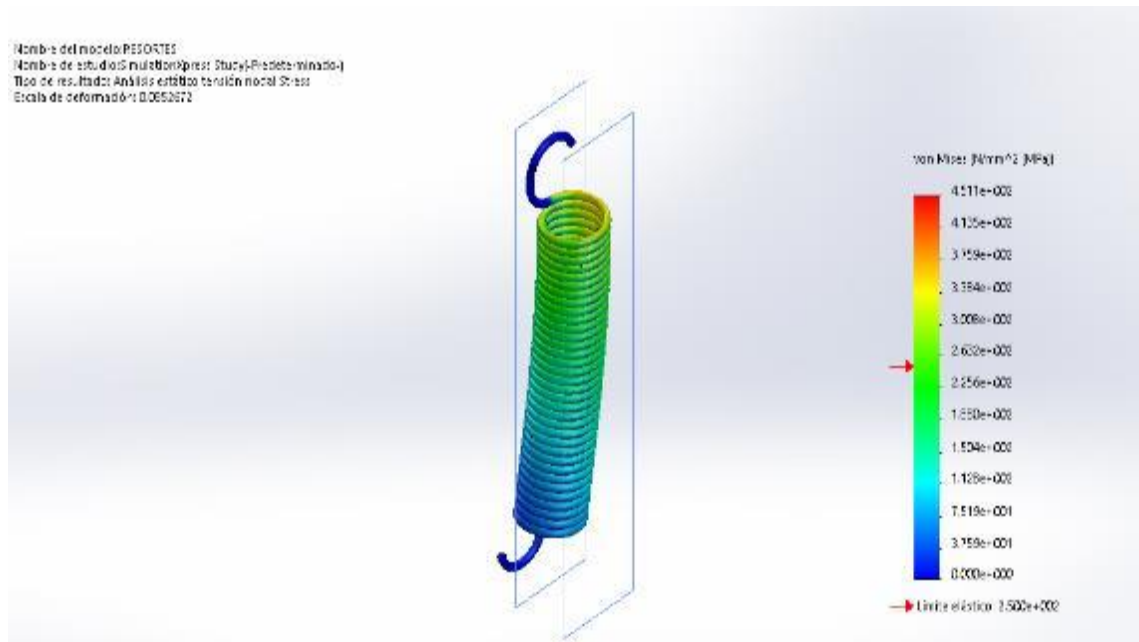


Figura 98-3: Stress del resorte

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 52-3: Resultado del desplazamiento por esfuerzo del resorte

NOMBRE	TIPO	MÍN.	MÁX.
Resultado del desplazamiento por esfuerzo	URES: Desplazamientos resultantes	0 mm Nodo: 29	153.211 mm Nodo: 7184

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

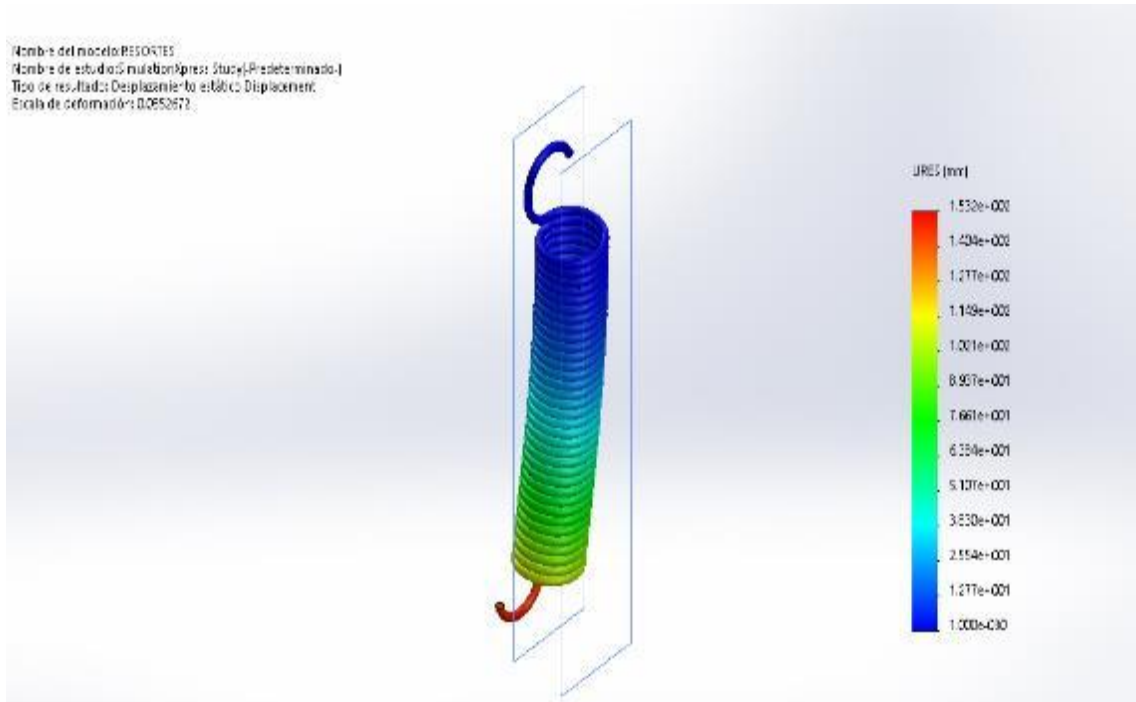


Figura 99-3: Desplazamiento por esfuerzo del resorte

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La construcción del resorte se lo realizó en un material estándar como es el acero ASTM A 36 a la cual se sometió una fuerza sobredimensionada de 250 N, obteniendo un desplazamiento de $1e+016$ en el Nodo: 39 por lo que se ubica un sujetador en U en la parte inferior, considerando que el diseño contiene dos resortes para reducir el esfuerzo aplicado.

Se puede cambiar de resorte en el caso que se desee aplicar mayor esfuerzo debido a que su diseño da la alternativa de intercambiar los elementos con facilidad.

Tabla 53-3: Resultado Stress de la base principal

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	1.02404e-010 N/mm ² (MPa) Nodo: 9758	1.58068 N/mm ² (MPa) Nodo: 15749

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

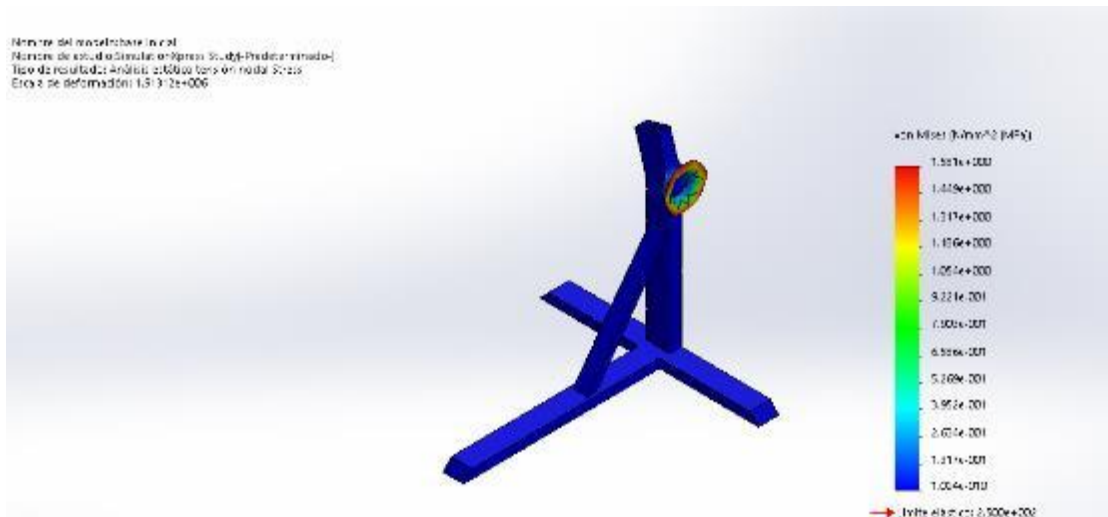


Figura 100-3: Stress de la base principal

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 54-3: Resultado del desplazamiento por esfuerzo de la base principal

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamiento del elemento sujeto a esfuerzos	URES: Desplazamientos resultantes	0 mm Nodo: 448	4.24668e-005 mm Nodo: 12

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

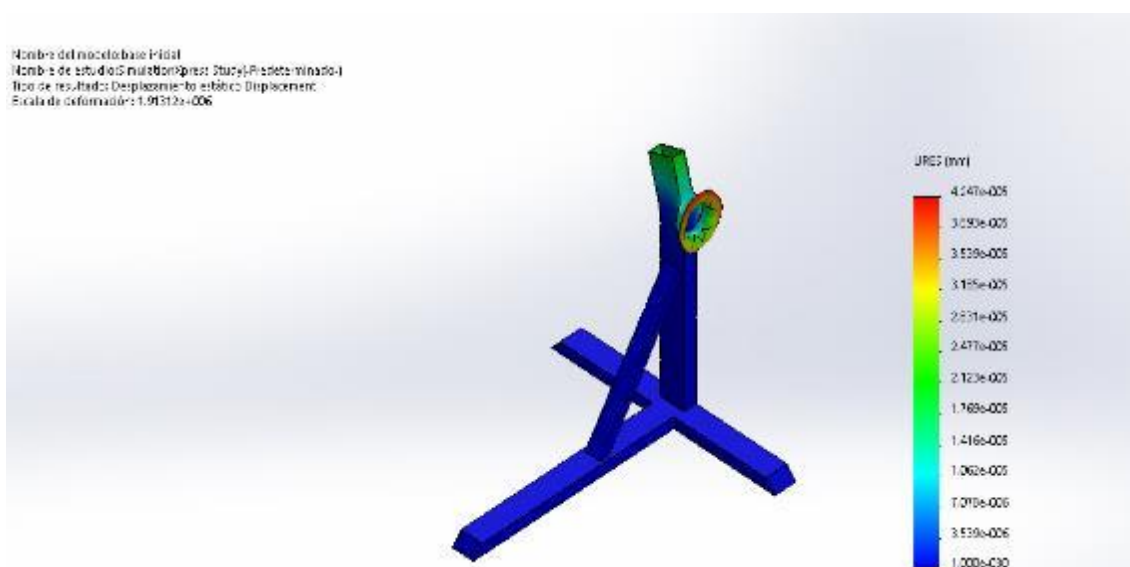


Figura 101-3: Desplazamiento por esfuerzo de la base principal

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se empleó para la construcción de la base principal acero estructural ASTM A36 Acero con las características puestas en el diseño debido a que se utiliza perfiles y de esta manera comprobó su resistencia al esfuerzo que se va a aplicar en el prototipo rehabilitador. La deformación que sufre el elemento aplicando una fuerza de 250 N es de 4.24668e-005 mm en el Nodo: 12 que es un desplazamiento despreciable por lo que se procede a la construcción de la estructura.

Al utilizar el tubo estructural se puede ver que estéticamente y funcionalmente son las adecuadas.

Tabla 55-3: Resultado Stress de la base del motor del antebrazo

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	0.00026381 N/mm ² (MPa) Nodo: 15630	24.5293 N/mm ² (MPa) Nodo: 108

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

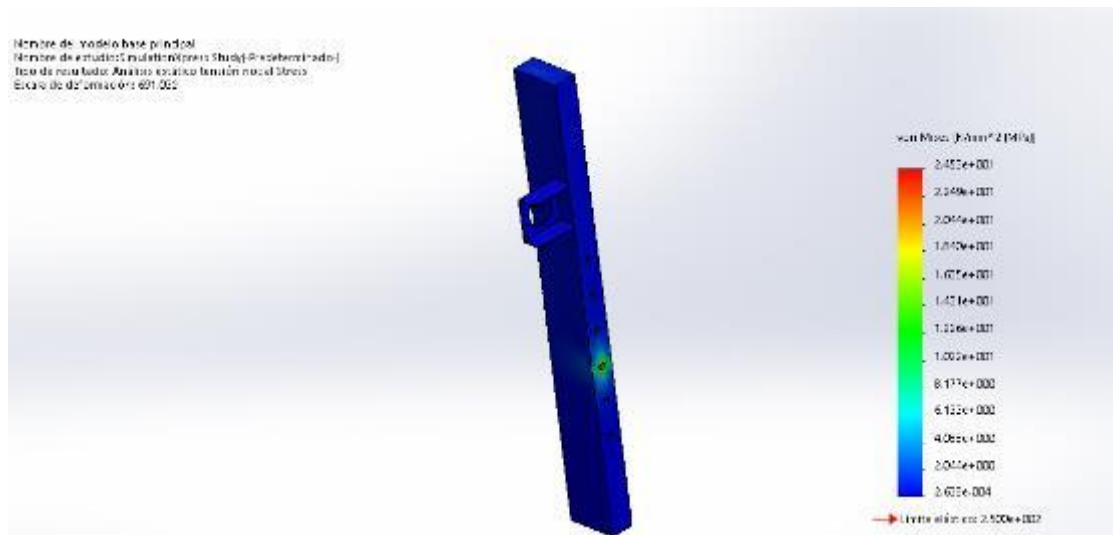


Figura 102-3: Stress de la base del motor del antebrazo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 56-3: Resultado del desplazamiento por esfuerzo de la base del motor del antebrazo

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamiento con las cargas que soporta	URES: Desplazamientos resultantes	0 mm Nodo: 103	0.0945556 mm Nodo: 988

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

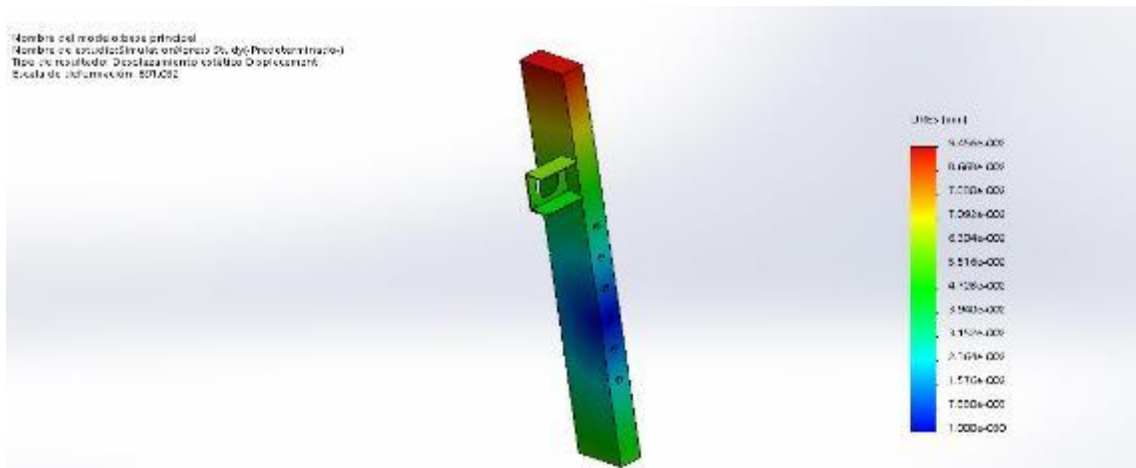


Figura 103-3: Desplazamiento por esfuerzo de la base del motor del antebrazo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Al aplicar una fuerza de 200 N en la parte superior y 10 N en cada agujero que va a soportar el motor del movimiento del antebrazo se tiene una deformación de 0.0945556 mm en el Nodo: 988 del perfil utilizado y elaborado en acero Estructural ASTM A 36 por lo que se considera despreciable y se procede a realizar esta pieza con las uniones respectivas.

El tubo estructural permite tener facilidad de adquisición y soldadura en el caso de la moldura que aloja al motor.

Tabla 57-3: Resultado Stress de la base de alojamiento del antebrazo

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	1.62071e-011 N/mm ² (MPa) Nodo: 5484	3.64395 N/mm ² (MPa) Nodo: 15801

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

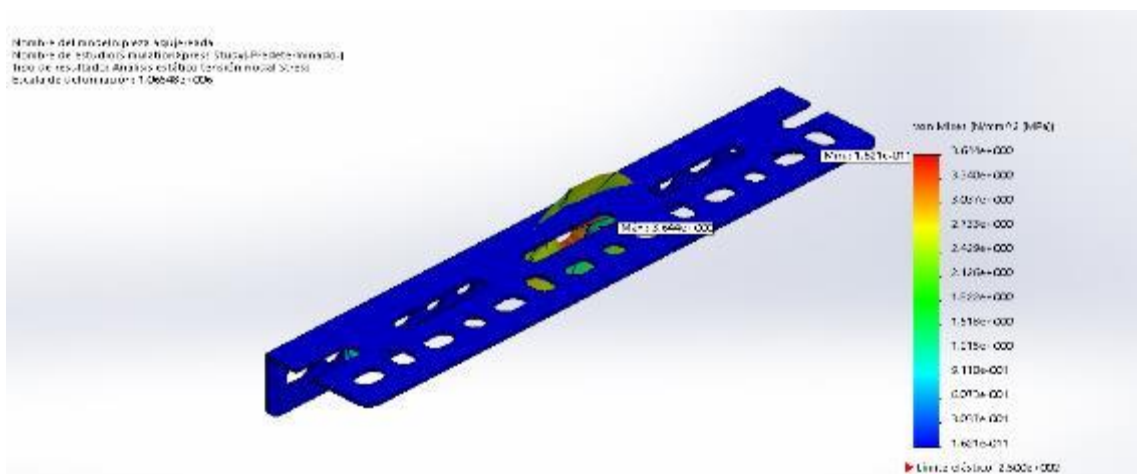


Figura 104-3: Desplazamiento por esfuerzo de la base principal

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 58-3: Resultado del desplazamiento por esfuerzo de alojamiento del antebrazo

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamiento con cargas y sujeciones aplicadas	URES: Desplazamientos resultantes	0 mm Nodo: 4	3.06386e-005 mm Nodo: 15813

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

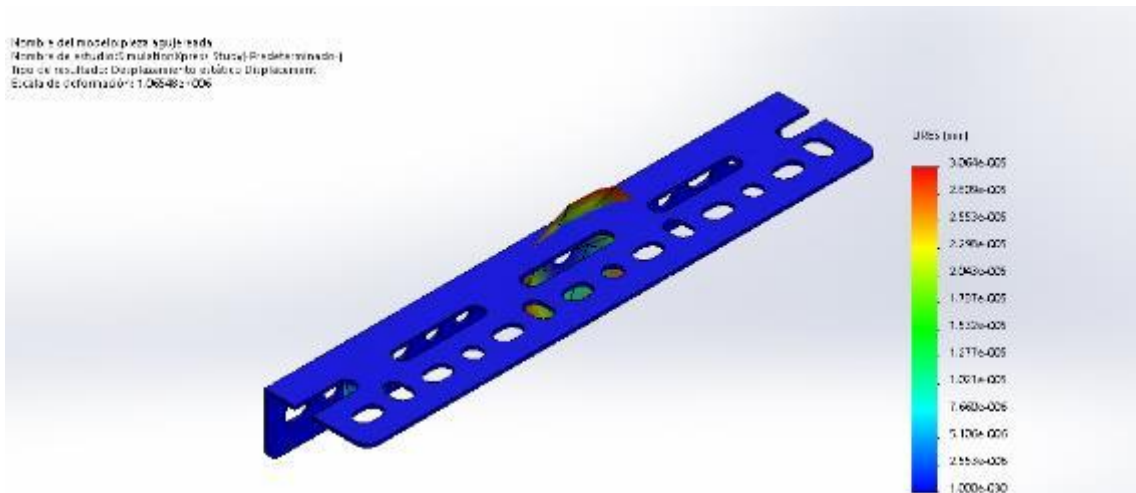


Figura 105-3: Desplazamiento por esfuerzo de la base de alojamiento del antebrazo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Los agujeros que tiene el perfil construido en acero ASTM A 36 permite acoplar los elementos según la disposición del prototipo rehabilitador, de esta manera se procedió a ubicar las fuerzas a la que va estar sometida como es de 150 N en la movilidad del antebrazo y 50 N en la palanca que viene del motor que va a dar la movilidad al antebrazo. Con estas fuerzas se obtiene una deformación de 3.06386e-005 mm en el Nodo: 15813 que es imperceptible y se procede a seleccionar como parte del rehabilitador.

Tabla 59-3: Resultado Stress del soporte del cilindro y resorte

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	3.14864e-006 N/mm ² (MPa) Nodo: 3122	2.11903 N/mm ² (MPa) Nodo: 36

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

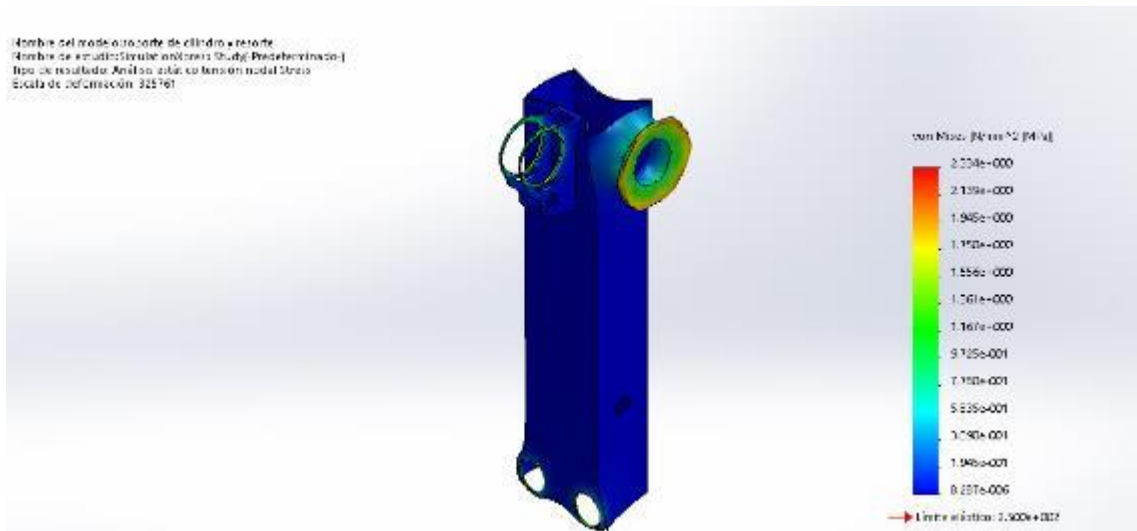


Figura 106-3: Stress del soporte del cilindro y resorte

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 60-3: Resultado del desplazamiento por esfuerzo del soporte del cilindro y resorte

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamientos con cargas y sujeciones aplicadas	URES: Desplazamientos resultantes	0 mm Nodo: 1	8.70383e-005 mm Nodo: 43

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

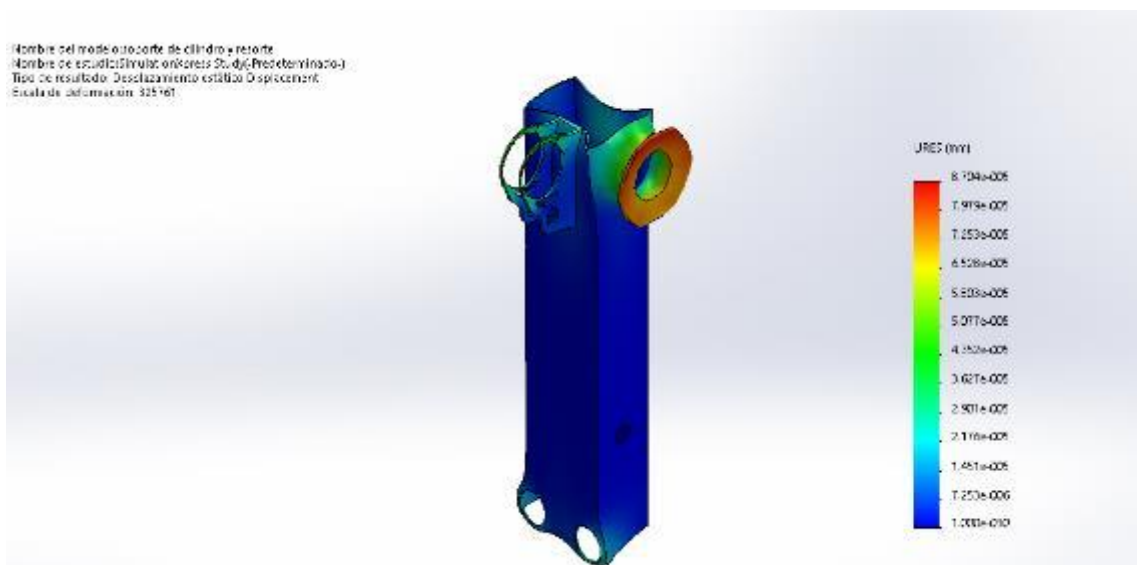


Figura 107-3: Desplazamiento por esfuerzo del soporte del cilindro y resorte

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se usó acero estructural ASTM A36 con sus características en el diseño y se comprobó su resistencia al esfuerzo aplicado en el prototipo rehabilitador. El desplazamiento muestra una deformación de 8.69627×10^{-5} mm en el Nodo: 43 que es despreciable a los esfuerzos sometidos de los resortes de 20 N, de movilidad de 40 N y sujeción al soporte de la mano de 100 N.

El diseño del soporte para giro de la muñeca soporta los esfuerzos y la movilidad requerida por lo que se procede a su construcción.

Tabla 61-3: Resultado Stress del soporte para el giro de la muñeca

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	1.46128e-006 N/mm ² (MPa) Nodo: 6192	1.99694 N/mm ² (MPa) Nodo: 62

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

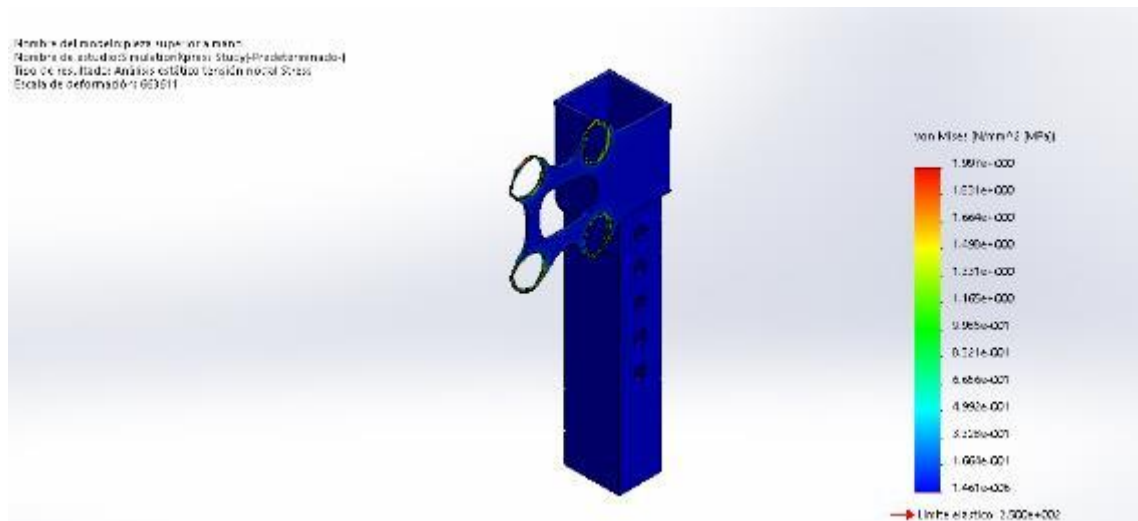


Figura 108-3: Stress del soporte del soporte para el giro de la muñeca

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 62-3: Resultado del desplazamiento por esfuerzo del soporte para el giro de la muñeca

NOMBRE	TIPO	MÍN.	MÁX.
Deformación con las cargas y sujeciones aplicadas	URES: Desplazamientos resultantes	0 mm Nodo: 101	4.03173e-005 mm Nodo: 16101

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

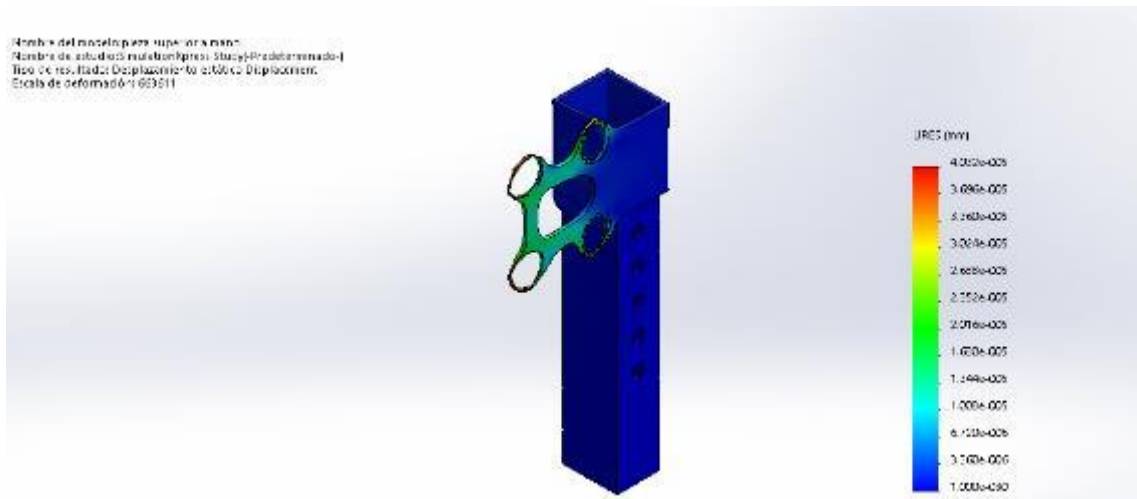


Figura 109-3: Desplazamiento por esfuerzo del soporte para el giro de la muñeca

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se utiliza el acero estructural ASTM A36 Acero con las características puestas en el diseño para comprobar su resistencia al esfuerzo que se va a aplicar en el rehabilitador. Se consideró una fuerza en los agujeros donde está colocado el motor debido a que se tendrá un esfuerzo al mover la mano, generando una deformación de $4.03173e-005$ mm en el Nodo: 16101 que es despreciable para una fuerza de 40 N en cada agujero de un total de 160 N, por lo que se procede a la construcción de esta pieza.

La sujeción de la pieza me permite alargar o disminuir su distancia permitiendo un rango de rehabilitación.

Tabla 63-3: Resultado Stress del elemento de movilidad para la mano

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	$1.5857e-011$ N/mm ² (MPa) Nodo: 3816	3.01044 N/mm ² (MPa) Nodo: 9847

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

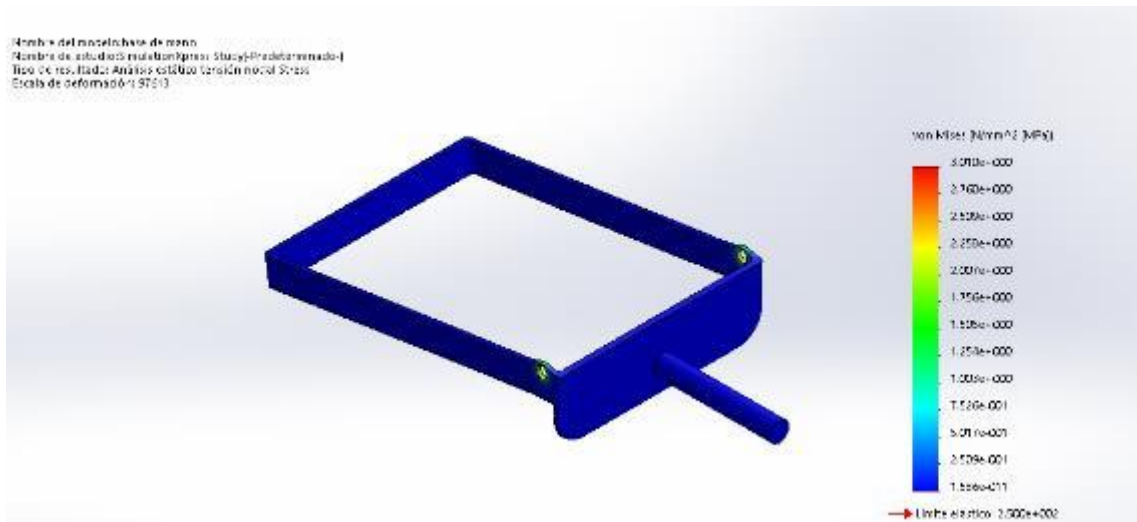


Figura 110-3: Stress del soporte del elemento de movilidad para la mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 64-3: Resultado desplazamiento por esfuerzo del elemento de movilidad para la mano

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamiento con las cargas y sujeciones	URES: Desplazamientos resultantes	0 mm Nodo: 29	0.000324688 mm Nodo: 14963

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

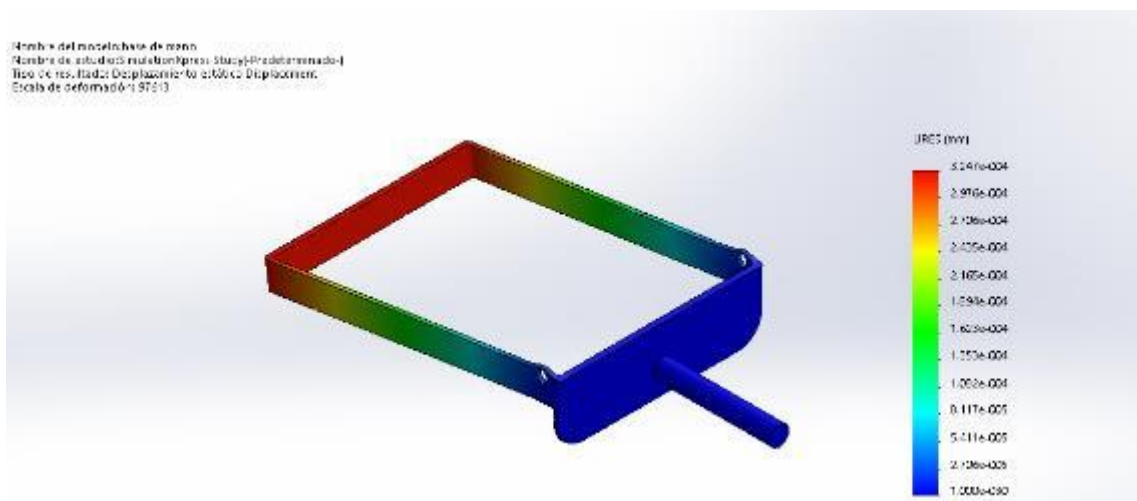


Figura 111-3: Desplazamiento por esfuerzo del elemento de movilidad para la mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Al aplicar un esfuerzo de 50 N en un material de acero ASTM A36, en cada agujero suponiendo que el esfuerzo que se aplica en el prototipo rehabilitador sea de 100N se tiene una deformación

de 0.000324688 mm en el Nodo: 14963 que es despreciable por lo que se recomienda realizar este tipo de diseño considerando que se mantenga a un esfuerzo constante.

En este diseño se considera que ayudará por medio de una palanca a realizar el movimiento de la mano en la etapa de rehabilitación.

Tabla 65-3: Resultado Stress del elemento de apoyo para rehabilitación de mano

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	0.0129316 N/mm ² (MPa) Nodo: 17047	55.3289 N/mm ² (MPa) Nodo: 17008

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

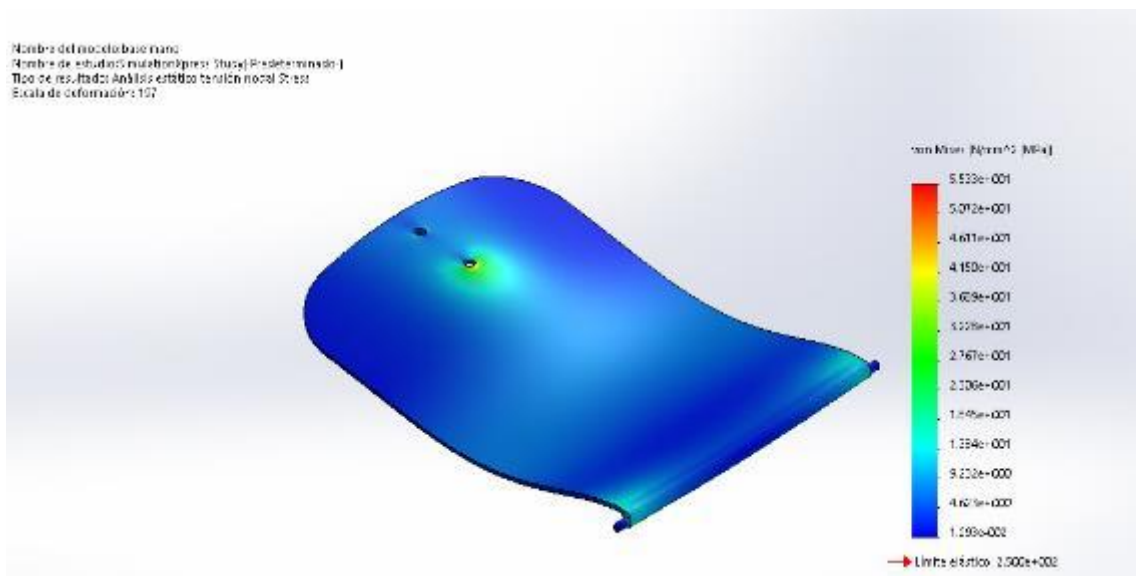


Figura 112-3: Stress del elemento de apoyo para rehabilitación de mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 66-3: Resultado desplazamiento por esfuerzo del apoyo para rehabilitación de mano

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamiento con cargas y sujeciones	URES: Desplazamientos resultantes	0 mm Nodo: 1	0.109137 mm Nodo: 16669

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

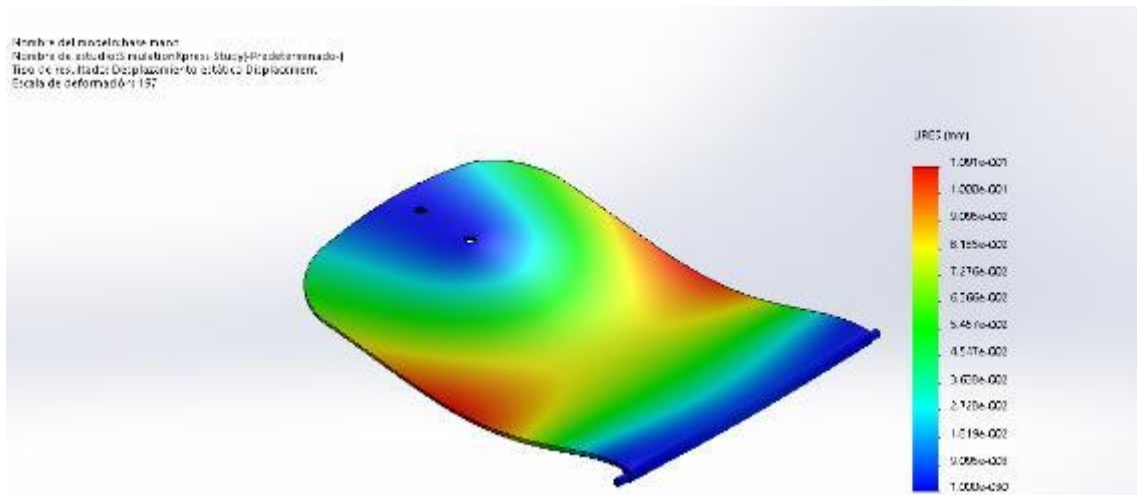


Figura 113-3: Desplazamiento por esfuerzo del elemento apoyo para rehabilitación de mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Al aplicar una fuerza de 100N en un material de acero ASTM A36 en la placa asumiendo que es el esfuerzo en la rehabilitación de la mano, se obtiene un desplazamiento de 0.109137 mm en el Nodo: 16669 que es despreciable, por tal motivo se procede a la construcción del modelo para el acople y movilidad de la mano.

Las dimensiones de este modelo permiten que la mano entre complete en la etapa de rehabilitación.

Tabla 67-3: Resultado Stress del elemento del soporte de movimiento de mano

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	2.2355e-009 N/mm ² (MPa) Nodo: 11725	1.65216 N/mm ² (MPa) Nodo: 12483

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

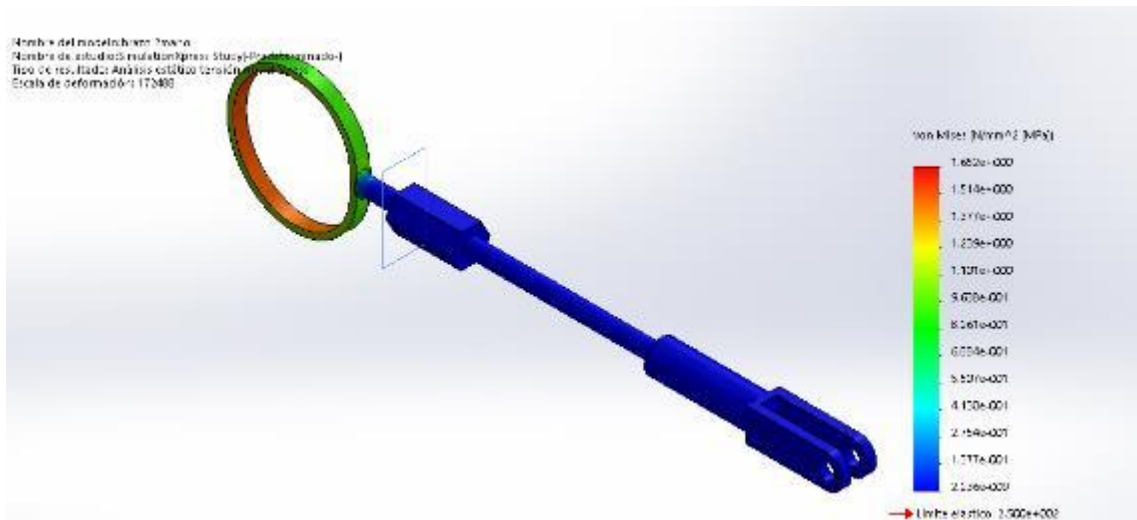


Figura 114-3: Stress del soporte de movimiento de mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 68-3: Resultado del desplazamiento por esfuerzo del soporte de movimiento de mano

NOMBRE	TIPO	MÍN.	MÁX.
Displacement	URES: Desplazamientos resultantes	0 mm Nodo: 254	0.000100316 mm Nodo: 12856

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

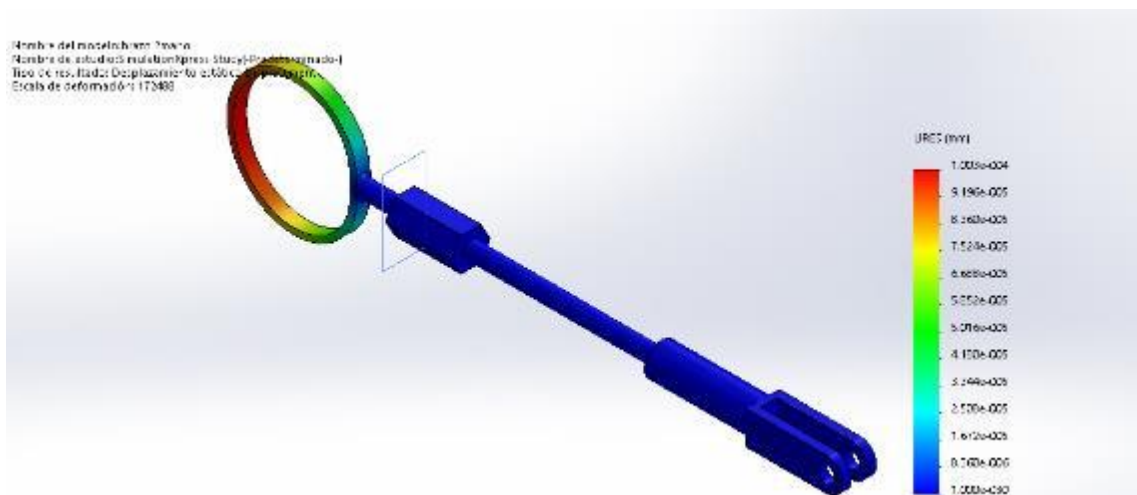


Figura 115-3: Desplazamiento por esfuerzo del soporte de movimiento de mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

El brazo para rehabilitación del movimiento de la mano se le aplica una fuerza de 100 N en un material de Acero ASTM A36 asumiendo que se puede poner algún peso adicional a una mano y

con lo que la deformación es de 0.000100316 mm en el Nodo: 12856 que es mínimo garantizando la construcción del soporte de movimiento de mano.

Se considera a esta pieza en el movimiento de la mano como crítico, pero con los ensayos por medio del software permite verificar que es el adecuado y se evita la prueba y error que anteriormente se realizaba.

Tabla 69-3: Resultado Stress del elemento del brazo de movilidad del antebrazo

NOMBRE	TIPO	MÍN.	MÁX.
Stress	VON: Tensión de von Mises	4.80413e-010 N/mm ² (MPa) Nodo: 114	2.22677 N/mm ² (MPa) Nodo: 256

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

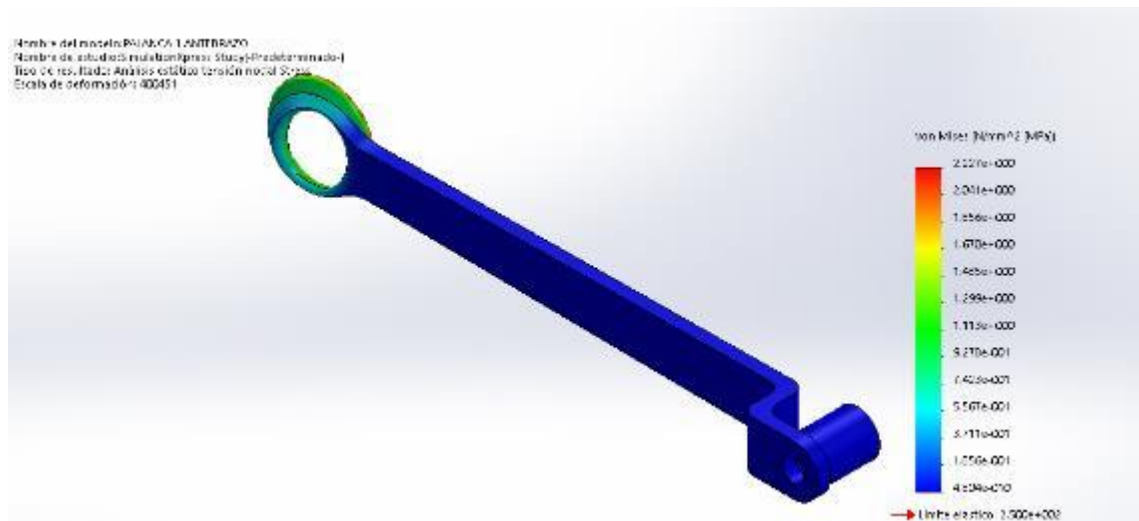


Figura 116-3: Stress del brazo de movilidad del antebrazo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Tabla 70-3: Resultado desplazamiento por esfuerzo del brazo de movilidad del antebrazo

NOMBRE	TIPO	MÍN.	MÁX.
Desplazamiento con cargas y sujeciones	URES: Desplazamientos resultantes	0 mm Nodo: 43	6.99367e-005 mm Nodo: 12640

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

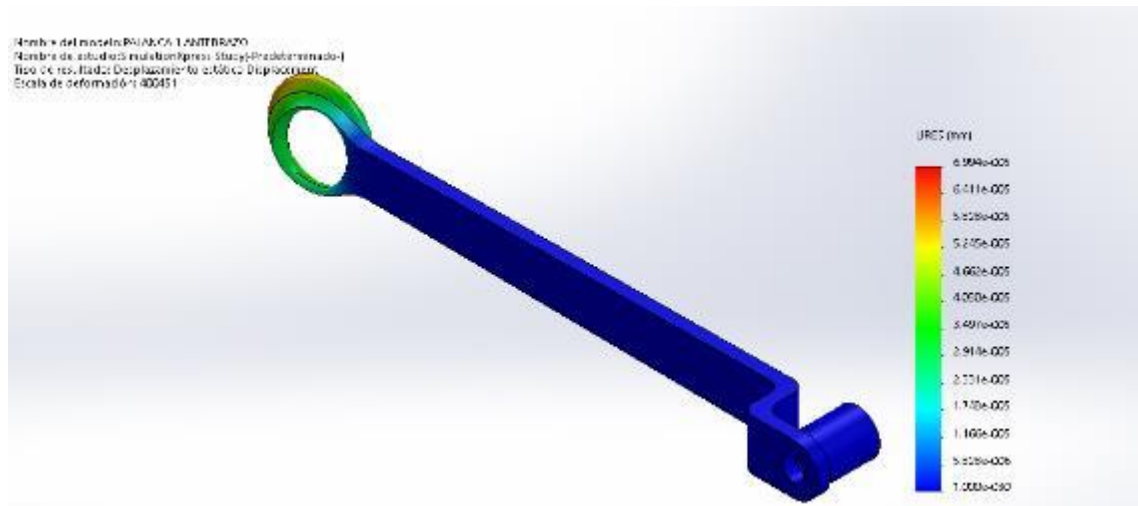


Figura 117-3: Desplazamiento por esfuerzo de la movilidad del antebrazo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Para la movilidad del ante brazo se utiliz3 acero ASTM A36 colocando sus caracter3sticas en el diseo y comprobando su resistencia al esfuerzo que se va a aplicar en el prototipo rehabilitador, considerando una fuerza de 200 N la deformaci3n es de $6.99367e-005$ mm en el Nodo: 12640 por lo que es un desplazamiento despreciable y se procede al desarrollo de esta pieza.

Esta palanca es la principal en cuanto a sujeci3n y soporte del peso que va en la rehabilitaci3n del antebrazo. De esta manera se comprob3 que el material y diseo de todas las piezas sometidas a esfuerzos y cargas cumplen normas de calidad.

3.5.2 An3lisis de resultados estructurales del prototipo

Con la construcci3n del rehabilitador se logr3 obtener como resultado un equipo accesible a varias personas, pues se dot3 al modelo flexibilidad en el ajuste de su tamao en zonas espec3ficas, pues se consider3 que las medidas antropom3tricas de las personas no son las mismas.

3.5.2.1 Regulaci3n de mec3nica para la altura del prototipo

La figura 133-3 permite observar la calibraci3n de la base principal del equipo, donde el juego de tubos conc3ntricos con la asistencia de un tornillo que act3a como “prisionero”, permiten dotar al equipo de una calibraci3n en su altura de 15 cent3metros.



Figura 118-3: Regulación de la base principal para ajuste de la altura del equipo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Con la calibración de la altura del equipo se facilita la adaptación del mismo al medio del paciente, es decir a que el mismo pueda complementar su set de rehabilitación con cualquier tipo de silla que pueda disponer en su casa.

3.5.2.2 Regulación de mecánica para el apoyo del brazo

Otro de los segmentos del equipo que se obtuvo regulable es el apoyo del brazo, pues de paciente a paciente puede diferir la longitud del brazo. En la figura 134-3 se puede observar que son 7 centímetros de regulación que se dispone en este segmento.

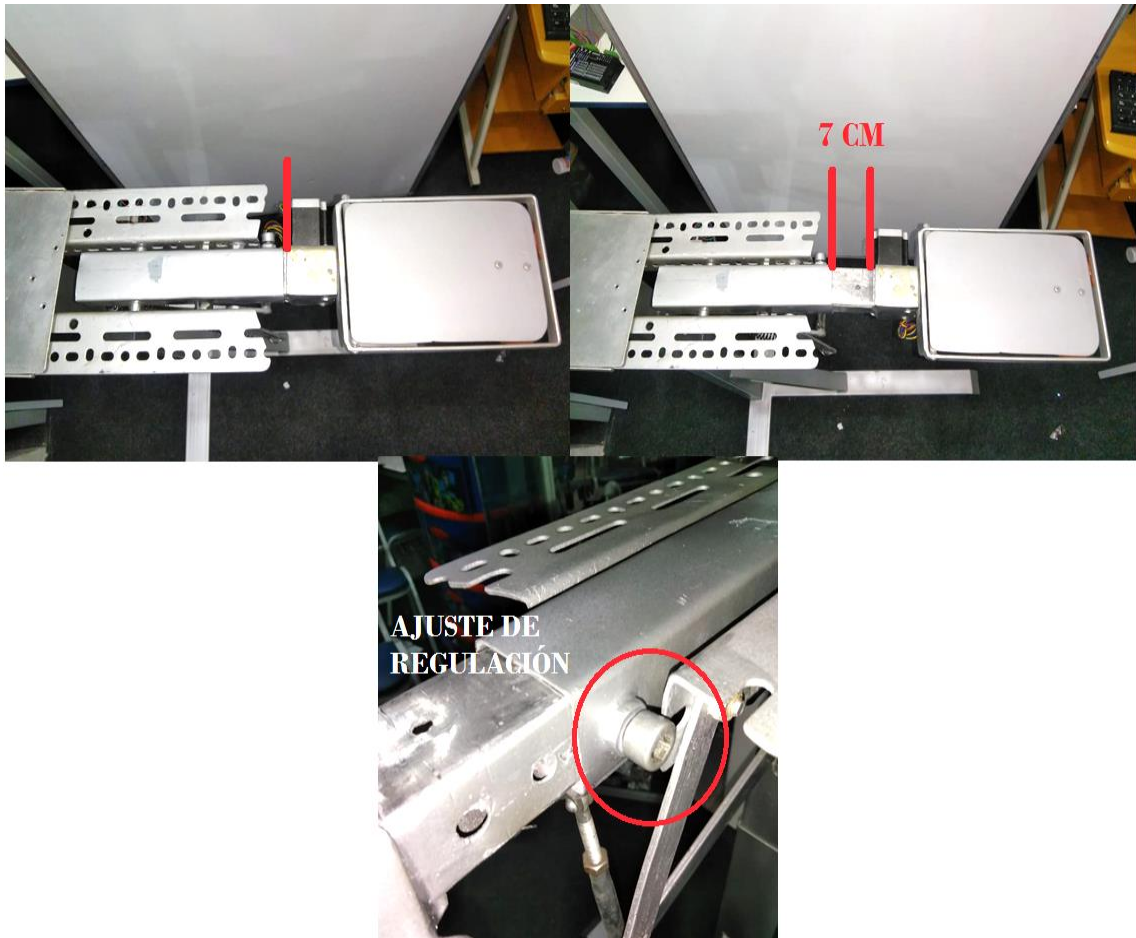


Figura 119-3: Regulación del apoyo del brazo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La tabla 71-3 representa los rangos de cobertura de dimensiones corporales determinando de esta manera la población que podrá acceder a las bondades del equipo rehabilitador.

Tabla 71-3: Resultados de coberturas corporales del equipo

SECCIÓN DEL MIEMBRO SUPERIOR	RANGO DE AJUSTE EN (CM)	DIMENSIÓN MÍNIMA	DIMENSIÓN MÁXIMA	OBSERVACIONES
ALTURA PACIENTE SENTADO	15	63	78	Asignación de postura del paciente y silla amplía el rango
ANTEBRAZO	7	20	27	Asignación de postura del paciente amplía el rango
MANO	--	--	--	Aplica a cualquier tamaño de mano

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.5.3 Pruebas de funcionalidad

Una vez implementado todo el prototipo integrando la parte electromecánica y la parte electrónica se procedió a las pruebas de funcionalidad del equipo.

3.5.3.1 Puesta en marcha de la interfaz gráfica

La figura 135-3 representa la totalidad del prototipo en su aspecto físico. Para darle funcionalidad al mismo, se debe proceder a arrancar inicialmente el ejecutable de la aplicación informática desarrollada en Python, el archivo ejecutable cargado en la Raspberry Pi3 es de extensión .py.



Figura 120-3: Sistema Electromecánico

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

La figura 136-3 representa el resultado obtenido de la programación de recursos para la interfaz gráfica elaborada en Python.



Figura 121-3: Resultado – Interfaz gráfica

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.5.3.2 Ejecución de conectividad de LabVIEW con Remote it

Para llevar a cabo la prueba de conectividad entre LabVIEW y Remote it se ejecutó la interfaz gráfica para el usuario desarrollada a partir de programación en Python, obteniendo la interfaz mostrada en la figura 137-3 donde para iniciar la comunicación de la Raspberry con la plataforma IoT se ingresa inicialmente la contraseña y se presiona el botón conectar, de esta manera si no genera error y el estado de la etiqueta de verificación conmuta a CONECTADO, el dispositivo se encuentra enlazado a Remote it y por ende la plataforma IoT está enviando información a través de su puerto y dirección TCP especificada a LabVIEW.



Figura 122-3: Resultado - Conectividad de la Raspberry Pi con Remote it

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Se ejecuta la aplicación desarrollado en LabVIEW y se podrá verificar la recepción de información desde la Raspberry Pi a través de la plataforma IoT. La figura 138-3 muestra paralelamente el terminal de la Raspberry en la que se puede verificar la información de salida hacia la plataforma Remote It y el resultado obtenido en la interfaz de LabVIEW que verifica la cadena LEYENDO emitida por el dispositivo del rehabilitador.

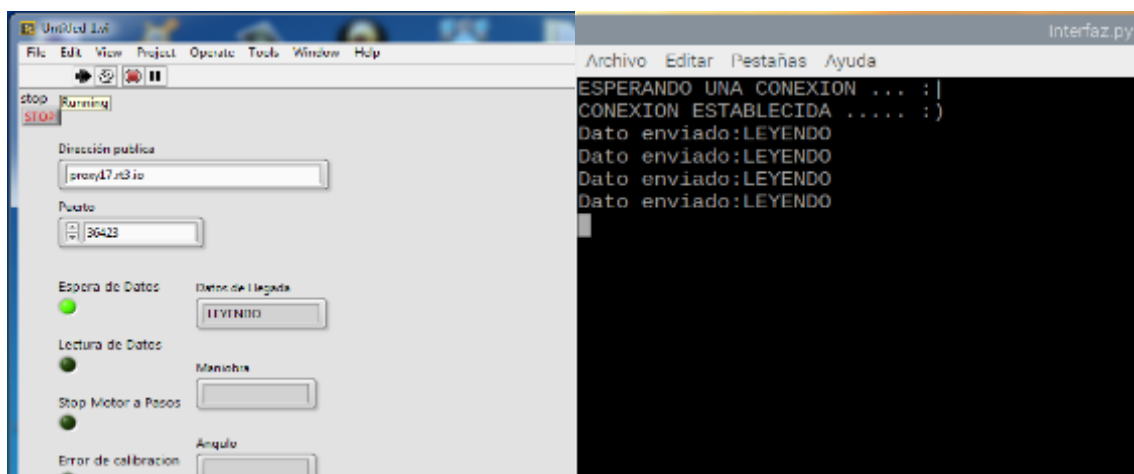


Figura 123-3: Enlace LabVIE y el dispositivo Raspberry Pi

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.5.3.3 Prueba funcionamiento del prototipo en Modo de Calibración

En la interfaz gráfica local, para calibración de los motores del prototipo rehabilitador se desbloquea el modo de calibración y se ingresa el ángulo para determinar avance o retroceso del motor. El proceso de calibración es importante previo a la maniobra, como se puede observar en la figura 139-3.



Figura 124-3: Resultado – Ejecución de Calibración del equipo.

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.5.3.4 Prueba funcionamiento del prototipo en Modo de Maniobra

Posterior a la calibración se realiza un test del modo maniobra ingresando los datos: ángulo de desplazamiento y número de repeticiones. Al ejecutar la acción con el botón ejercitar el dispositivo electromecánico iniciará la rutina y como se muestra en la figura 140-3, el algoritmo de control emitirá la información a la plataforma IoT y será adquirida a su vez desde LabVIEW.

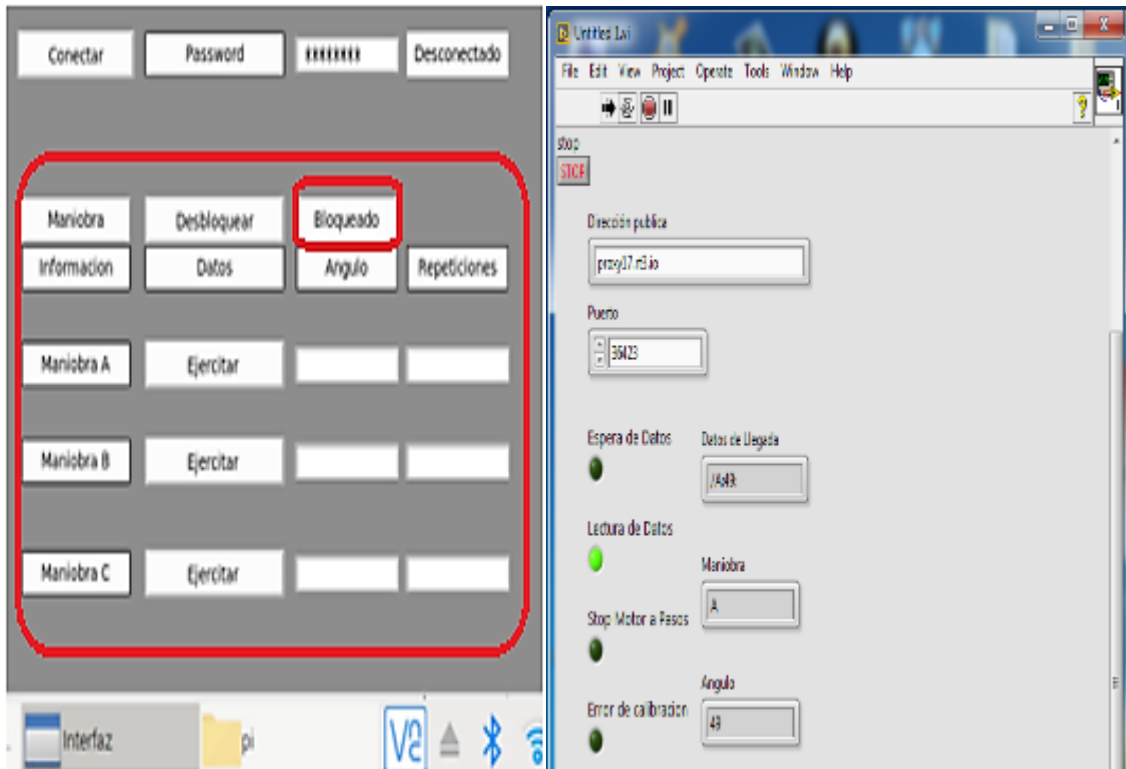


Figura 125-3: Resultado – Recepción remota de datos de la maniobra en LabVIEW

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Al ejecutar la instrucción de maniobra se desarrollan paralelamente el movimiento del equipo electromecánico y la visualización de igual manera del movimiento en el punto de acceso remoto integrado por las plataformas LabVIEW y SolidWorks.

La figura 141-3 simboliza la función de monitoreo remoto del prototipo, donde se verifica el inicio del movimiento del prototipo rehabilitador y también el modelo en 3D del brazo. El movimiento descrito en la figura 141-3 representa parte de la secuencia de flexoextensión muñeca determinado como requerimiento del trabajo.



Figura 126-3: Ejecución monitoreo remoto del rehabilitador - Flexoextensión de la muñeca

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

Las figuras 142-3, 143-3 y 144-3 representan los movimientos logrados como resultado de la implementación del prototipo rehabilitador. Se conjuga en las ilustraciones los movimientos requeridos con los ejecutados por el equipo.

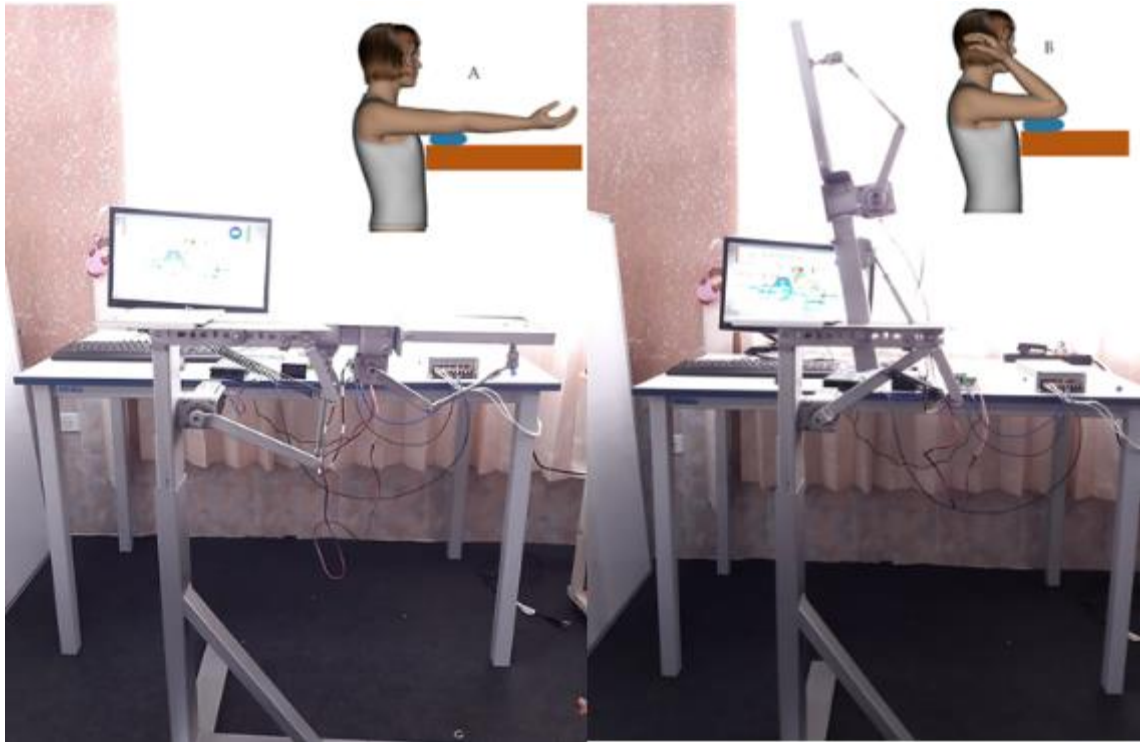


Figura 127-3: Resultado – Ejecución del Rehabilitador Flexoextensión del codo

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

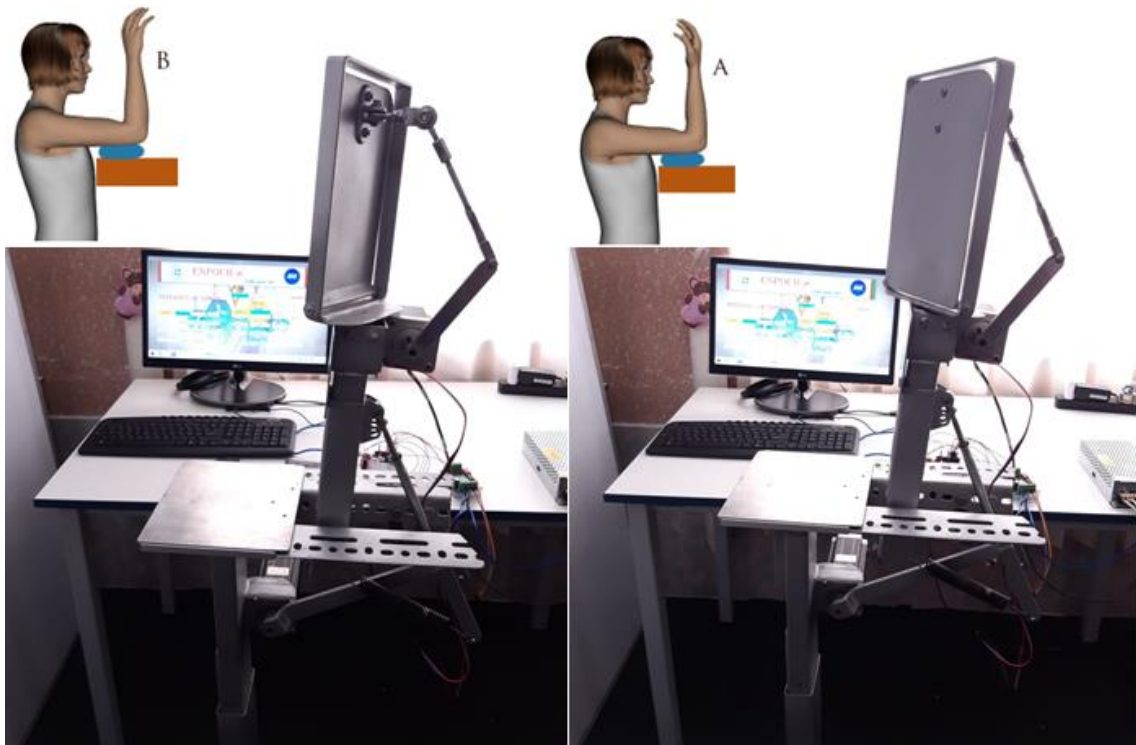


Figura 128-3: Resultado – Ejecución del Rehabilitador Pronosupinación

Realizado por: Tania Quinatoa, Wilmer Morales, 2020



Figura 129-3: Resultado – Ejecución del Rehabilitador Flexoextensión de muñeca

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

3.5.4 Consumo de corriente del prototipo

Se consideró importante conocer el consumo de energía de todo el prototipo, por lo que a continuación se describen en la Tabla 72-3 todas las cargas con su consumo individual para determinar el total.

Tabla 72-3: Consumo de los dispositivos del prototipo

# DE CARGAS	CARGA	CONSUMO INDIVIDUAL POR HORA (A-Wh)	CONSUMO TOTAL (Wh)
1	Raspberry Pi3	0.35 - 1,8	1,8
1	Arduino UNO R3	0.046 - 0,23	0,23
1	Motor NEMA 23	4,2 - 15,8	15,8
1	Motor NEMA 23	2,8 - 10,5	10,5
1	Motor NEMA 17	0.35 - 4,2	4,2
1	Monitor	50	50
			Total 82,53Wh

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

4. GESTIÓN DEL PROYECTO

En este capítulo se detallan el cronograma de actividades realizada en la implementación del prototipo rehabilitador electromecánico, el análisis financiero, los recursos humanos y materiales utilizados. Forman parte también las conclusiones y recomendaciones del presente trabajo de investigación.

4.1 Cronograma

Tabla 73-4: Cronograma.

Actividad	Mes 1		Mes 2		Mes 3		Mes 4		Mes 5		Mes 6	
	Semanas		Semanas		Semanas		Semanas		Semanas		Semanas	
	1-2	3-4	1-2	3-4	1-2	3-4	1-2	3-4	1-2	3-4	1-2	3-4
Etapa de Análisis y Recolección de la Información Necesaria	■	■										
Etapa de Selección de los Elementos			■	■								
Etapa de diseño del prototipo				■	■	■	■					
Etapa de implementación del prototipo						■	■	■	■			
Etapa de Análisis de resultados y pruebas									■	■		
Etapa de redacción del informe final				■	■	■	■	■	■	■	■	■
Etapa de Articulación Científica										■	■	■

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

4.2 Recursos y materiales: humanos, equipos, financiamiento

4.2.1 Costos

La siguiente tabla muestra el costo total para la construcción del prototipo rehabilitador electromecánico.

Tabla 74-3: Costos directos e indirectos del proyecto.

N°	Detalle	Cantidad	Valor Unitario (\$)	Total (\$)
1	Raspberry Pi3	1	110,00	110,00
2	Micro SD	1	12,00	12,00
3	Fuente de poder 24VCC- 10 ^a	1	20,00	20,00
4	NEMA 23 (425 oz.in) +DRIVER	1	160,00	160,00
5	NEMA 23 (269 oz.in) +DRIVER	1	80,00	80,00
6	NEMA 17 + DRIVER	1	20,00	20,00
7	Cable HDMI 3m	1	7,00	7,00
8	Arduino UNO R3	1	10,00	10,00
9	Kit de cables H-M, M-H	1	5,00	5,00
10	Material Estructural	1	400,00	400,00
			Total (USD)	824,00

Realizado por: Tania Quinatoa, Wilmer Morales, 2020

4.2.2 Talento Humano

Tania Cristina Quinatoa Chicaiza y Wilmer Luis Morales Rivera, son los ejecutores del trabajo. La integración de metodologías de la investigación y el compromiso personal permitieron la culminación exitosa del proyecto, consiguiendo alcanzar los requerimientos planteados y los objetivos.

Ing. Edwin Altamirano Santillán, Ing. Marco Antonio Viteri Barrera, director y miembro del trabajo de titulación quienes, con su conocimiento y experiencia colaboraron para el desarrollo y culminación de este proyecto.

Lic. Danny Fuenmayor especialista médico que brindó el asesoramiento y acompañamiento para validación del protocolo de rehabilitación electromecánico para pacientes con lesiones en el codo y posterior validación del prototipo.

4.2.3 Recursos Materiales

El hardware y el software empleados para el desarrollo del prototipo han sido presentados en los apartados anteriores, integrando varias plataformas con el fin de innovar y presentar una propuesta original y de utilidad para las personas que han sufrido alguna lesión en el codo.

El financiamiento fue cubierto por los estudiantes promotores de este trabajo, con la finalidad de presentar una idea innovadora que pueda inclusive servir como base para futuros trabajos que integren sistemas inteligentes de rehabilitación de bajo costo con un impacto de ayuda a sectores vulnerables de la sociedad.

CONCLUSIONES

Se logró la construcción de un prototipo de rehabilitación electromecánico tras traumatismos de codo, integrando plataformas hardware (Raspberry Pi3 y Arduino) y software (Python), se modeló haciendo uso de la herramienta CAD (SolidWorks) que integrada con LabView, permitieron el desarrollo de un sistema de control para gestionar la movilidad del prototipo, que combinado con una interfaz gráfica monitorea los movimientos establecidos para personas que sufren este tipo de lesión.

Se determinó mediante un estudio del arte en la teoría planteada por Traumatología Hellín y validada por la consulta a un especialista, que la flexoextensión del codo, pronosupinación y flexoextensión existe un protocolo de rehabilitación tras traumatismos de codo, mediante movimientos establecidos que contribuyeron como fundamento para el diseño e implementación del prototipo.

Mediante un análisis estático se validó las piezas de la estructura que fueron expuestas a cargas y/o sujeciones, obteniéndose un resultado satisfactorio en el diseño y la determinación del material que brindó garantía para su construcción.

Se realizaron las pruebas al prototipo consiguiendo una funcionalidad total del mismo, identificando una respuesta en tiempo real en la ejecución de los movimientos al enviar una instrucción desde el HMI local, un retraso de 1 segundo en la adquisición remota de la información en Labview y 3 segundos de retardo en la respuesta en la simulación de movimientos en Solidworks.

Se determinó que la flexibilidad para regular la altura del prototipo rehabilitador es de 15cm teniendo una altura mínima de 63cm y una máxima de 78cm permitiendo adaptarse a cualquier silla disponible en la casa de un paciente, así como también regular la longitud del soporte del antebrazo en un rango de 7cm obteniendo una cobertura para pacientes con una longitud mínima de 20 cm y una máxima de 27cm en el antebrazo.

RECOMENDACIONES

La metodología para la implementación del prototipo de rehabilitador electromecánico para rehabilitación de codo desarrollado en este trabajo puede ser replicada en la construcción de cualquier tipo de rehabilitador, el principio de control y monitoreo puede servir de base para trabajos futuros.

El diseño del prototipo se ve limitado a la ejecución de sólo tres movimientos planteados como requerimientos para el caso de estudio. Existiendo la posibilidad de que al diseño se lo podría modificar para añadirle más articulaciones que permitan nuevos movimientos.

El presente trabajo empleó plataformas potenciales como LabVIEW y SolidWorks que requieren licenciamiento para su utilización, quedando la posibilidad de investigar e implementarlo mediante el uso de software open source, generando sistemas o prototipos de bajo costo.

Como medida de prevención se debe evitar que el prototipo se encuentre en lugares no húmedos para la conservación de sus componentes y una mejor utilización del mismo.

La recomendación del especialista es la mejora del tapizado y material de construcción más flexibles más la inserción de correas de sujeción para comodidad de la persona que ha sufrido este tipo de lesión.

GLOSARIO

Internet de las Cosas (IoT): Una infraestructura de red global dinámica con capacidades de autoconfiguración basadas en protocolos de comunicación estándar e interoperables donde las cosas físicas y virtuales tienen identidades, atributos físicos y personalidades virtuales, y usan interfaces inteligentes de manera integrada en el red de información referimos a la biomecánica del pie en el cual está involucrada la anatomía de las extremidades inferiores del mismo, por ende, es necesario tener un conocimiento previo desde el punto de vista anatómico. (Llaneza González, 2018, p. 20)

LabVIEW: Es una herramienta de programación gráfica. Originalmente este programa estaba orientado para aplicaciones de control de equipos electrónicos usados en el desarrollo de sistemas de instrumentación, lo que se conoce como instrumentación virtual. (Vizcaíno y Sebastián, 2011, p. 22)

Arduino: es una plataforma embebida que contiene como elemento principal un microcontrolador reprogramable, el mismo que permite la adquisición de señales, procesamiento mediante sentencias programadas y la emisión de señales de control. Este sistema embebido además se acopla a una amplia gama de sensores de la misma marca u otra que proporcionen señales del tipo digital y/o analógico para ser procesadas y según su evaluación por medio de interfaces de potencia poder realizar el control de actuadores sin importar la carga que manejen. (Arduino, 2020)

Microcontrolador: Se define como un dispositivo electrónico embebido o un circuito integrado que encapsula varios recursos, la ventaja de este elemento que se puede programar de acuerdo a la necesidad que suscite el entorno en el que se lo va a emplear, es capaz de ejecutar con total autonomía previo a la carga de instrucciones predefinidas cualquier tipo de trabajo que se lo encomiende. (Artero, 2013, p. 62)

Raspberry Pi: Es una minicomputadora de la gama de hardware libre que constituida de una placa simple empleada en el ámbito educativo para estudiantes del área de informática y electrónica principalmente, el soporte de software para este dispositivo es de código abierto. Está constituida esencialmente por un procesador, memoria RAM, puertos USB, GPU, HDMI, ETHERNET, requiere de una micro SD pues no incluye memoria interna. La potencialidad de la Raspberry Pi la hace suficientemente potente como para poder tomar el lugar de una computadora de escritorio. (A, B rev 1, B rev 2, B+). (Salcedo & Cendrós, 2006, pp. 63-64)

BIBLIOGRAFÍA

ARTERO, T.Ó. *ARDUINO Curso básico de formación*. Alfaomega, 2013. DOI 9788494072505. p. 62.

ANGULO M, Álvarez A, Fuentes Y. *Biomecánica clínica Biomecánica de la Extremidad Superior Exploración del Codo*. 2011. p. 22.

BARRIO M. *Internet de las Cosas* [en línea]. Madrid - Spain: REUS; 2018. [Consulta: 20 septiembre 2020]. Disponible en: https://www.editorialreus.es/static/pdf/primeraspaginas_9788429020380_internetdelascosas.pdf

CELEDÓN HJ. *Diseño mecatrónico de un robot exoesqueleto de extremidad superior para rehabilitación de personas con discapacidad parcial en el codo* titulación [en línea] (Trabajo de titulación) (Ingeniería). Universidad Santo Tomás. Bogotá – Colombia. 2016. [Consulta: 20 septiembre 2020]. Disponible en: <https://pdfs.semanticscholar.org/22e0/1c806c9ac4e31be36b867b1d286478ad7bf1.pdf>

CIBSIR. *Memorias del I Congreso Internacional de Bioingeniería y Sistemas Inteligentes de Rehabilitación* [en línea]. Quito - Ecuador: Editorial Universitaria Abya-Yala. 2017. 567 p. [Consulta: 04 de marzo 2020]. Disponible en: <https://dspace.ups.edu.ec/bitstream/123456789/17080/1/Memorias%20del%20I%20Congreso%20Internacional%20de%20bioingenieria%20y%20sistemas%20inteligentes%20de%20rehabilitacion.pdf>

Echevarría S. *Traumatología y ortopedia* [en línea]. México, D.F., MEXICO: Editorial Alfil, S. A. de C. V.; 2013 [Consulta: 04 de enero de 2020]. Disponible en: <http://ebookcentral.proquest.com/lib/epochsp/detail.action?docID=3220007>.

EVANS D. *Internet de las cosas Cómo la próxima evolución de Internet lo cambia todo* [en línea]. (Cisco Internet Business Solutions Group (IBSG)). 2011. [Consulta: 04 de enero de 2020] Disponible en: https://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf

KREBS HI, Ferraro M, Buerger SP, Newbery MJ, Makiyama A, Sandmann M, et al. *Rehabilitation robotics: pilot trial of a spatial extension for MIT-Manus*. J Neuro Engineering Rehabil. 2004. p 5

LAJARA JR, Pelegri J. *LabVIEW: Entorno gráfico de programación* [en línea]. Segunda Ediciones Técnicas. 2011 [Consulta: 29 de diciembre de 2019]. Disponible en: https://books.google.com.ec/books?hl=es&lr=&id=ZFQua3-eeQEC&oi=fnd&pg=PA21&dq=COMUNICACION+SERIAL+EN+LABVIEW&ots=qI1xU8SFhu&sig=gfJ14V93tVGnHRzA5eglxa5zszI&redir_esc=y#v=onepage&q&f=false

LIÑÁN Colina A, Vives Á, Bagula A, Zennaro M, Pietrosemoli E. *Internet de las cosas* [en línea]. 2015. [Consulta: 24 de febrero de 2020]. Disponible en: <http://wireless.ictp.it/Papers/InternetdelasCosas.pdf>

LÓPEZ M. *Internet de las cosas La transformación digital de la sociedad* [Internet]. ESPAÑA: RA-MA Editorial; 2019. [Consulta: 21 de abril de 2020]. Disponible en: <http://www.tecnolibro.es/ficheros/descargas/9788499647999.pdf>

LÓPEZ Aldea Eugenio. *Raspberry Pi Fundamentos y aplicaciones*. Madrid, SPAIN: RA-MA Editorial; 2017. p 264.

LLANEZA González P. *Seguridad y responsabilidad en la Internet de las cosas (IoT)*. ESPAÑA: Wolters Kluwer España, S.A. 2018. p. 360.

MURILLO S. *The impact of IoT on prevention, assistance, detection and rehabilitation of patients with cognitive impairment*. 2015. p. 17.

RISCHPATER R. *Application Development with Qt Creator*. Packt Publishing Ltd. 2013. p. 222.

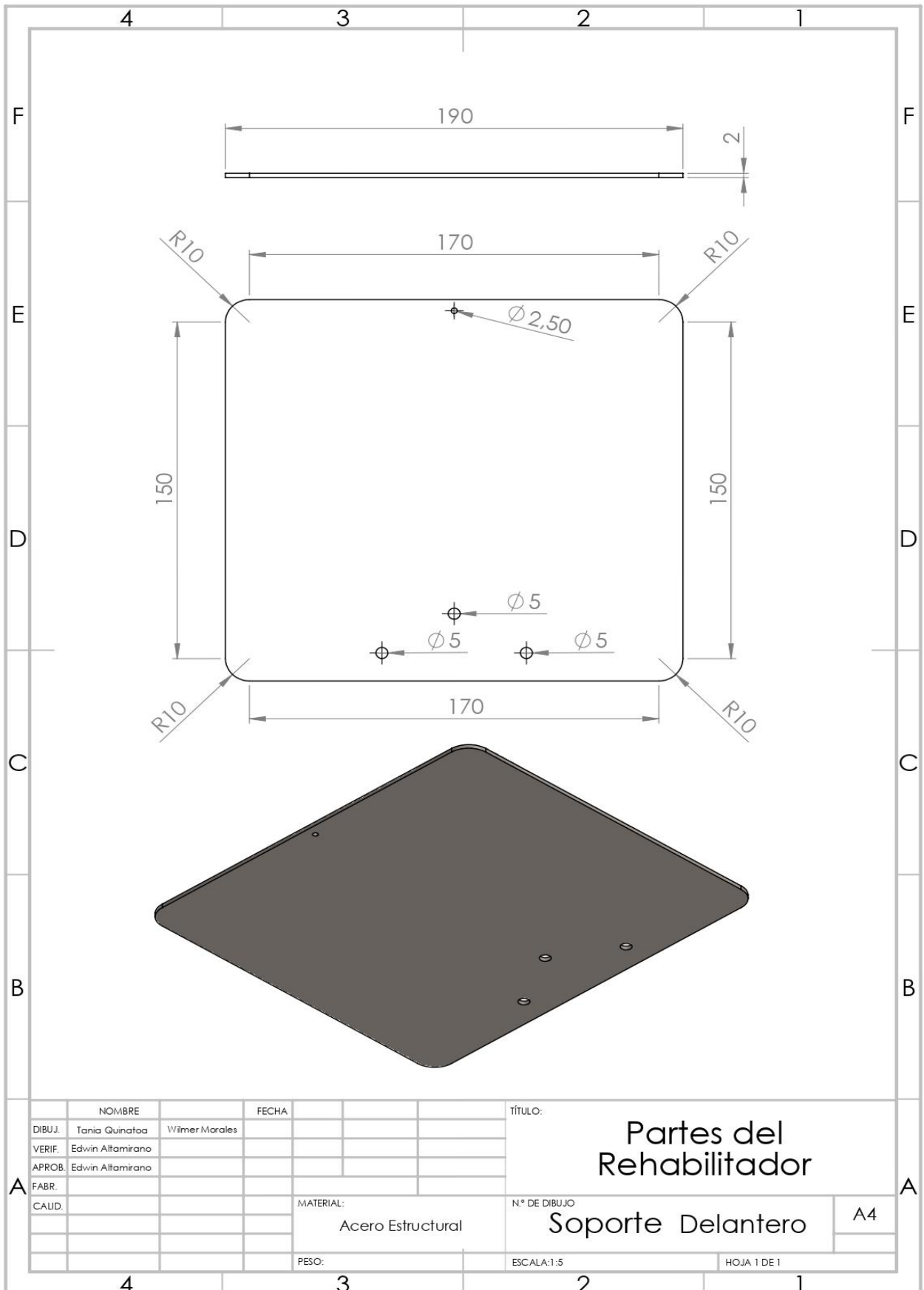
RODRÍGUEZ Vidal C. *Diseño mecánico con SolidWorks 2015* [en línea]. Madrid, SPAIN: RA-MA Editorial; 2015 [Consulta 2 de julio de 2020]. Disponible en: <http://ebookcentral.proquest.com/lib/epochsp/detail.action?docID=5758939>

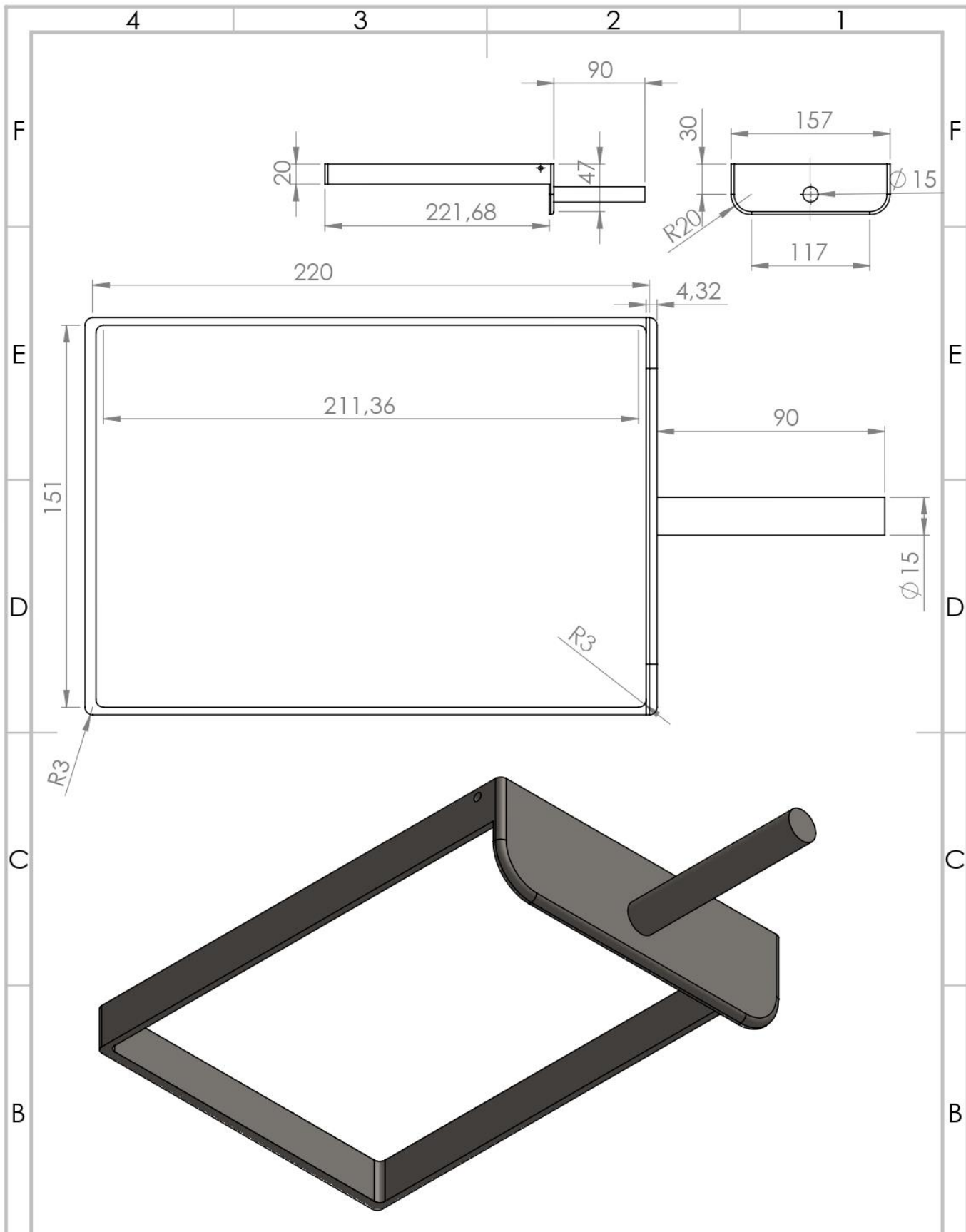


Firmado electrónicamente por:
**JHONATAN RODRIGO
PARREÑO UQUILLAS**

ANEXOS

ANEXO A: LÁMINAS – PLANOS DE PIEZAS





NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Quinatoa	Wlmer Morales		<h1>Partes del Rehabilitador</h1>	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.				MATERIAL:	N° DE DIBUJO
				Acero Estructural	Soporte Delantero
				PESO:	ESCALA: 1:5
					HOJA 1 DE 1

A

A

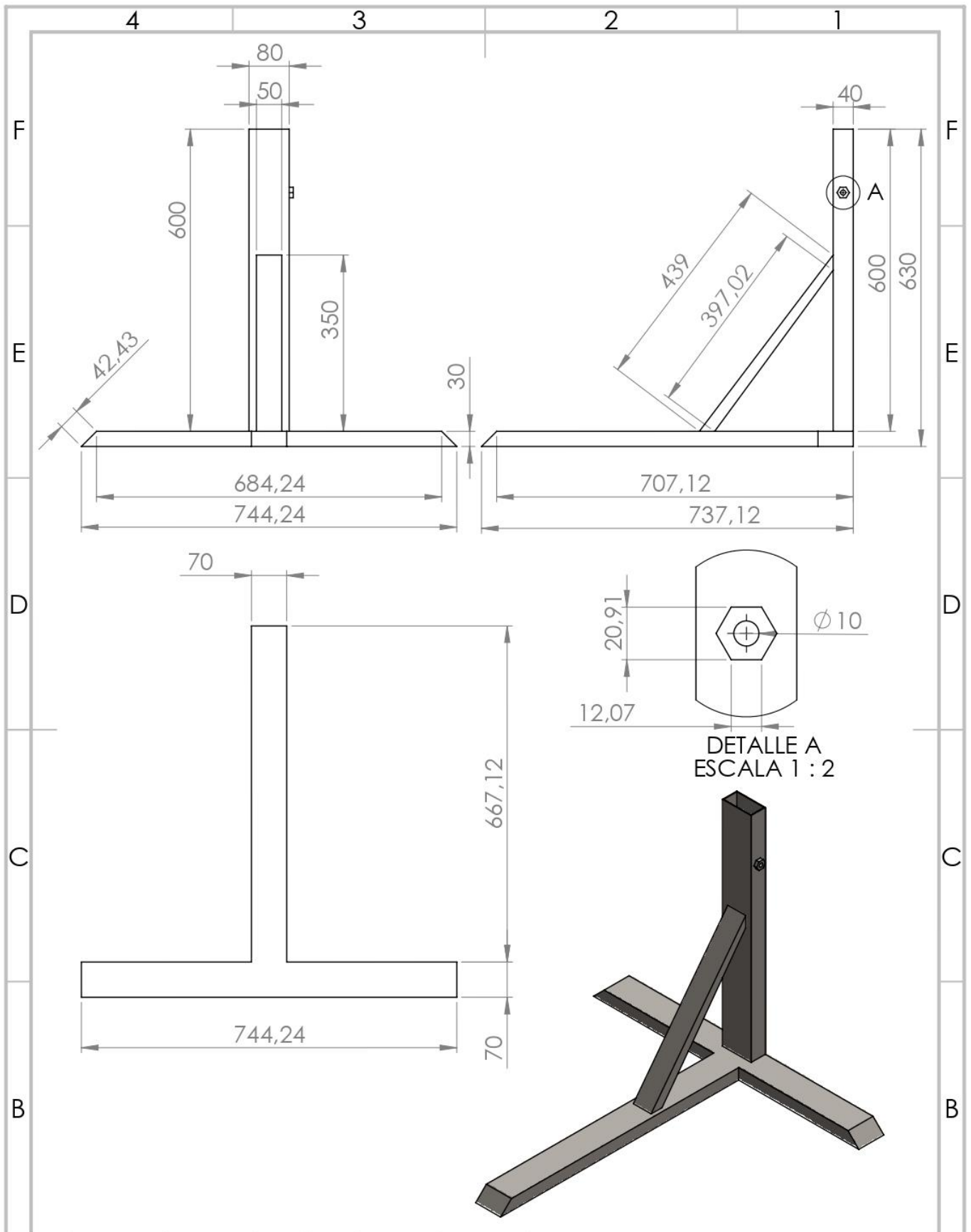
A4

4

3

2

1



	NOMBRE	FECHA			
DIBUJ.	Tania Quinatoa	Wlmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.					

TÍTULO:
Partes del Rehabilitador

N.º DE DIBUJO
Base Principal

MATERIAL:
Acero Estructural

PESO:

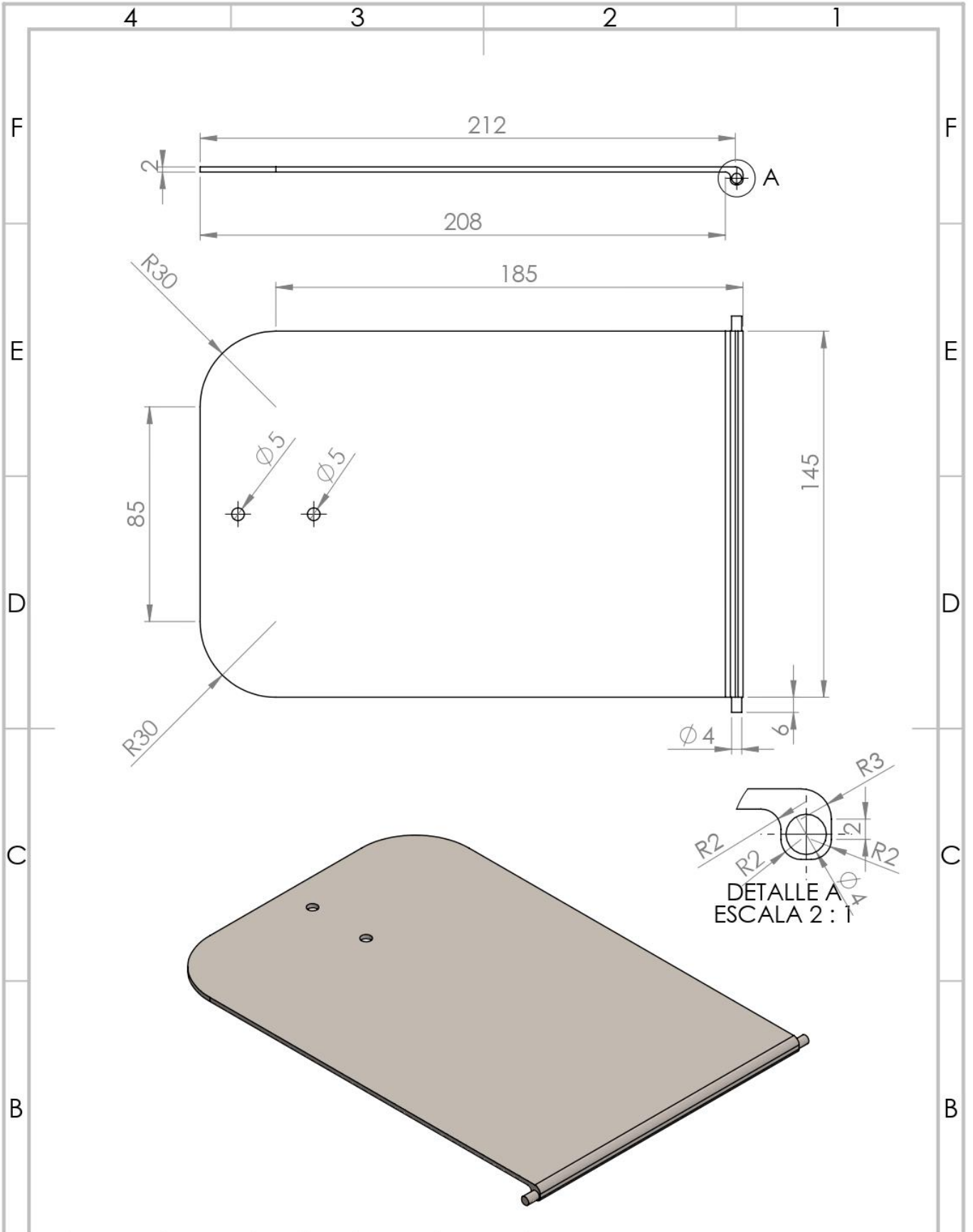
ESCALA: 1:10

HOJA 1 DE 1

A4

A

A



	NOMBRE	FECHA			
DIBUJ.	Tania Quinatoa	Wlmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.					

TÍTULO:
Partes del Rehabilitador

N° DE DIBUJO
Soporte Delantero

MATERIAL:
 Acero Estructural

PESO:

ESCALA: 1:5

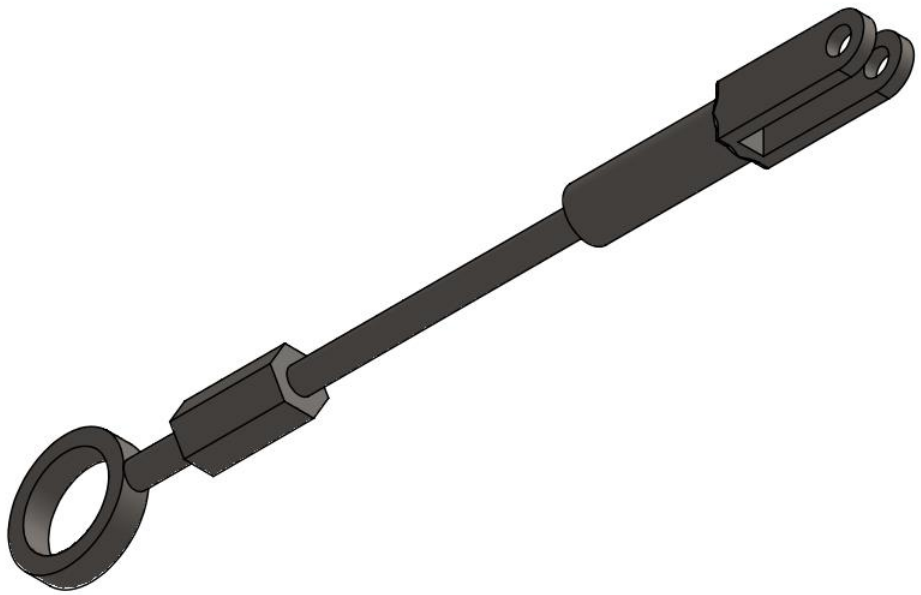
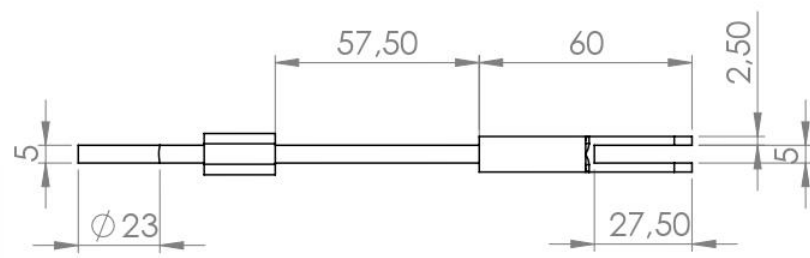
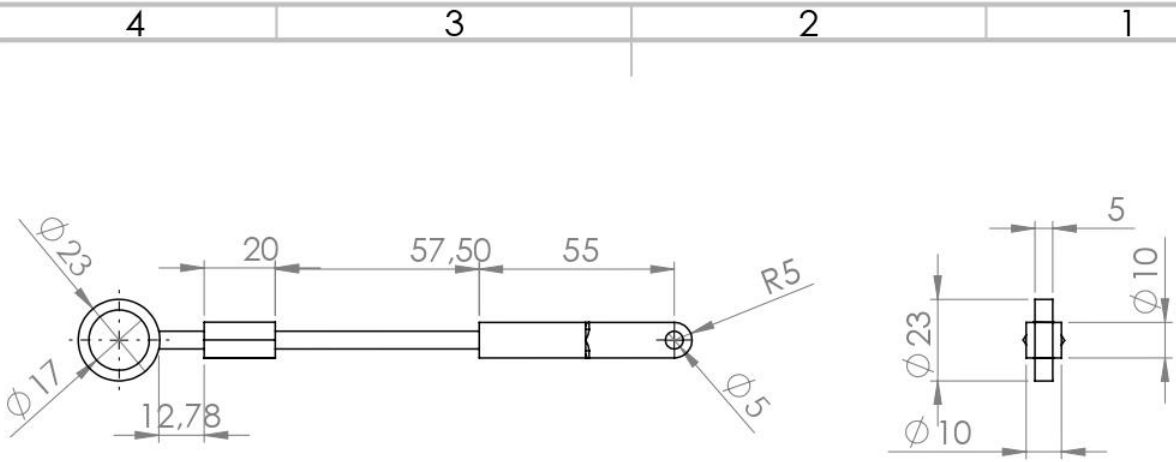
HOJA 1 DE 1

A4

A

A

4 3 2 1



	NOMBRE	FECHA			
DIBUJ.	Tania Guinatoa	Wilmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUD.					

TÍTULO:	Partes del Rehabilitador	
N° DE DIBUJO		
MATERIAL:	ASTM A 36	A4
PESO:	ESCALA: 1:2	HOJA 1 DE 1

4 3 2 1

F F

E E

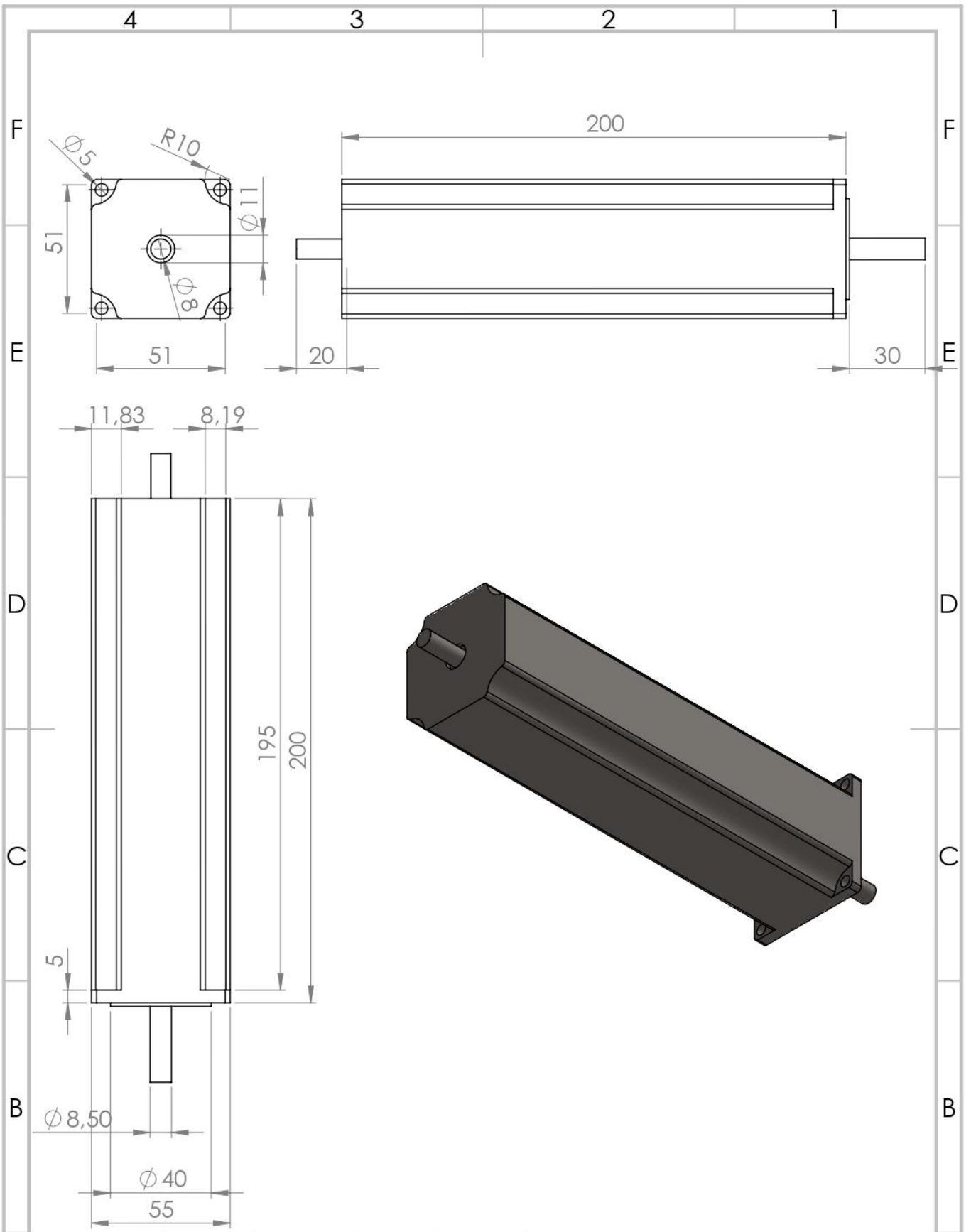
D D

C C

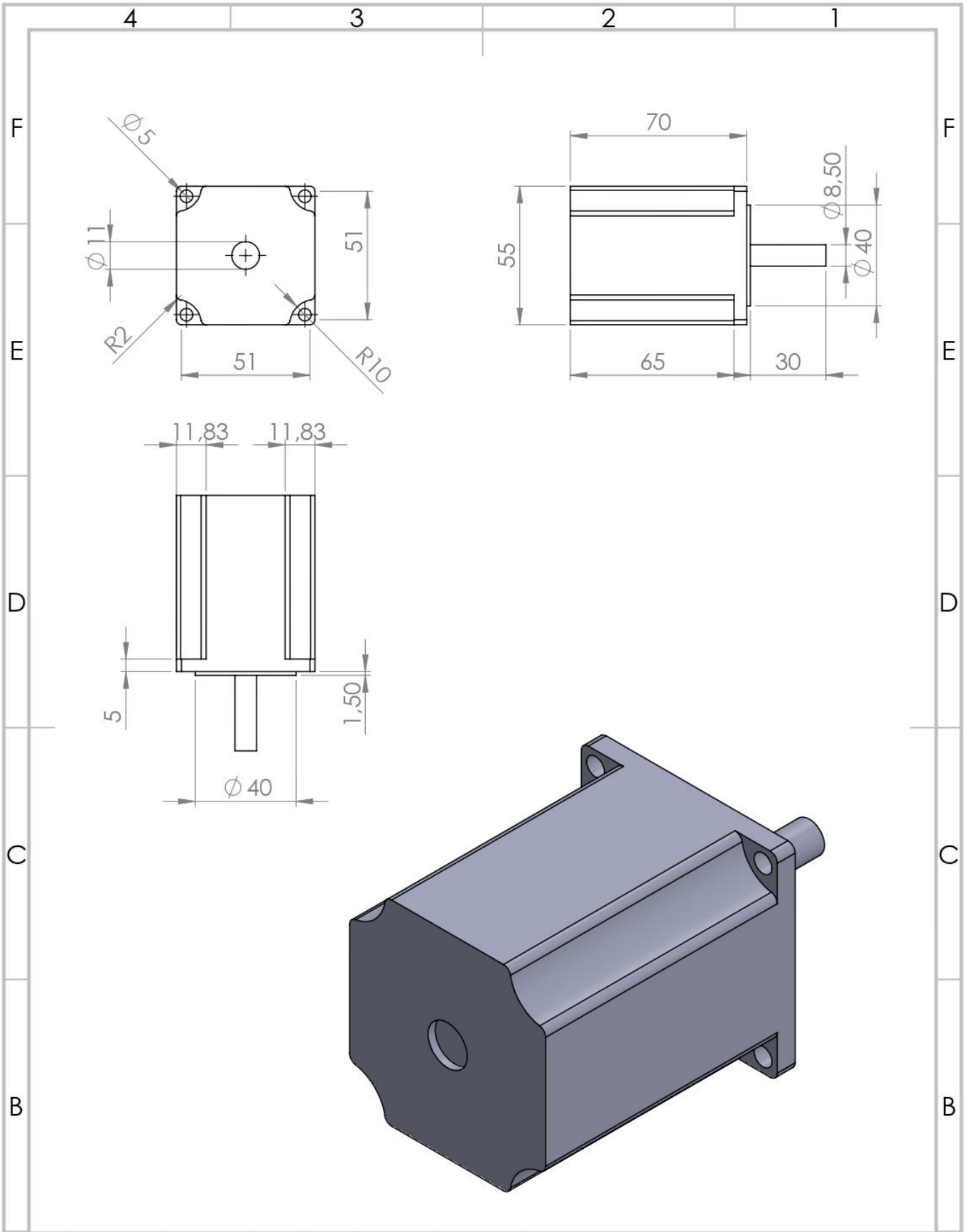
B B

A A

4 3 2 1



NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Guinatao	Wlmer Morales		<h1>Partes del Rehabilitador</h1>	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUD.		MATERIAL:	N° DE DIBUJO		A4
		Varios	<h2>Motor Antebrazo</h2>		
		PESO:	ESCALA: 1:5	HOJA 1 DE 1	



	NOMBRE	FECHA		TÍTULO:	
DIBUJ.	Tania Quinatoa	Wlmer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.		MATERIAL:	Varios	N° DE DIBUJO	Motor mano
		PESO:		ESCALA:1:2	HOJA 1 DE 1

A

A

A4

4 3 2 1

F

E

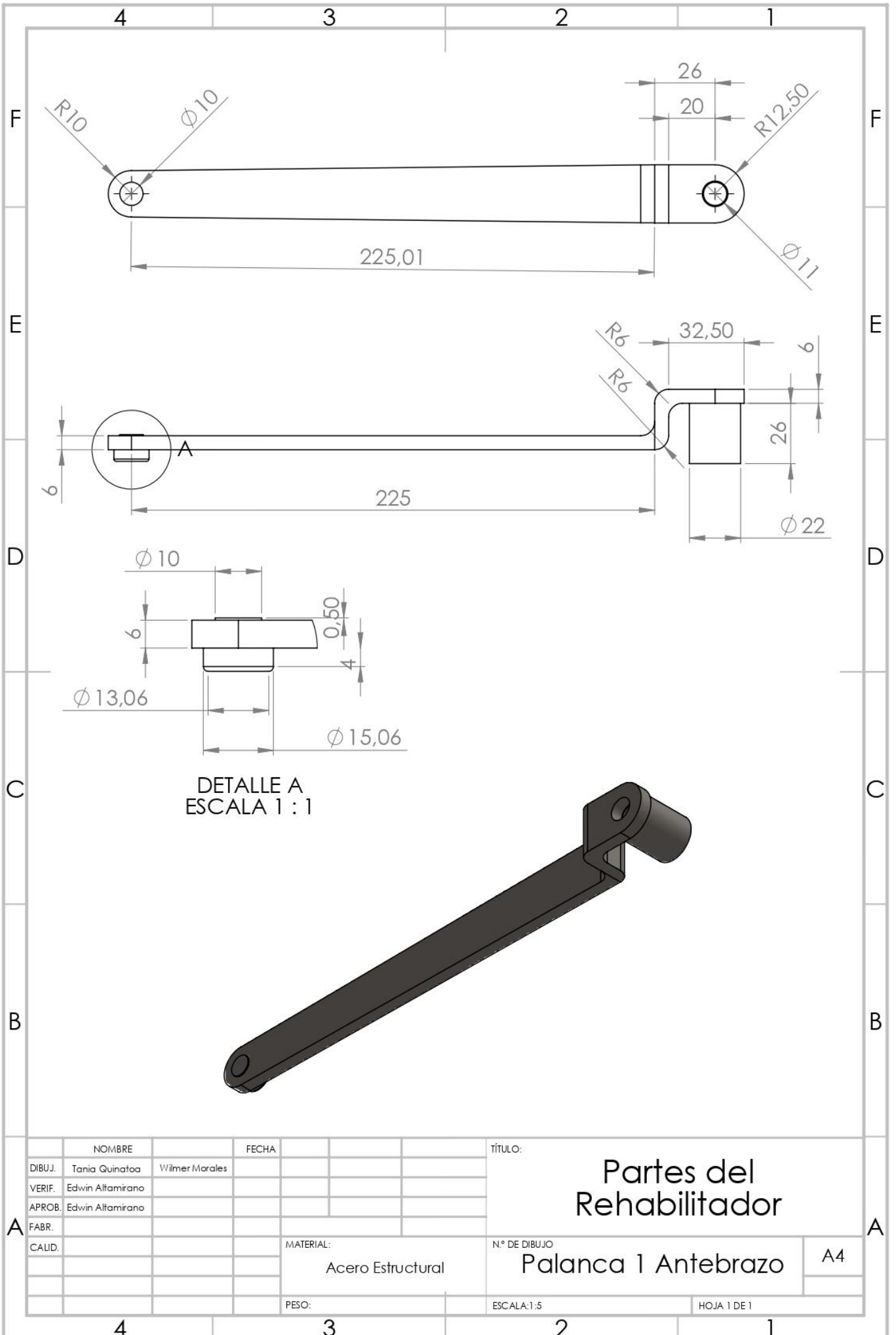
D

C

B

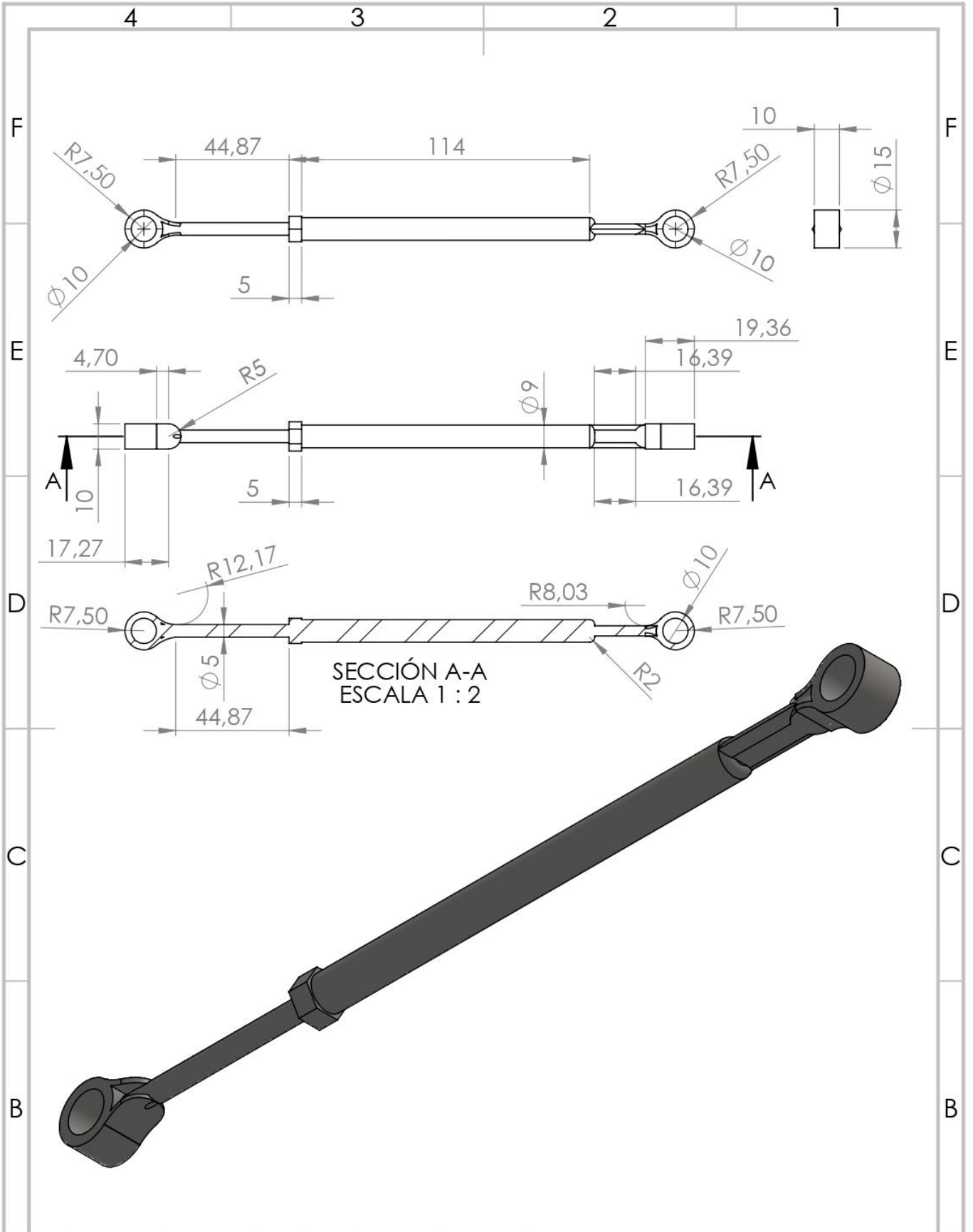
A

4 3 2 1

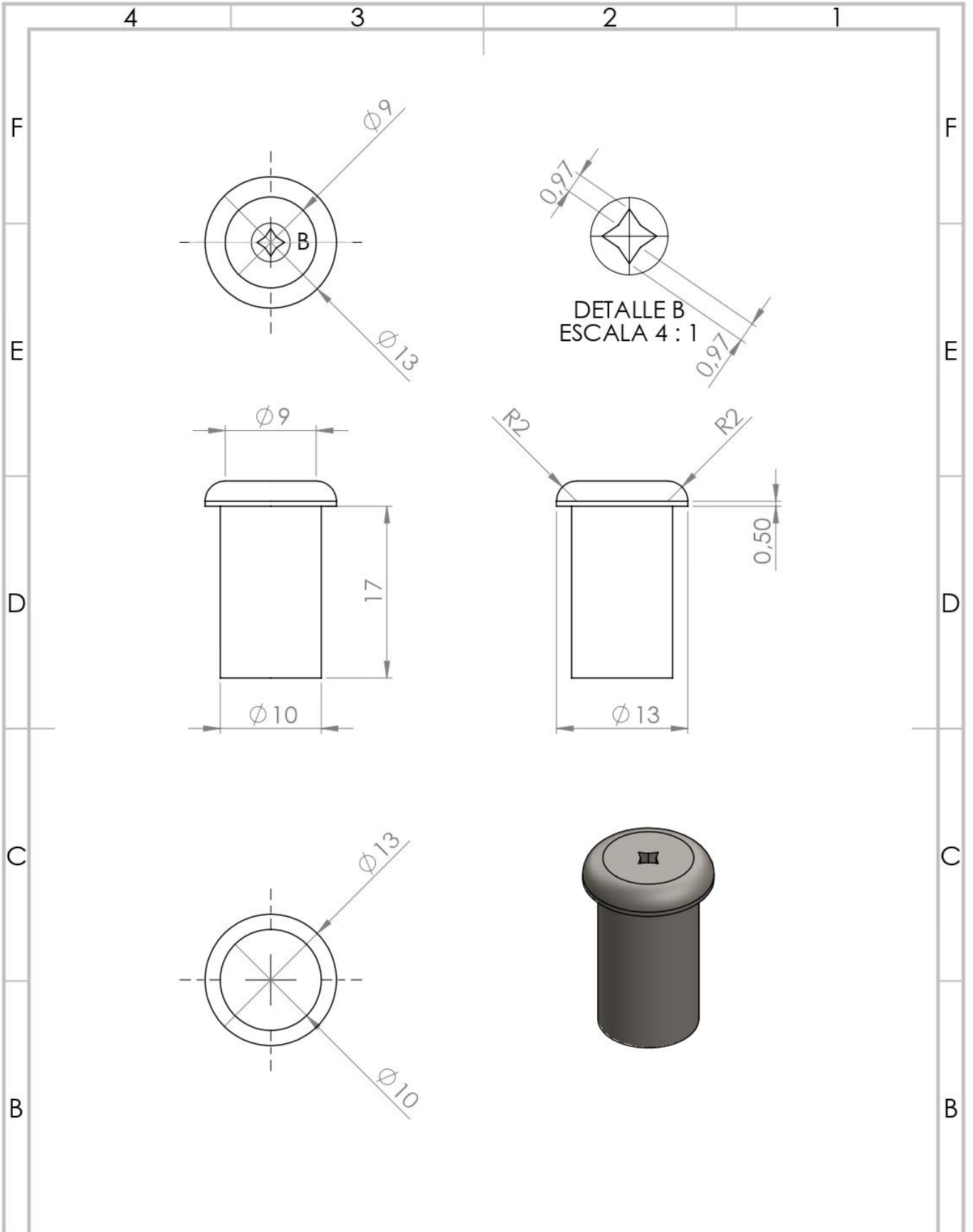


DETALLE A
ESCALA 1 : 1

NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Quinatoa	Wlmer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.				N.º DE DIBUJO	
CAUID.				Palanca 1 Antebrazo	
MATERIAL:			ESCALA: 1:5		
Acero Estructural			HOJA 1 DE 1		
PESO:			A4		



NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Quinatoa	Wáimer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.				N° DE DIBUJO	
CAUID.			MATERIAL:	Palanca Antebrazo	
			AISI 304	A4	
			PESO:	ESCALA:1:5	HOJA 1 DE 1



	NOMBRE	FECHA			
DIBUJ.	Tania Quinatoa	Wlmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.					

TÍTULO:
Partes del Rehabilitador

N° DE DIBUJO
Pasador Antebrazo

MATERIAL:
ASTM A 36

PESO:

ESCALA:2:1

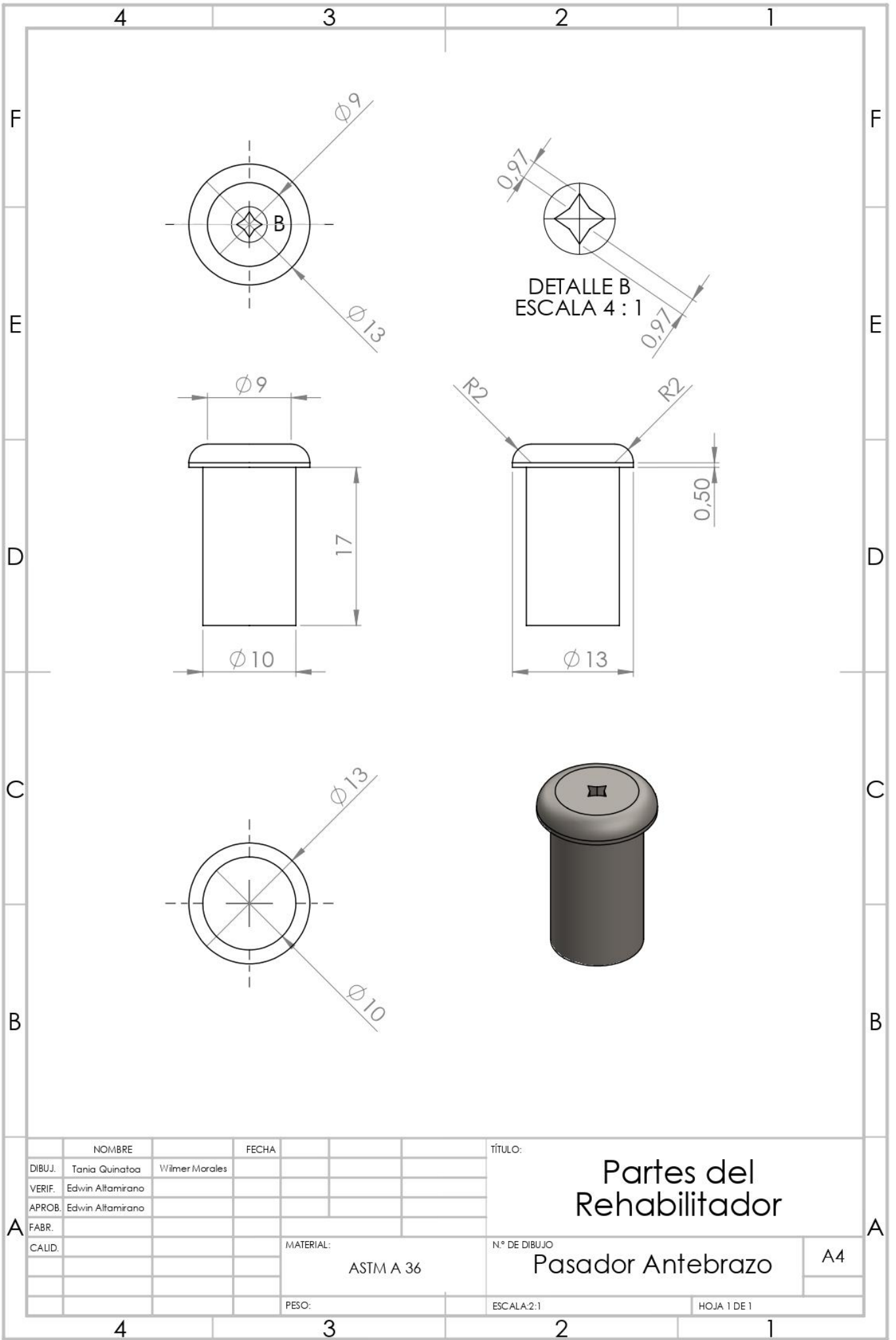
HOJA 1 DE 1

A4

A

A

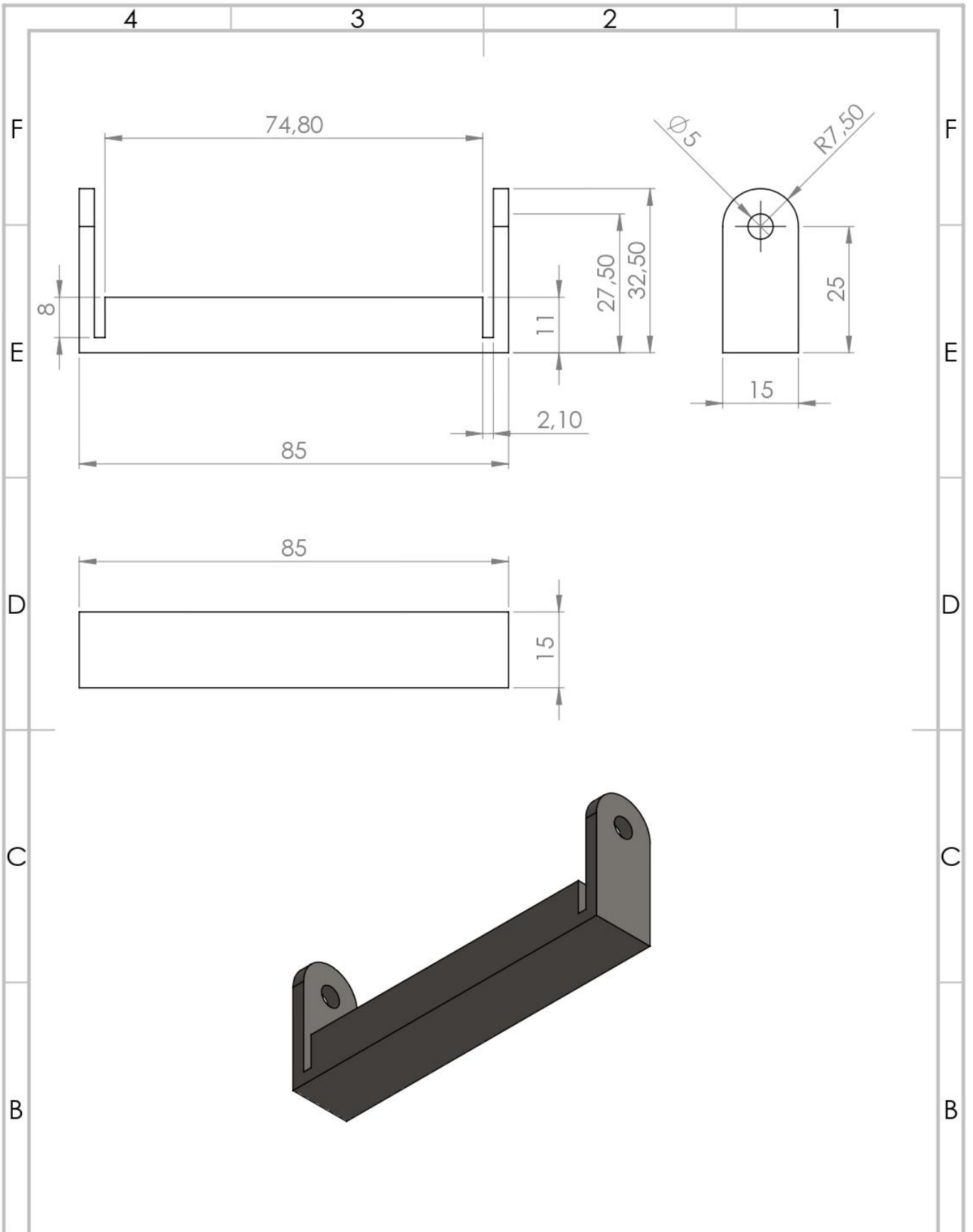
4 3 2 1



	NOMBRE		FECHA
DIBUJ.	Tania Quinatoa	Wilmer Morales	
VERIF.	Edwin Altamirano		
APROB.	Edwin Altamirano		
FABR.			
CAUID.			

TÍTULO:		<h1>Partes del Rehabilitador</h1>	
MATERIAL:		N° DE DIBUJO	
ASTM A 36		Pasador Antebrazo	
PESO:		ESCALA: 2:1	HOJA 1 DE 1

A4



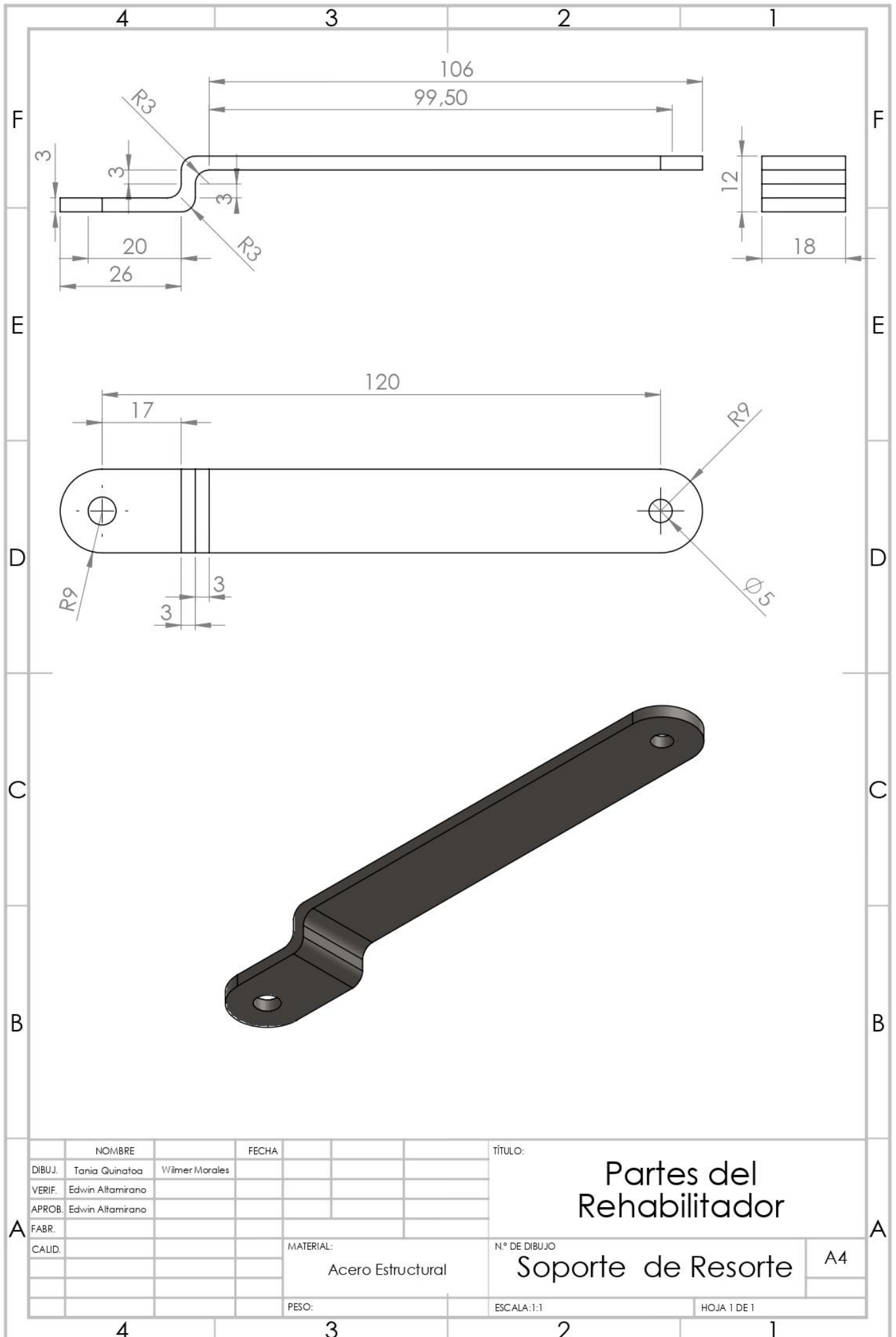
NOMBRE		FECHA	
DIBUJ.	Tania Quinatoa	Wlmer Morales	
VERIF.	Edwin Altamirano		
APROB.	Edwin Altamirano		
FABR.			
CAUID.			

TÍTULO:		<h1>Partes del Rehabilitador</h1>
MATERIAL:		
ASTM A 36		N° DE DIBUJO
		Soporte en U
PESO:		ESCALA: 1:1
		HOJA 1 DE 1

A

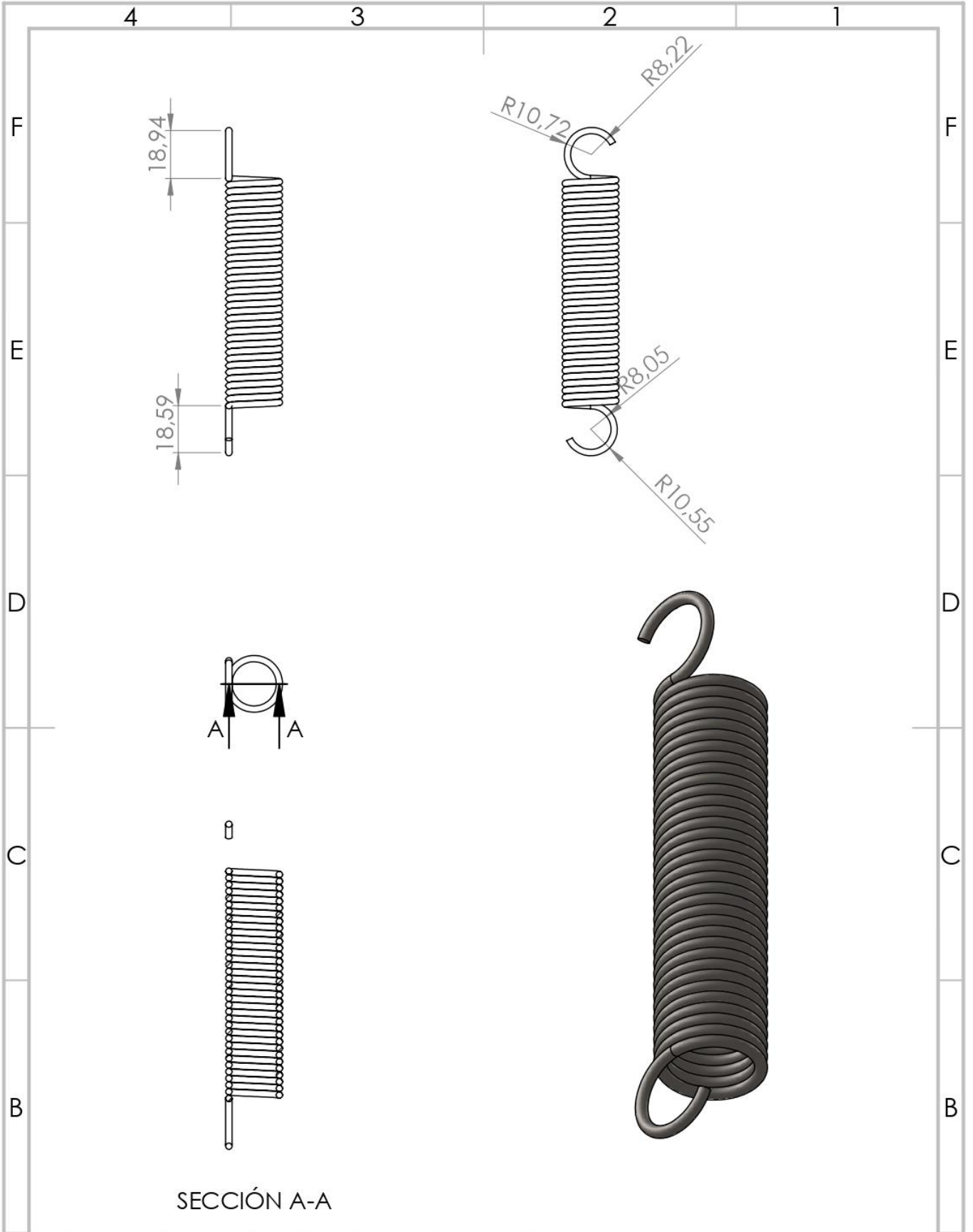
A

A4



	NOMBRE	FECHA			
DIBUJ.	Tania Guinatao	Wilmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.					

TÍTULO:		<h2>Partes del Rehabilitador</h2>	
N° DE DIBUJO			
MATERIAL:		Soporte de Resorte	
PESO:		ESCALA: 1:1	HOJA 1 DE 1



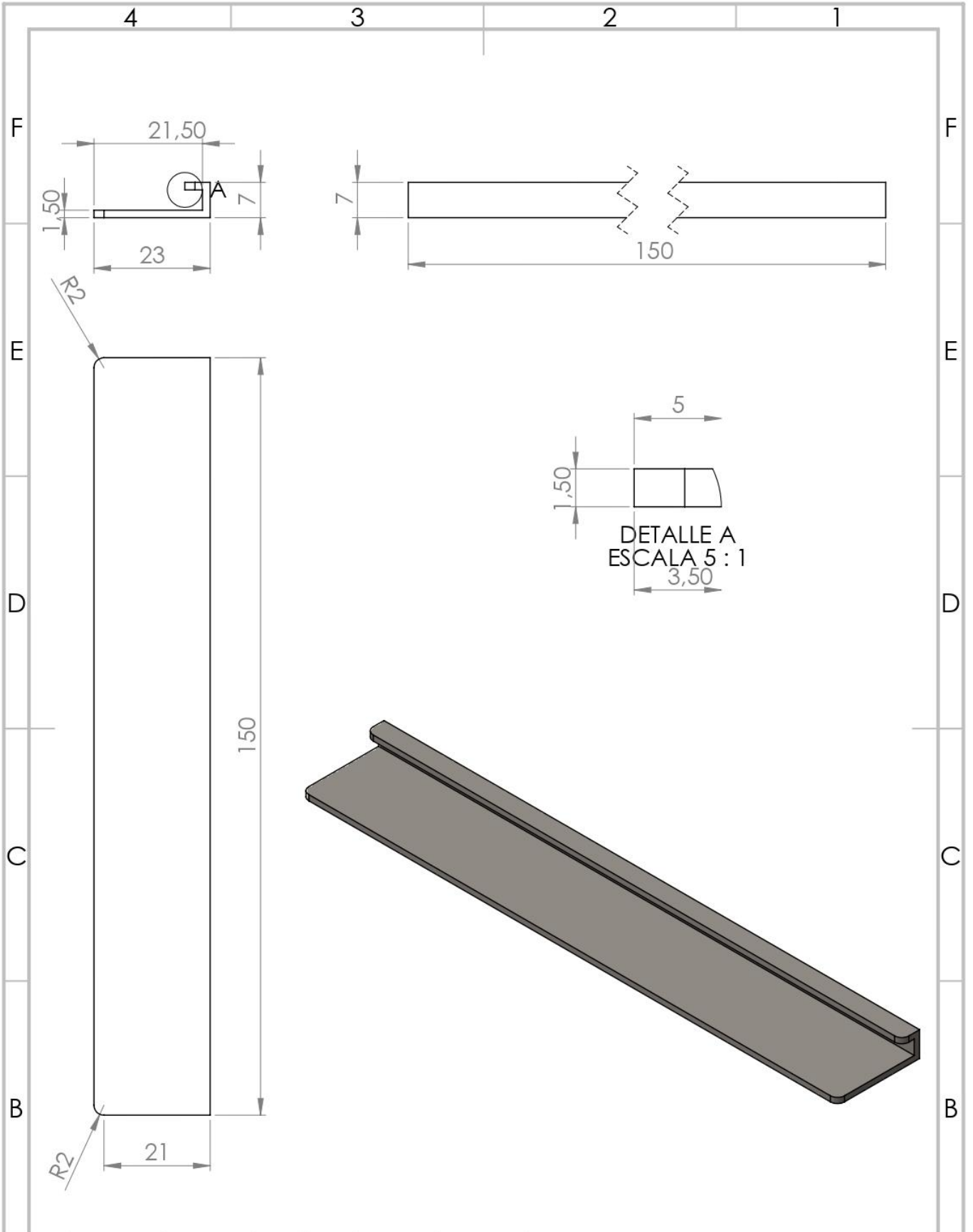
SECCIÓN A-A

NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Guinotoa	Wlmer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.			MATERIAL:	N° DE DIBUJO	RESORTE
			ASTM A36 Diámetro 20, 34 vueltas		
			PESO:	ESCALA: 1:2	HOJA 1 DE 1

A

A

A4



	NOMBRE	FECHA			
DIBUJ.	Tania Quinatoa	Wlmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUD.					

TÍTULO:
Partes del Rehabilitador

N° DE DIBUJO
Soporte del brazo

MATERIAL:
Acero Estructural

PESO:

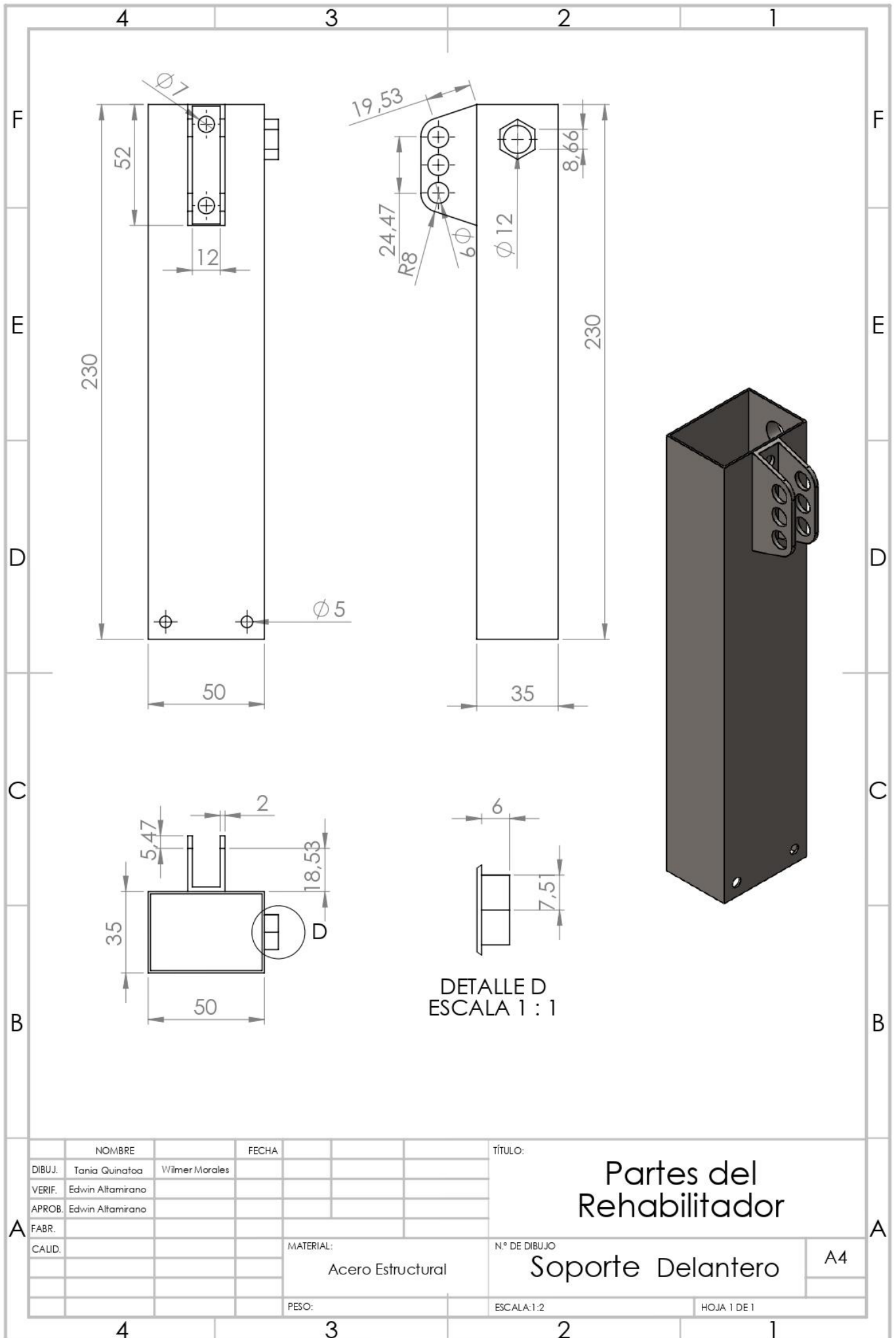
ESCALA: 1:2

HOJA 1 DE 1

A4

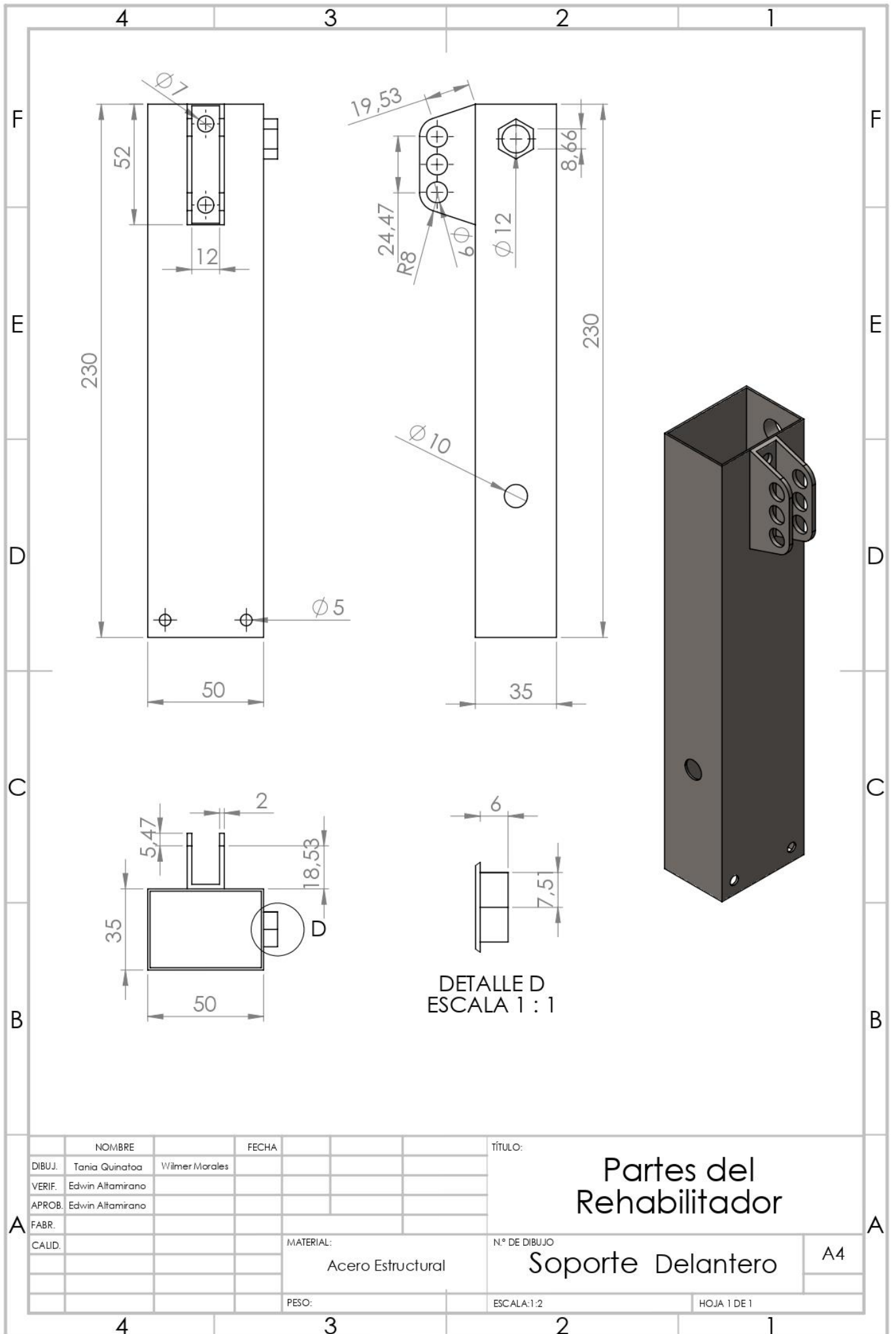
A

A



NOMBRE		FECHA	
DIBUJ.	Tania Quinatoa Wilmer Morales		
VERIF.	Edwin Altamirano		
APROB.	Edwin Altamirano		
FABR.			
CALID.			
MATERIAL:		Acero Estructural	
PESO:			

TÍTULO:	
Partes del Rehabilitador	
Soporte Delantero	
N.º DE DIBUJO	A4
ESCALA: 1:2	HOJA 1 DE 1

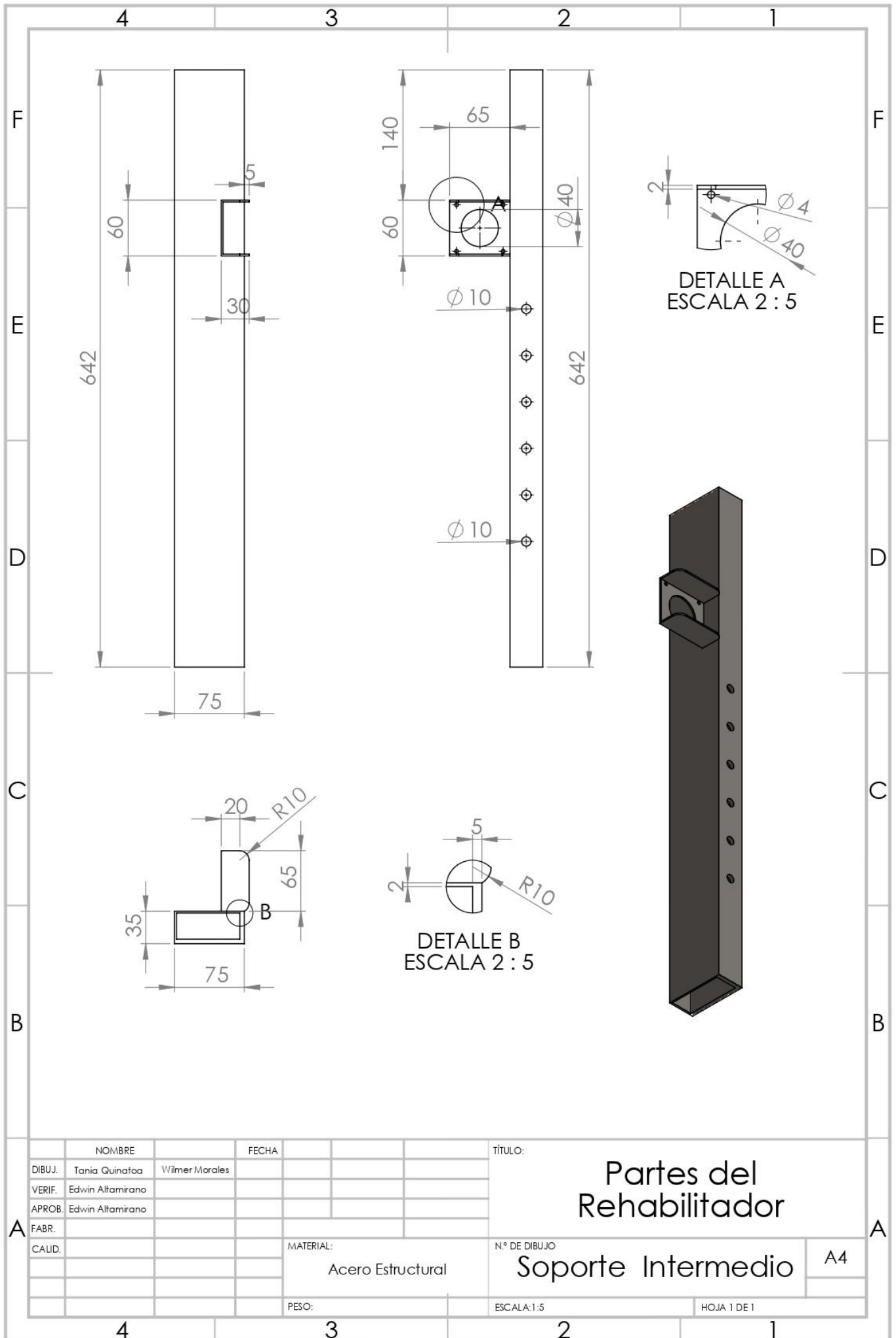


NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Guinatoa	Wilmer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.				N° DE DIBUJO	
CAUD.				Soporte Delantero	
				MATERIAL:	
				Acero Estructural	
				PESO:	
				ESCALA: 1:2	
				HOJA 1 DE 1	

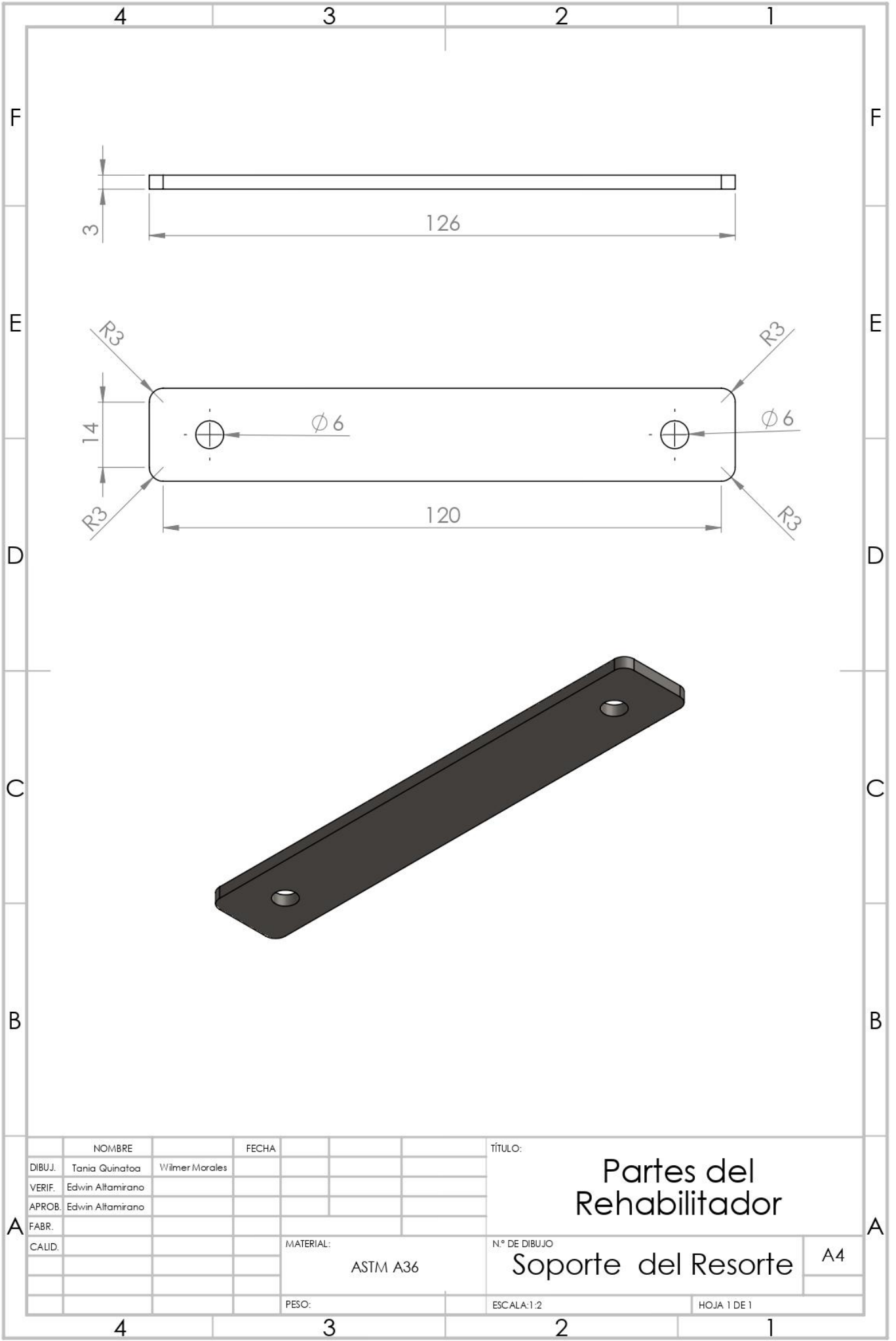
Partes del Rehabilitador

Soporte Delantero

A4



NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Guinotoa	Wilmer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.				Soporte Intermedio	
CAUD.					
		MATERIAL:		N° DE DIBUJO	
		Acero Estructural		A4	
		PESO:		ESCALA: 1:5	
				HOJA 1 DE 1	



	NOMBRE	FECHA			
DIBUJ.	Tania Quinatoa	Wlmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.					

TÍTULO:
Partes del Rehabilitador

N° DE DIBUJO
Soporte del Resorte

MATERIAL:
ASTM A36

PESO:

ESCALA: 1:2

HOJA 1 DE 1

A4

A

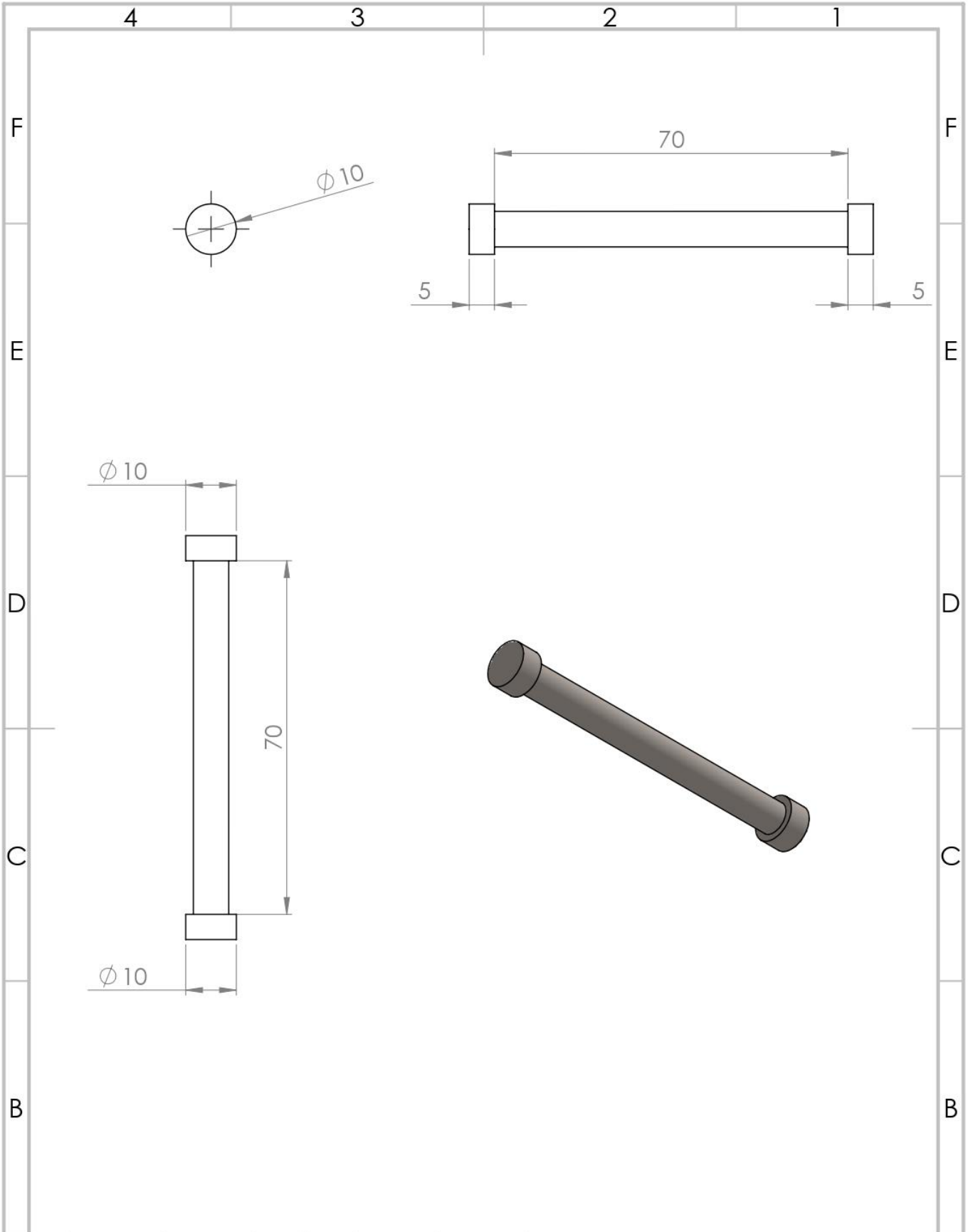
A

4

3

2

1



	NOMBRE	FECHA			
DIBUJ.	Tania Quinatoa	Wlmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.					

TÍTULO:
Partes del Rehabilitador

N° DE DIBUJO
Soporte del Seguro

MATERIAL:
Acero Aleado

PESO:

ESCALA: 1:1

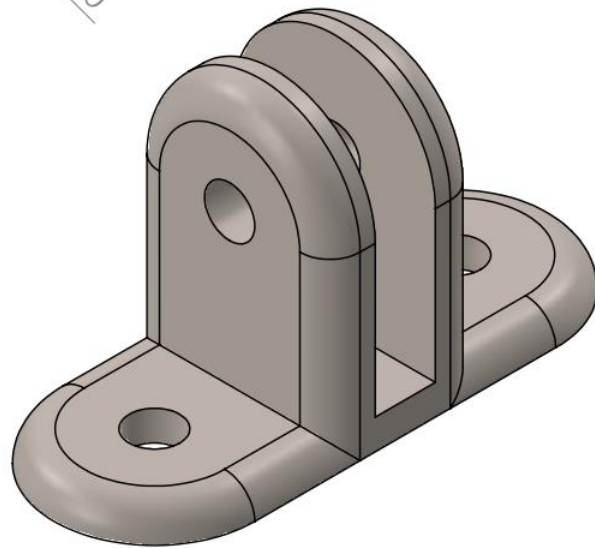
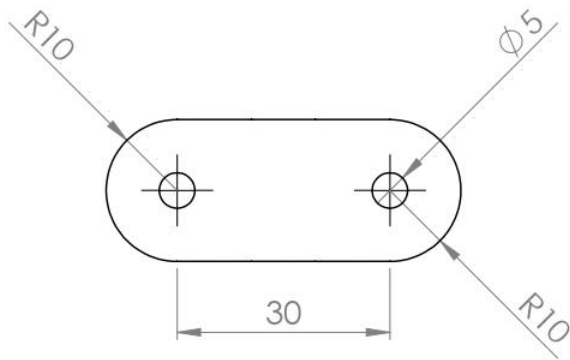
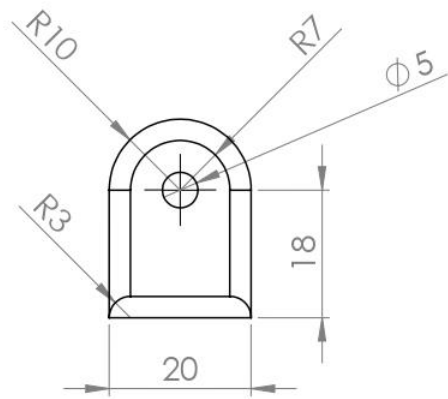
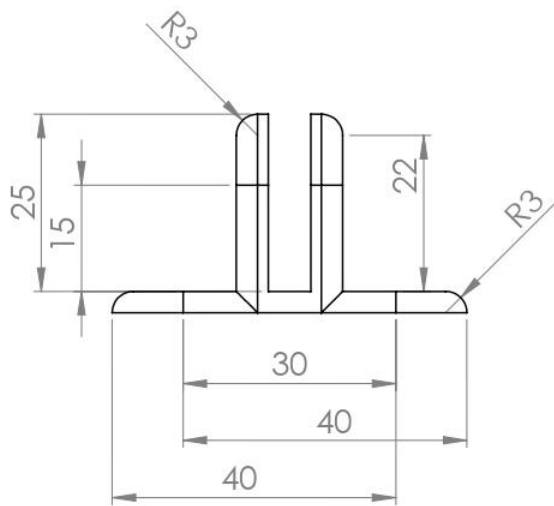
HOJA 1 DE 1

A4

A

A

4 3 2 1



	NOMBRE	FECHA		
DIBUJ.	Tania Quinatoa	Wáimer Morales		
VERIF.	Edwin Altamirano			
APROB.	Edwin Altamirano			
FABR.				
CAUID.				

TÍTULO:

Partes del Rehabilitador

N.º DE DIBUJO

Sujeción de mano

A4

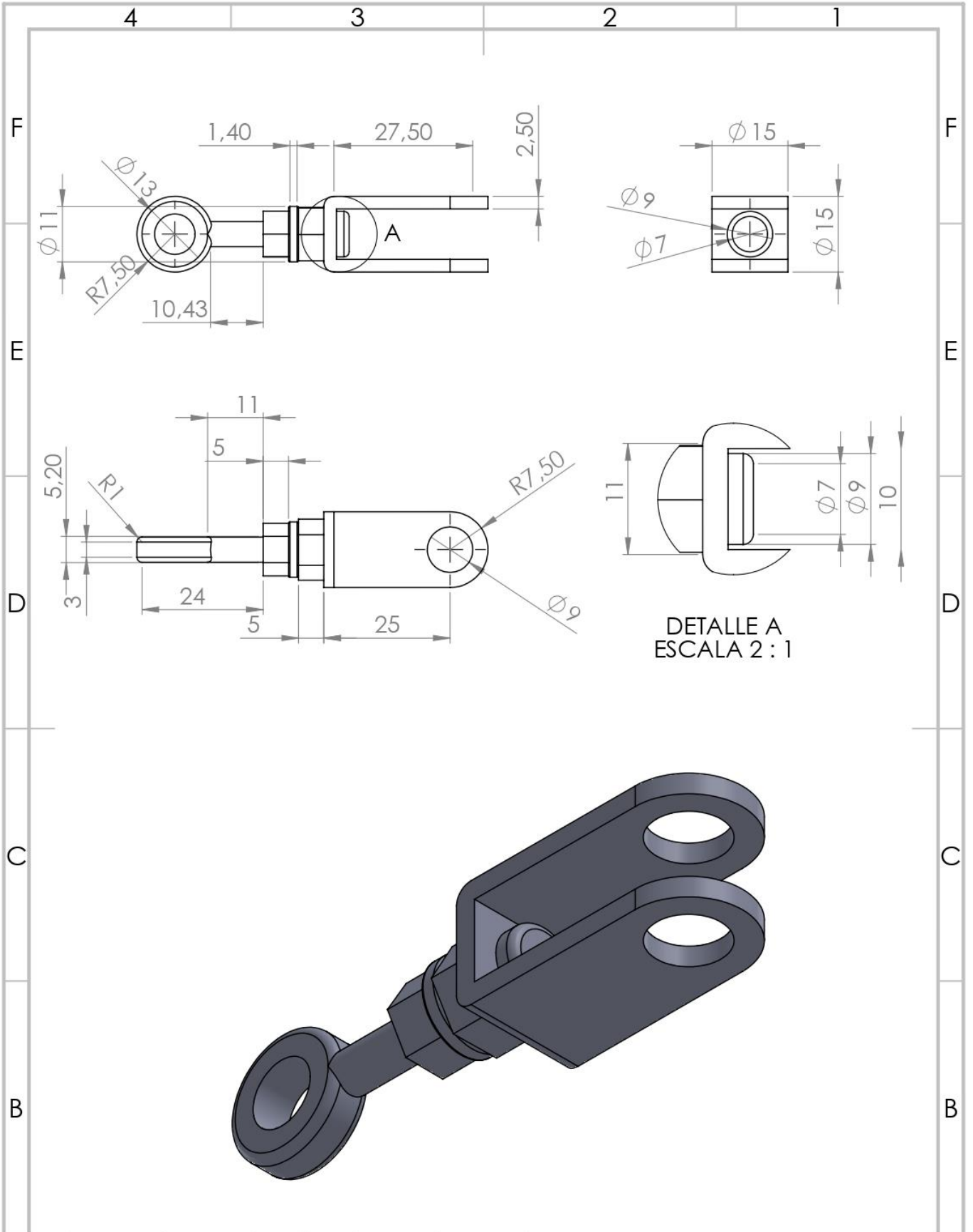
MATERIAL:

Acero Aleado

PESO:

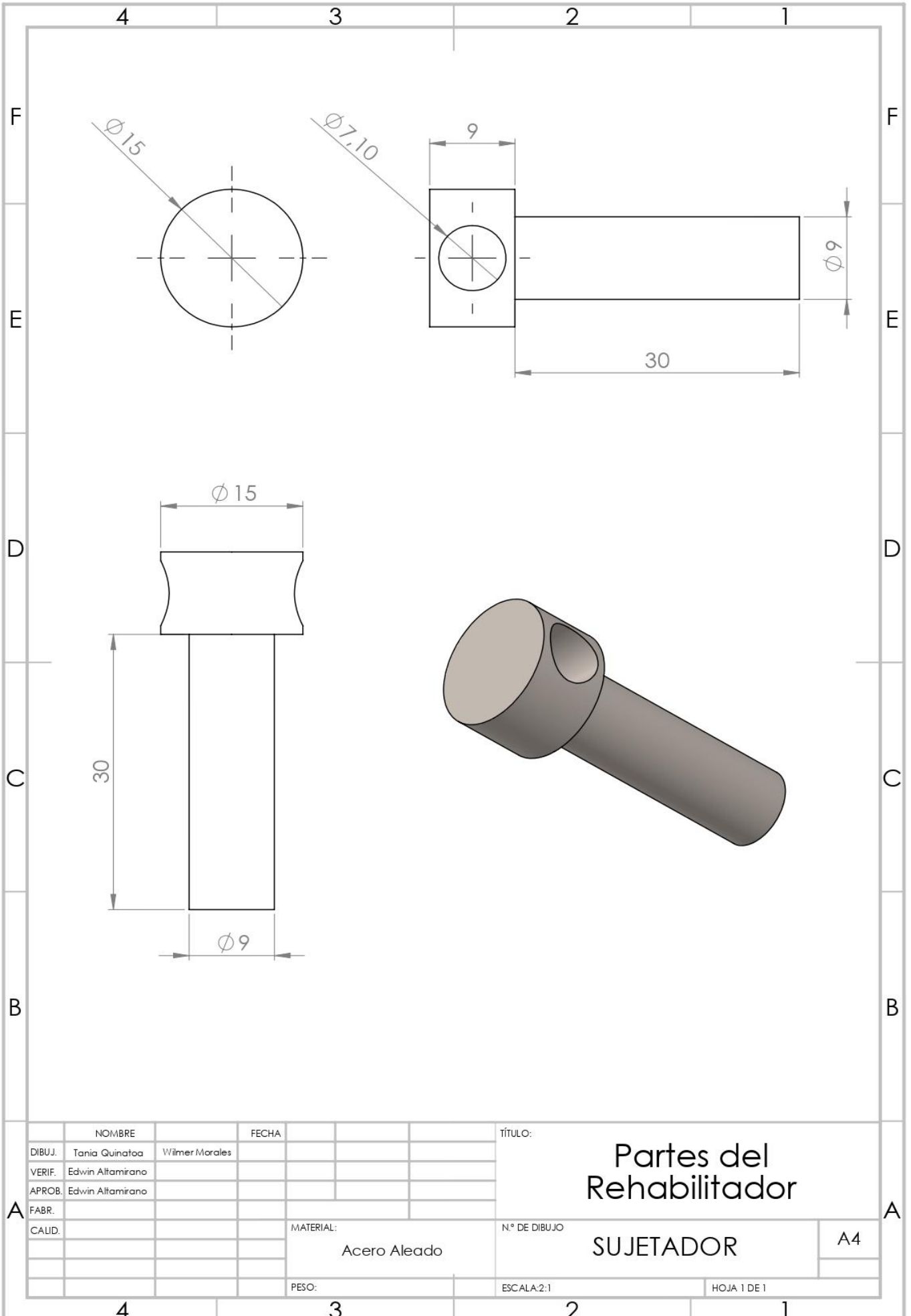
ESCALA: 1:1

HOJA 1 DE 1



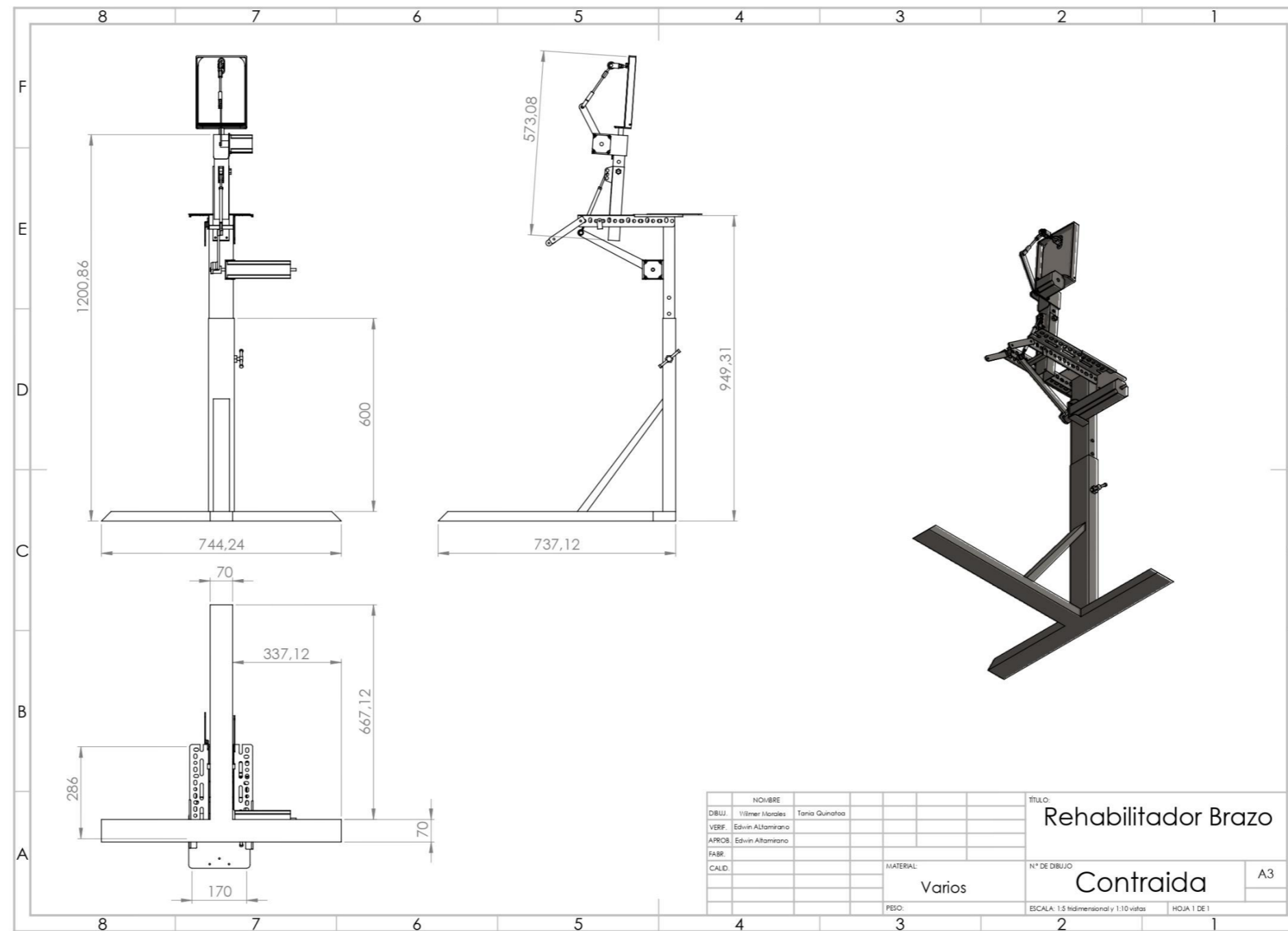
DETALLE A
ESCALA 2 : 1

NOMBRE		FECHA		TÍTULO:	
DIBUJ.	Tania Quinatoa	Wálmer Morales		Partes del Rehabilitador	
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CAUID.			MATERIAL:	N° DE DIBUJO	A4
			Acero Estructural	Sujeción a mano	
			PESO:	ESCALA: 1:1	HOJA 1 DE 1



	NOMBRE	FECHA			
DIBUJ.	Tania Guinatao	Wilmer Morales			
VERIF.	Edwin Altamirano				
APROB.	Edwin Altamirano				
FABR.					
CALID.					

TÍTULO:	Partes del Rehabilitador	
N° DE DIBUJO	SUJETADOR	A4
PESO:	ESCALA: 2:1	HOJA 1 DE 1



ANEXO C: LÁMINA – DIAGRAMA DE CONEXIONES ELÉCTRICAS

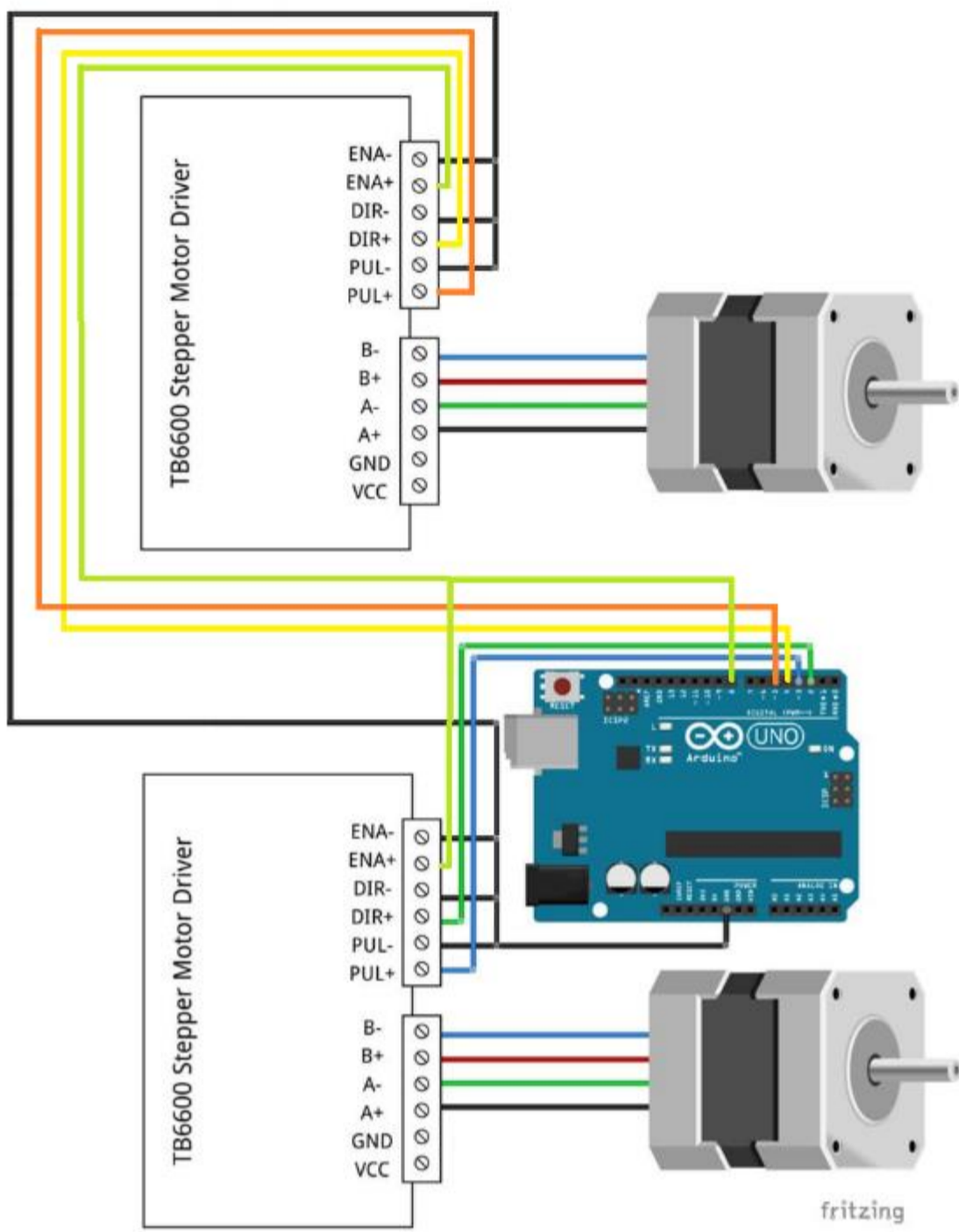


Diagrama de Conexión Arduino - Driver DM542T – NEMA 23

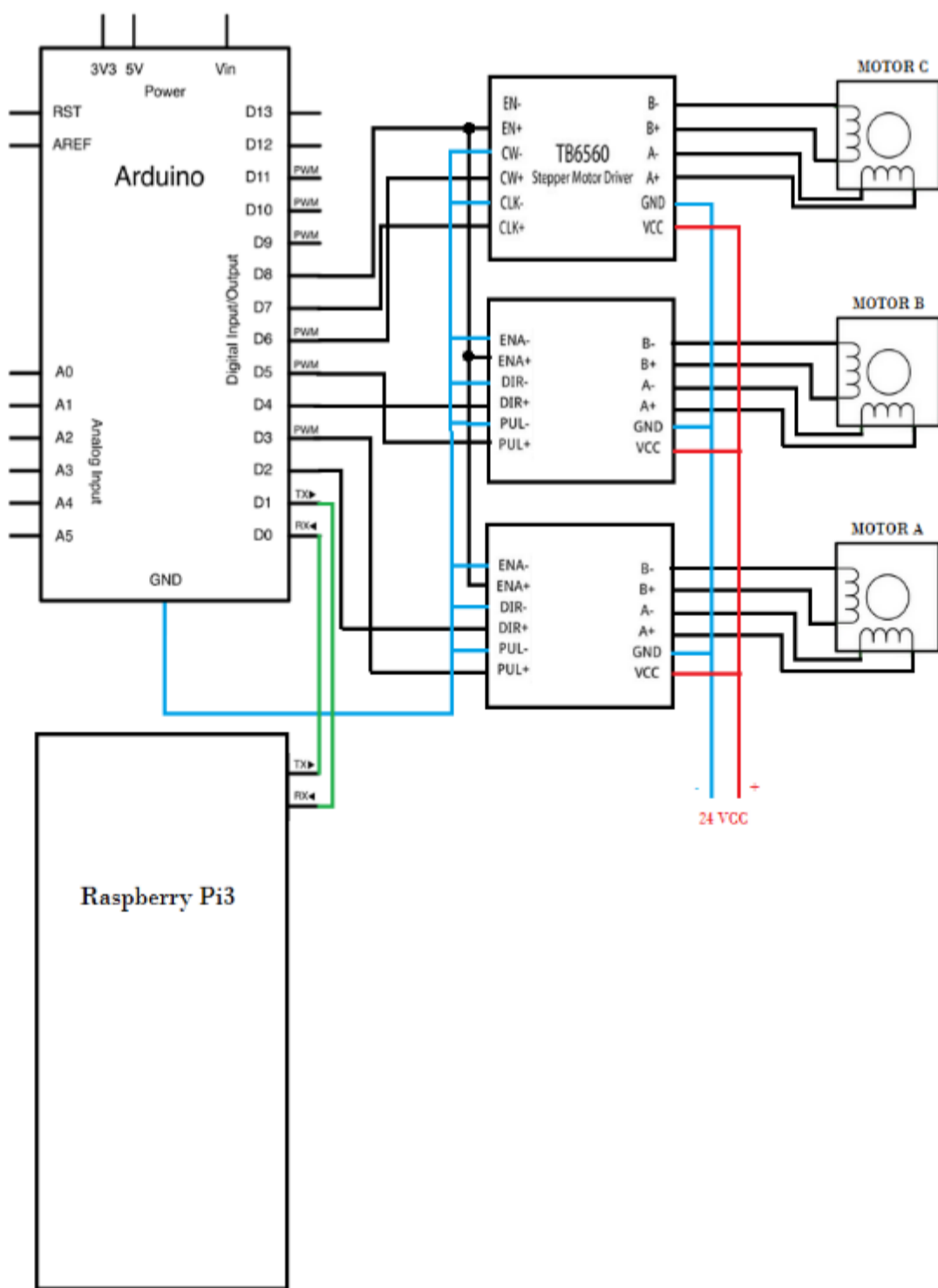


Diagrama eléctrico del prototipo

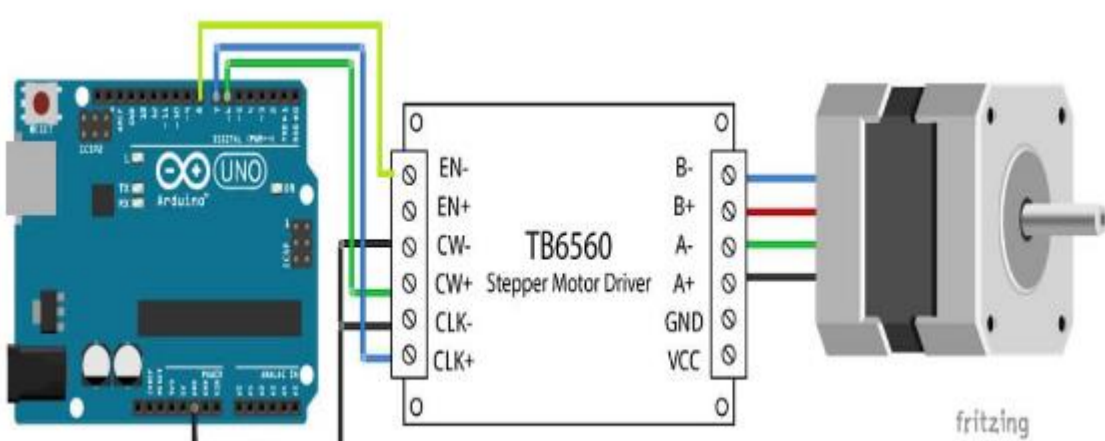


Diagrama de Conexión Arduino - Driver BL-TB6560-V2.0 – NEMA 17

ANEXO D: PROGRAMACIÓN ARDUINO UNO

```
//Variables para almacenamiento de datos
String str1="";
String str2="";
String str3="";
//Variables para comparacion de sentencias
int x=0;
int y=1;
int z=1;

int val=0;

int direc=0;
int giro=0;
int pasos=0;
int repet=0;
int cont=1;

int a=1;
int m=0;
int p=0;
int q=0;

//variables para el envio de angulos
int angulo=0;
String angle="";
String angulo_str="";
String informacion="";
String maniobra="";

//definiendo pines para el control de los motores a
pasos

//Motor A
int PUL_A=3; //Pin para la señal de pulso
int DIR_A=2; //define Direction pin

//Motor B
int PUL_B=5; //Pin para la señal de pulso
int DIR_B=4; //define Direction pin

//Motor C
int PUL_C=7; //Pin para la señal de pulso
int DIR_C=6; //define Direction pin

//
int EN_ABC=8; //define Enable Pin de los 3 Motores
a paso
//
int boton=10; //Pin detener el Motor a paso
int fcarrera=12; //Pin Fin de carrera
//////////Tiempo de calibrado

int t=6000;

//////////Tiempo de Maniobra

//Maniobra A
int t1=6000;

//Maniobra B
int t2=6000;

//Maniobra C
int t3=6000;

//definiendo Entradas/salidas//////////

void setup()
{
  Serial.begin(9600);

  //Motor A
  pinMode(PUL_A, OUTPUT);
  pinMode(DIR_A, OUTPUT);

  //Motor B
  pinMode(PUL_B, OUTPUT);
  pinMode(DIR_B, OUTPUT);

  //Motor C
  pinMode(PUL_C, OUTPUT);
  pinMode(DIR_C, OUTPUT);

  //Motores ABC
  pinMode(EN_ABC, OUTPUT);
  digitalWrite(EN_ABC,LOW);

  //Boton y final de carrera
  pinMode(boton, INPUT);
  pinMode(fcarrera, INPUT);
}

//definiendo funciones ////////////

//ciclo de pulsos (motor a paso)
//////////Calibracion

void pulsos_A()
{
  delay(500);

  if (val==1)
  {
    digitalWrite(DIR_A,LOW);
  }
  if (val==2)
  {
    digitalWrite(DIR_A,HIGH);
  }
  for (int i=0; i<pasos; i++)
  {
    if (digitalRead(boton))
    {
      break;
    }
    if (digitalRead(fcarrera))
    {
      break;
    }
  }
  else
  {
    digitalWrite(PUL_A,HIGH);
    delayMicroseconds(t);
    digitalWrite(PUL_A,LOW);
    delayMicroseconds(t);
  }
}

void pulsos_B()
{
  delay(500);

  if (val==3)
  {
    digitalWrite(DIR_B,LOW);
  }
  if (val==4)
  {
    digitalWrite(DIR_B,HIGH);
  }
  for (int i=0; i<pasos; i++)
  {
    if (digitalRead(boton))
    {
      break;
    }
    if (digitalRead(fcarrera))
    {
      break;
    }
  }
  else
  {
    digitalWrite(PUL_B,HIGH);
    delayMicroseconds(t);
    digitalWrite(PUL_B,LOW);
    delayMicroseconds(t);
  }
}

void pulsos_C()
{
  delay(500);

  if (val==5)
  {
    digitalWrite(DIR_C,LOW);
  }
  if (val==6)
  {
    digitalWrite(DIR_C,HIGH);
  }
  for (int i=0; i<pasos; i++)
  {
    if (digitalRead(boton))
    {
      break;
    }
    if (digitalRead(fcarrera))
    {
      break;
    }
  }
  else
  {
    digitalWrite(PUL_C,HIGH);
    delayMicroseconds(t);
    digitalWrite(PUL_C,LOW);
    delayMicroseconds(t);
  }
}
}
```



```

}
//IMPRESION DE DATOS

void imprimir()
{
    ////////////// Impresion angulo inicial
    if (m==1)
    {
        delay(500);
        angulo =0;
        angulo_str= String (angulo);
        informacion=maniobra + angulo_str+":";
        Serial.println(informacion);
    }

    if (a==89)
    {
        ////////////// conversion de pasos a angulos

        angulo=map(m,0,3200,0,360);

        ////////////// impresion cada cierto paso

        angulo_str = String (angulo);
        informacion=maniobra + angulo_str+":";
        Serial.println(informacion);
        a=1;
    }

    a=a+1;

    //////////////Impresion angulo final

    if (m==pasos)
    {
        delay(500);
        informacion=maniobra + angulo+":";
        Serial.println(informacion);
        a=1;
    }

    //////////////

}

void loop()
{
    ///////////////////////////////////////////////////////////////////
    if (Serial.available())
    {

        char c = Serial.read();

        ///////////////deteccion de separadores
        if ((c=='/') && (y==1))
        {
            z=2;
            x=1;
        }
        if ((c=='/') && (y==2))
        {
            z=4;
            x=2;
        }
        if ((c=='/') && (y==3))
        {
            z=7;
            x=0;
            y=1;
        }

        /////////////// adicion de caracteres
        if(z==1)
        {
            str1 = str1+c;
            direc=c;
        }
        if(z==3)
        {
            str2 = str2+c;

```

ANEXO E: PROGRAMACIÓN PYTHON

```

import serial
import time
##Importar el modulo para la interfaz grafica
import Tkinter as tk
from Tkinter import *
import tkMessageBox
#Importar el modulo para la conexion
import socket
#Importar el modulo de hilos
import threading

serial_Arduino=serial.Serial('/dev/ttyACM0',9600)
serial_Arduino.flushInput()

calibrar=1
maniobra=1
ejecutar=0
datos=""
cont=0
signal=0
env1=0
env2=0
x=0

contador=0

#####

def retroceder_1 ():
    if calibrar==0 :
        if maniobra==1:
            global giro
            giro="1"
            global angulo
            angulo=etiqueta11.get()
            global repeticion
            repeticion="0"
            validar1()
            var11 = StringVar()
            etiqueta11.config(textvariable=var11)
        else:

tkMessageBox.showwarning("Atencion","Bloquear la Maniobra")
    else:
        print ("Dato no enviado a calibracion")

def avanzar_1 ():
    if calibrar==0:
        if maniobra==1:
            global giro
            giro="2"
            global angulo
            angulo=etiqueta11.get()
            global repeticion
            repeticion="0"
            validar1()
            var11 = StringVar()
            etiqueta11.config(textvariable=var11)
        else:

tkMessageBox.showwarning("Atencion","Bloquear la Maniobra")
    else:

```

```

}

if (z==5)
{
    str3 = str3+c;
}

/////////////////salto de lectura

if(x==1)
{
    z=3;
    y=2;
}

if(x==2)
{
    z=5;
    y=3;
}

///////////////// Ejecucion de Movimientos
if(z==7)
{
    val=str1.toInt(); //direccion de str a Int
    angle=str2; //Angulo en str
    giro=str2.toInt(); //Angulo de str a Int
    repet=str3.toInt(); //Numero de repeticiones
para la maniobra de str a int
    pasos=map(giro,0,360,0,3200); //Conversion de
angulos a pasos

    switch(direc)
    {
        case '1':
            pulsos_A();
            break;

        case '2':
            pulsos_A();
            break;

        case '3':
            pulsos_B();
            break;

        case '4':
            pulsos_B();
            break;

        case '5':
            pulsos_C();
            break;

        case '6':
            pulsos_C();
            break;

        case '7':
            pulsos_AA();
            break;

        case '8':
            pulsos_BB();
            break;

        case '9':
            pulsos_CC();
            break;
    }

    //////////////restablece la lectura
    z=1;
    //////////////borra datos almacenados
    str1="";
    str2="";
    str3="";
}
///////////////////////////////////////////////////////////////////
}

```

```

        print ("Dato no enviado a calibracion")

def retroceder_2 ():
    if calibrar==0:
        if maniobra==1:
            global giro
            giro="3"
            global angulo
            angulo=etiqueta22.get()
            global repeticion
            repeticion="0"
            validar1()
            var22 = StringVar()
            etiqueta22.config(textvariable=var22)
        else:

tkMessageBox.showwarning("Atencion","Bloquear la Maniobra")
    else:
        print ("Dato no enviado a calibracion")

def avanzar_2 ():
    if calibrar==0:
        if maniobra==1:
            global giro
            giro="4"
            global angulo
            angulo=etiqueta22.get()
            global repeticion
            repeticion="0"
            validar1()
            var22 = StringVar()
            etiqueta22.config(textvariable=var22)
        else:

tkMessageBox.showwarning("Atencion","Bloquear la Maniobra")
    else:
        print ("Dato no enviado a calibracion")

def retroceder_3 ():
    if calibrar==0:
        if maniobra==1:
            global giro
            giro="5"
            global angulo
            angulo=etiqueta33.get()
            global repeticion
            repeticion="0"
            validar1()
            var33 = StringVar()
            etiqueta33.config(textvariable=var33)
        else:

tkMessageBox.showwarning("Atencion","Bloquear la Maniobra")
    else:
        print ("Dato no enviado a calibracion")

def avanzar_3 ():
    if calibrar==0:
        if maniobra==1:
            global giro
            giro="6"

```

```

        global angulo
        angulo=etiqueta33.get()
        global repeticion
        repeticion="0"
        validar1()
        var33 = StringVar()
        etiqueta33.config(textvariable=var33)
    else:

tkMessageBox.showwarning("Atencion","Bloquear
la Maniobra")
    else:
        print ("Dato no enviado a calibracion")

def ejercitarA():
    if maniobra==0:
        if calibrar==1 :
            global giro
            giro="7"
            global angulo
            angulo=etiqueta101.get()
            global repeticion
            repeticion=etiqueta104.get()
            validar2()
            var101 = StringVar()
            etiqueta101.config(textvariable=var101)
            var104 = StringVar()
            etiqueta104.config(textvariable=var104)
        else:

tkMessageBox.showwarning("Atencion","Bloquear
el Calibrado")
    else:
        print ("Dato no enviado a maniobra")

def ejercitarB():
    if maniobra==0:
        if calibrar==1 :
            global giro
            giro="8"
            global angulo
            angulo=etiqueta102.get()
            global repeticion
            repeticion=etiqueta105.get()
            validar2()
            var102 = StringVar()
            etiqueta102.config(textvariable=var102)
            var105 = StringVar()
            etiqueta105.config(textvariable=var105)
        else:

tkMessageBox.showwarning("Atencion","Bloquear
el Calibrado")
    else:
        print ("Dato no enviado a maniobra")

def ejercitarC():
    if maniobra==0:
        if calibrar==1 :
            global giro
            giro="9"
            global angulo
            angulo=etiqueta103.get()
            global repeticion
            repeticion=etiqueta106.get()
            validar2()
            var103 = StringVar()
            etiqueta103.config(textvariable=var103)
            var106 = StringVar()
            etiqueta106.config(textvariable=var106)
        else:

tkMessageBox.showwarning("Atencion","Bloquear
el Calibrado")
    else:
        print ("Dato no enviado a maniobra")

def validar1():
    global val
    val=angulo.isdigit()
    if val==1:
        limite1()
    else:
        tkMessageBox.showwarning("Cuidado","
Valores Incorrectos : \n Ingrese angulos \n
validos ")

def validar2():
    global val
    val=angulo.isdigit()
    global val2
    val2=repeticion.isdigit()
    if val==1 and val2==1:
        limite2()
    else:
        tkMessageBox.showwarning("Cuidado","
Valores Incorrectos : \nIngrese
angulos/repeticiones\n validos ")

def limite1():
    global angle
    angle= int(angulo)
    if angle>=0 and angle<=25:
        mover1()
    else:
        tkMessageBox.showwarning("ANGULO","
Ingrese valores \n del 0 al 25 ")

def limite2():
    global ejecutar
    global angle
    angle= int(angulo)
    global repeticion
    repeticion= int(repeticion)
    if angle>=0 and angle<=100:
        if repeticion>=1 and repeticion<=5:
            if ejecutar==0:
                ejecutar=1
                mover2()
            else:

tkMessageBox.showwarning("Atencion","Ya ejecuto
una maniobra espere")
    else:

tkMessageBox.showwarning("REPETICION","
Ingrese valores \n del 1 al 5 ")
    else:
        tkMessageBox.showwarning("ANGULO","
Ingrese valores \n del 0 al 100 ")

def mover1():
    global cadena
    cadena=giro+"/"+angulo+"/"+repeticion+"/"
    print (cadena)
    comando_utf = cadena.encode()
    serial_Arduino.write(comando_utf)
    time.sleep(0.1)

def mover2():
    global x

x=1
global cadena
cadena=giro+"/"+angulo+"/"+repeticion+"/"
print (cadena)
comando_utf = cadena.encode()
serial_Arduino.write(comando_utf)
time.sleep(0.1)
serial_Arduino.flushInput()
emitir2= threading.Thread(name="hilo_3",
target=enviar2)
emitir2.start()

def enviar2():
    global env2
    env2 = 0
    global env1
    global datos
    global contador
    global informacion
    global ejecutar
    while(env2==0):
        try:
            print(contador)
            contador=contador+1
            time.sleep(0.1)
            while (serial_Arduino.inWaiting(>0):
                contador=0
                print(contador)
                informacion= serial_Arduino.readline()
                dt=informacion.rstrip('\n')
                dt_utf = dt.decode('utf-8')
                datos=str(dt_utf)
                print (datos)
                etiqueta70.config(text=datos)
                envio()
            if contador==50:
                env2=1
                ejecutar=0
                contador=0
                time.sleep(0.1)
                #llamando reenvio
                etiqueta70.config(text="Datos")
                reenviar()
        except:
            print("No hubo datos")

def envio():
    try:
        socket_s.send(datos)
    except:
        etiqueta50.config(text="Desconectado")
        print ("Dato No Enviado")

def reenviar():
    global env1
    env1=0
    global x
    x=0
    print ("Abrir envio")
    #Hilo de envio de datos(segundo plano)
    emitir = threading.Thread(name="hilo_2",
target=enviar1)
    emitir.start()

def password():
    if etiqueta44.get()=='internet':
        call_me()
    else:

tkMessageBox.showwarning("Cuidado","Password
Incorrecto")

def call_me():
    answer=tkMessageBox.askquestion("Conexion","
Desea continuar, presione 'Si' ")
    if answer=='yes':
        var44 = StringVar()
        etiqueta44.config(textvariable=var44)
        #hilo de conexion
        Conectar= threading.Thread(name="hilo_1",
target=accept)
        Conectar.start()
    else:
        var44 = StringVar()
        etiqueta44.config(textvariable=var44)

#Conexion: Hilo1 (Programacion paralela)
def accept():
    global cont
    global signal
    global env1
    if cont==0:
        cont=1
        signal=1
        env1=0
        global socket_s
        socket_s = socket.socket()
        global host
        host = "
        global port
        port = 9999
        global backlog
        backlog = 5
        socket_s.bind ((host,port))
        socket_s.listen(backlog)
        print ("ESPERANDO UNA CONEXION ... :)")
        socket_s, (host,port) = socket_s.accept()
        #Hilo de envio de datos(segundo plano)
        emitir = threading.Thread(name="hilo_2",
target=enviar1)
        emitir.start()
        print ("CONEXION ESTABLECIDA ..... :)")
        etiqueta50.config(text="Conectado")
    else :

tkMessageBox.showwarning("Cuidado","Conexion
Realizada")

def enviar1():
    global env1
    global informacion
    global cont
    global x
    while (env1==0):
        try:
            informacion="LEYENDO"
            socket_s.send(informacion)
            print('Dato enviado:LEYENDO')
            time.sleep(1)
            if (x==1):
                env1=1
        except:
            time.sleep(1)
            print('Dato no enviado:LEYENDO')
            etiqueta50.config(text="Desconectado")
            env1=1
            cont=0

def Habilitar1():
    global calibrar

```

```

if calibrar==1 :
answer=tkMessageBox.askquestion("Desbloquear","
Desea continuar, presione 'Si' ")
if answer=='yes':
etiqueta5.config(text="Desbloqueado")
etiqueta5.config(bg="light cyan")
BT_Habilitar1.config(text="Bloquear")
BT_Habilitar1.config(bg='Orchid1')
calibrar=0
else:
answer=tkMessageBox.askquestion("Bloquear","
Desea continuar, presione 'Si' ")
if answer=='yes':
etiqueta5.config(text="Bloqueado")
etiqueta5.config(bg="thistle1")
BT_Habilitar1.config(text="Desbloquear")
BT_Habilitar1.config(bg='cyan2')
calibrar=1

def Habilitar2():
global maniobra
if maniobra==1 :
answer=tkMessageBox.askquestion("Desbloquear","
Desea continuar, presione 'Si' ")
if answer=='yes':
etiqueta6.config(text="Desbloqueado")
etiqueta6.config(bg="light cyan")
BT_Habilitar2.config(text="Bloquear")
BT_Habilitar2.config(bg='Orchid1')
maniobra=0
else:
answer=tkMessageBox.askquestion("Bloquear","
Desea continuar, presione 'Si' ")
if answer=='yes':
etiqueta6.config(text="Bloqueado")
etiqueta6.config(bg="thistle1")
BT_Habilitar2.config(text="Desbloquear")
BT_Habilitar2.config(bg='cyan2')
maniobra=1

#####
#####

root = Tk()
root.title("Interfaz")
root.minsize(1280,600)
root.resizable(width=False, height=False)
#Color de fondo pantalla
#root.config(bg="aquamarine3")

img = PhotoImage(file="FONDO.png")
widget = Label(root, image=img)
widget.place(x=0,y=0,relwidth=1.0,relheight=1.0)

#####Imágenes de
apoyo

#####Horizontal

img2 = PhotoImage(file="x1.png")
img2 = img2.subsample(1,1)
widget2 = Label(root, image=img2)
widget2.grid(column=2, row=0)

img3 = PhotoImage(file="x2.png")
img3 = img3.subsample(1,1)
widget3 = Label(root, image=img3,bg="snow2")
widget3.grid(column=5, row=0)

img4 = PhotoImage(file="x3.png")
img4 = img4.subsample(1,1)
widget4 = Label(root, image=img4,bg="snow2")
widget4.grid(column=7, row=0)

img5 = PhotoImage(file="x4.png")
img5 = img5.subsample(1,1)
widget5 = Label(root, image=img5,bg="snow2")
widget5.grid(column=9, row=0)

img6 = PhotoImage(file="x5.png")
img6 = img6.subsample(1,1)
widget6 = Label(root, image=img6,bg="snow2")
widget6.grid(column=11, row=0)

img7 = PhotoImage(file="x6.png")
img7 = img7.subsample(1,1)
widget7 = Label(root, image=img7,bg="snow2")
widget7.grid(column=13, row=1)

#####Vertical
img8 = PhotoImage(file="y1.png")
img8 = img8.subsample(1,1)
widget8 = Label(root, image=img8,bg="gray68")
widget8.grid(column=0, row=0)

img9 = PhotoImage(file="y2.png")
img9 = img9.subsample(1,1)
widget9 = Label(root, image=img9,bg="gray74")
widget9.grid(column=0, row=2)

img10 = PhotoImage(file="y3.png")
img10 = img10.subsample(1,1)
widget10 = Label(root, image=img10,bg="gray70")
widget10.grid(column=0, row=5)

img11 = PhotoImage(file="y4.png")
img11 = img11.subsample(1,1)
widget11 = Label(root, image=img11,bg="gray65")
widget11.grid(column=0, row=7)

img12 = PhotoImage(file="y5.png")
img12 = img12.subsample(1,1)
widget12 = Label(root, image=img12,bg="gray59")
widget12.grid(column=0, row=9)

img13 = PhotoImage(file="y6.png")
img13 = img13.subsample(1,1)
widget13 = Label(root, image=img13,bg="gray52")
widget13.grid(column=0, row=11)

#####TITULO1#####
#####

etiqueta01 = Label(root, text="Calibrar",width="14",
relief=GROOVE,bg="goldenrod1", pady=5)
etiqueta01.grid(column=1, row=4)

##### Motor A
var1 = StringVar()
etiqueta1 = Label(root, textvariable=var1,width="14",
relief=SOLID,bg="bisque", pady=5)
var1.set("Motor A")
etiqueta1.grid(column=1, row=6)

var11 = StringVar()
etiqueta11 = Entry(root, textvariable=var11
,width="10")
etiqueta11.grid(column=6, row=6)
#####

##### Motor B
var2 = StringVar()

```

```

etiqueta2 = Label(root, textvariable=var2,width="14",
relief=SOLID,bg="bisque", pady=5)
var2.set("Motor B")
etiqueta2.grid(column=1, row=8)

var22 = StringVar()
etiqueta22 = Entry(root, textvariable=var22
,width="10")
etiqueta22.grid(column=6, row=8)
#####

##### Motor C
var3 = StringVar()
etiqueta3 = Label(root, textvariable=var3,width="14",
relief=SOLID,bg="bisque", pady=5)
var3.set("Motor C")
etiqueta3.grid(column=1, row=10)

var33 = StringVar()
etiqueta33 = Entry(root, textvariable=var33
,width="10")
etiqueta33.grid(column=6, row=10)
#####

#####Ingreso de
Password

var4 = StringVar()
etiqueta4 = Label(root, textvariable=var4,
width="18", relief=SOLID,bg="bisque", pady=5)
var4.set("Password")
etiqueta4.grid(column=10, row=1)

var44 = StringVar()
etiqueta44 = Entry(root, textvariable=var44
,width="13",show="*")
etiqueta44.grid(column=12, row=1)

#####Etiqueta
Bloqueo Calibrado

etiqueta5 = Label(root,
text="Bloqueado",width="13",
relief=FLAT,bg="thistle1", pady=5)
etiqueta5.grid(column=4, row=4)

#####Etiqueta
Bloqueo Maniobra

etiqueta6 = Label(root,
text="Bloqueado",width="13",
relief=FLAT,bg="thistle1", pady=5)
etiqueta6.grid(column=12, row=3)

#####Etiqueta
Angulo de calibrado

etiqueta7 = Label(root, text="Angulo",width="10",
relief=SOLID,bg="khaki1", pady=5)
etiqueta7.grid(column=6, row=4)

#####Etiqueta
Angulo de Maniobra

etiqueta8 = Label(root, text="Angulo",width="13",
relief=SOLID,bg="khaki1", pady=5)
etiqueta8.grid(column=12, row=4)

#####TITULO2#####
#####

etiqueta02 = Label(root,
text="Maniobra",width="14",
relief=GROOVE,bg="goldenrod1", pady=5)
etiqueta02.grid(column=8, row=3)

#Maniobra Motor A

var10 = StringVar()
etiqueta10 = Label(root,
textvariable=var10,width="14",
relief=SOLID,bg="bisque", pady=5)
var10.set("Maniobra A")
etiqueta10.grid(column=8, row=6)

var101 = StringVar()
etiqueta101 = Entry(root, textvariable=var101
,width="13")
etiqueta101.grid(column=12, row=6)

#Maniobra Motor B

var20 = StringVar()
etiqueta20 = Label(root,
textvariable=var20,width="14",
relief=SOLID,bg="bisque", pady=5)
var20.set("Maniobra B")
etiqueta20.grid(column=8, row=8)

var102 = StringVar()
etiqueta102 = Entry(root, textvariable=var102
,width="13")
etiqueta102.grid(column=12, row=8)

#Maniobra Motor C

var30 = StringVar()
etiqueta30 = Label(root,
textvariable=var30,width="14",
relief=SOLID,bg="bisque", pady=5)
var30.set("Maniobra C")
etiqueta30.grid(column=8, row=10)

var103 = StringVar()
etiqueta103 = Entry(root, textvariable=var103
,width="13")
etiqueta103.grid(column=12, row=10)

##### Numero
de repeticiones para la maniobra
#####TITULO 3 #####

var40 = StringVar()
etiqueta40 = Label(root,
textvariable=var40,width="13",
relief=SOLID,bg="khaki1", pady=5)
var40.set("Repeticiones")
etiqueta40.grid(column=14, row=4)

#Maniobra A
var104 = StringVar()
etiqueta104 = Entry(root, textvariable=var104
,width="13")
etiqueta104.grid(column=14, row=6)

#Maniobra B
var105 = StringVar()
etiqueta105 = Entry(root, textvariable=var105
,width="13")
etiqueta105.grid(column=14, row=8)

#Maniobra C
var106 = StringVar()
etiqueta106 = Entry(root, textvariable=var106
,width="13")
etiqueta106.grid(column=14, row=10)

```

```

#####Etiqueta
de aviso de conexion

etiqueta50 = Label(root,
text="Desconectado",width="13",
relief=FLAT,bg="lightPink1", pady=5)
etiqueta50.grid(column=14, row=1)

#####Etiqueta
de Muestreo de datos

etiqueta60 = Label(root,
text="Informacion",width="14",
relief=SOLID,bg="bisque", pady=5)
etiqueta60.grid(column=8, row=4)

etiqueta70 = Label(root, text="Datos",width="18",
relief=SOLID,bg="light yellow", pady=5)
etiqueta70.grid(column=10, row=4)

## Botones para el calibrado##### Motor A

#boton Retroceder
MA_retroceder = Button(root,
text=" Retroceder ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="10",
command=retroceder_1)
MA_retroceder.grid(column=4,row=6)
#boton Avanzar
MA_avanzar = Button(root,
text=" Avanzar ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="10",
command=avanzar_1)
MA_avanzar.grid(column=3,row=6)

##### Motor B
#boton Retroceder
MB_retroceder = Button(root,
text=" Retroceder ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="10",
command=retroceder_2)
MB_retroceder.grid(column=4,row=8)
#boton Avanzar
MB_avanzar = Button(root,
text=" Avanzar ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="10",
command=avanzar_2)
MB_avanzar.grid(column=3,row=8)
##### Motror C
#boton Retroceder
MC_retroceder = Button(root,
text=" Retroceder ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="10",
command=retroceder_3)
MC_retroceder.grid(column=4,row=10)
#boton Avanzar
MC_avanzar = Button(root,
text=" Avanzar ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="10",
command=avanzar_3)
MC_avanzar.grid(column=3,row=10)

#####boton Conectar
Bconec = Button(root,
text=" Conectar ",
bg='turquoise',
activebackground='OliveDrab1',
relief=GROOVE,
width="12",
command=password)
Bconec.grid(column=8,row=1)

#boton Bloqueo/Desbloqueo Calibrado
BT_Habilitar1 = Button(root,
text=" Desbloquear ",
bg='cyan2',
activebackground='OliveDrab1',
relief=GROOVE,
width="10",
command=Habilitar1)
BT_Habilitar1.grid(column=3,row=4)

#boton Bloqueo/Desbloqueo Maniobra
BT_Habilitar2 = Button(root,
text=" Desbloquear ",
bg='cyan2',
activebackground='OliveDrab1',
relief=GROOVE,
width="15",
command=Habilitar2)
BT_Habilitar2.grid(column=10,row=3)

#####boton Ejercitar Maniobra A
BT_EjercitarA = Button(root,
text=" Ejercitar ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="15",
command=ejercitarA)
BT_EjercitarA.grid(column=10,row=6)

#boton Ejercitar Maniobra B
BT_EjercitarB = Button(root,
text=" Ejercitar ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="15",
command=ejercitarB)
BT_EjercitarB.grid(column=10,row=8)

#boton Ejercitar Maniobra C
BT_EjercitarC = Button(root,
text=" Ejercitar ",
bg='DarkSeaGreen1',
activebackground='green yellow',
relief=GROOVE,
width="15",
command=ejercitarC)
BT_EjercitarC.grid(column=10,row=10)

root.mainloop()

env1=1
env2=1
if signal==1:
print ("DESCONECTANDO.....")
global socket_s
socket_s.close()
print ("DESCONECTADO")

serial_Arduino.close()

print ("Proceso Finalizado")

```





ESCUELA SUPERIOR POLITÉCNICA DE
CHIMBORAZO
DIRECCIÓN DE BIBLIOTECAS Y RECURSOS
PARA EL APRENDIZAJE Y LA INVESTIGACIÓN



UNIDAD DE PROCESOS TÉCNICOS
REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 30 / 11 / 2020

INFORMACIÓN DEL AUTOR/A (S)
Nombres – Apellidos: Tania Cristina Quinatoa Chicaiza Wilmer Luis Morales Rivera
INFORMACIÓN INSTITUCIONAL
Facultad: Informática y Electrónica
Carrera: Ingeniería Electrónica en Control y Redes Industriales
Título a optar: Ingeniero en Electrónica, Control y Redes Industriales
f. Analista de Biblioteca responsable: Ing. CPA. Jhonatan Rodrigo Parreño Uquillas. MBA.
 <small>Verificar autenticidad por: JHONATAN RODRIGO PARRENO UQUILLAS</small>
 30-11-2020 0307-DBRAI-UPT-2020