**UNIVERSIDAD DE TARAPACÁ**



**ESCUELA UNIVERSITARIA DE INGENIERÍA INDUSTRIAL, INFORMÁTICA Y DE SISTEMAS**



Área de Ingeniería en Computación e Informática



**Avance Proyecto V2  
DANTRON**

Autor(es):

Angelo Coriza

David Orellana

Nicolas Vargas

Asignatura: Proyecto 1

Profesor(es): Diego Aracena

Ricardo Valdivia

ARICA,22 de Noviembre del 2018

# Historial de Cambios

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor(es)** |
| 16/08/2018 | 1.0 | Creación y formulación de  proyecto | Angelo Coriza  David Orellana  Nicolás Vargas |
| 16/08/2018 | 1.1 | Complementación de ideas existentes , se añadió nuevas tareas y se asignaron nuevos roles debido a la cantidad limitada de integrantes | Nicolás Vargas |
| 06/09/2018 | 1.2 | Estudio de posibles riesgos que puedan existir a lo largo del proyecto , soluciones de estos , completación de contenidos faltantes | Nicolás Vargas |
| 15/11/2018 | 1.3 | Arquitectura de software propuestas y requerimiento | Angelo Coriza  David Orellana  Nicolás Vargas |
| 21/11/2018 | 1.4 | Requerimiento y diseño de interfaz | Angelo Coriza  David Orellana  Nicolás Vargas |
| 22/11/2018 | 1.5 | implementación, estado actual del proyecto y anexo de código | Angelo Coriza  David Orellana  Nicolás Vargas |

# Tabla de Contenidos

1. **Panorama General**
   1. Introducción
   2. Objetivo General
   3. Objetivos Específicos
   4. Restricciones:
   5. Entregables:
2. **Organización del Personal**

2.1. Descripción de Roles

2.2. Personal que cumplirá los Roles

2.3. Mecanismos de Comunicación

1. **Planificación del Proyecto**

3.1. Actividades:

3.2. Asignación de tiempo

3.3. Personal-rol asignado

3.3. Gestión de Riesgos:

1. **Planificación de los Recursos**

4.1. Recursos Hardware-Software requeridos

4.2. Estimación de Costos (Hardware, Software, Recursos Humanos)

1. **Análisis-diseño**5.1 Especificación de Requerimientos (incluyendo método/algoritmos considerados para resolver el cubo Rubik)  
   5.2 Arquitectura Propuesta (incluyendo aspectos de comunicación)  
   5.3 Diseño de la Interfaz Usuario
2. **Implementación**6.1 Descripción de los programas implementados (entradas, salidas, procesos)  
   6.2 Diagrama de interacción entre programas
3. **resultados**7.1 Estado actual del proyecto  
   7.2 Problemas encontrados y soluciones propuestas  
   7.3 Conclusiones
4. **Referencias (estándar IEEE)**
   1. **Introducción**

Lego Mindstorms es la tercera generación de bloques EV3 la cual posee de ciertos puertos de comunicación RJ12 los cuales nos permiten realizar comunicación entre EV3 y el dispositivo ya sea sensor o motor ,para así lograr una tarea determinada , con este modelo de lego se pretenderá ayudar a una persona a resolver el famoso cubo de Rubik de 3x3 el cual consta de 5 colores distintos

La finalidad de este informe es dar a conocer el proyecto “Dantron ” que consiste en el armado de un Robot Mindstorms de Lego el cual mediante unos algoritmos codificados en lenguaje de programación Python y escrito en el IDE (Entorno de Desarrollo Integrado ) Visual Studio Code será capaz de ayudar a una persona a armar un cubo de Rubik’s (mediante un algoritmo escogido por los miembros del equipo) en un lapso de tiempo determinado adicionalmente en el presente informe se detallaran los miembros del equipo a los cuales se les ha asignado una/s tareas específicas para contribuir con el desarrollo del proyecto y este sea todo un éxito , así mismo se informará de los recursos y gastos necesarios para realizar el proyecto , analizar los posibles riesgos que puedan existir mediante el transcurso de su desarrollo y las posibles soluciones para poder resolverlos de manera adecuada y sin detener el avance de dicho proyecto .



* 1. Objetivo General
     1. Armar un robot Lego Mindstorms que puede ejecutar algoritmos de movimiento para ayudar al armado de un cubo de Rubik de 3x3x3.
  2. Objetivos Específicos
     1. Seguir instrucciones de armado de robot Lego Mindstorms EV3
     2. Lograr codificar algoritmos de movimientos a lenguaje de programación Python
     3. Lograr una comunicación exitosa entre el robot y el dispositivo (PC, Teléfono celular) mediante Bluetooth para enviar algoritmos de movimiento los cuales debieran ayudar al armado del cubo de Rubik’s .
  3. **Restricciones:**

Para el exitoso desarrollo del proyecto como restricciones tenemos los siguientes puntos a considerar.

* **Tiempo**: El tiempo es un factor clave para el proyecto ya que es el tiempo en el cual el proyecto se debe realizar sí o sí , ya que no disponemos de ningún mes , día o semana de más debido al fin del semestre académico.
* **Cantidad de Integrantes**: Otra restricción ya que algunos miembros del equipo deberán realizar más de alguna tarea debido que solo somos 3 integrantes en comparación a otros equipos de la asignatura.
* **Lego Mindstorms:** Kit del robot que se nos ha dado para trabajar (Lego Mindstorms) no tenemos otra opción para trabajar.
* **Programación:** Para codificar los algoritmos del cubo Rubik sólo podemos utilizar el lenguaje de programación Python por lo cual también nos vemos limitado.
* **Sistema Operativo:** Solo podemos elegir entre las 2 imágenes ISO entregadas por el profesor en clases EV3D.



* 1. **Entregables**
* **Bitácoras:** Al finalizar la clase del día jueves se deberá subir una Bitácora con los avances realizados en la semana, los problemas surgidos las posibles soluciones a estos y la planificación de las actividades a realizar en la semana siguiente.
* **Presentaciones:** Pueden ser realizadas en formato Power Point, estás presentaciones sirven para promocionar el proyecto realizado indicando detalles, costes, avances, resolución de problemas surgidos, etc.
* **Producto Final:** EL producto final será entregado con un manual de usuario en el cual se detalla cómo funciona el producto, sus características su funcionamiento y una pequeña demostración que garantice su funcionamiento.
* **Wiki:** Promoción del proyecto en él se detallan avance actual del proyecto, presupuesto , fotografías del proceso , etc.



**2.1. Descripción de Roles**

* **Líder del equipo:** Encargado de verificar el cumplimiento de los avances del proyecto en las fechas establecidas , velar por el trabajo en equipo , generar buenas relaciones entre los miembros del equipo .
* **Programador:** Persona que se dedicara previamente a estudiar el lenguaje de programación Python para posteriormente codificar algoritmos de armado de cubo Rubik a dicho lenguaje de programación.
* **Armador:** Persona encargada de arma el robot, conseguir piezas faltantes (en caso de de no estén dichas piezas) si están piezas no logran ser conseguidas se tratará de adaptar un mecanismo con las piezas que ya poseemos.
* **Secretario:** Persona encargada de subir las Bitácoras y/o avances del proyecto.
* **Wiki :** Persona encargada de subir fotografías con avances para promocionar el proyecto que se está realizando .

**2.2. Personal que cumplirá los Roles**



* **Líder** : Nicolás Vargas
* **Programador:** Angelo Coriza / Nicolás Vargas
* **Armador:** Nicolás Vargas /David Orellana / Angelo Coriza
* **Secretario :** David Orellana / Angelo Coriza / Nicolás Vargas
* **Wiki :** David Orellana

**2.3. Mecanismos de Comunicación**

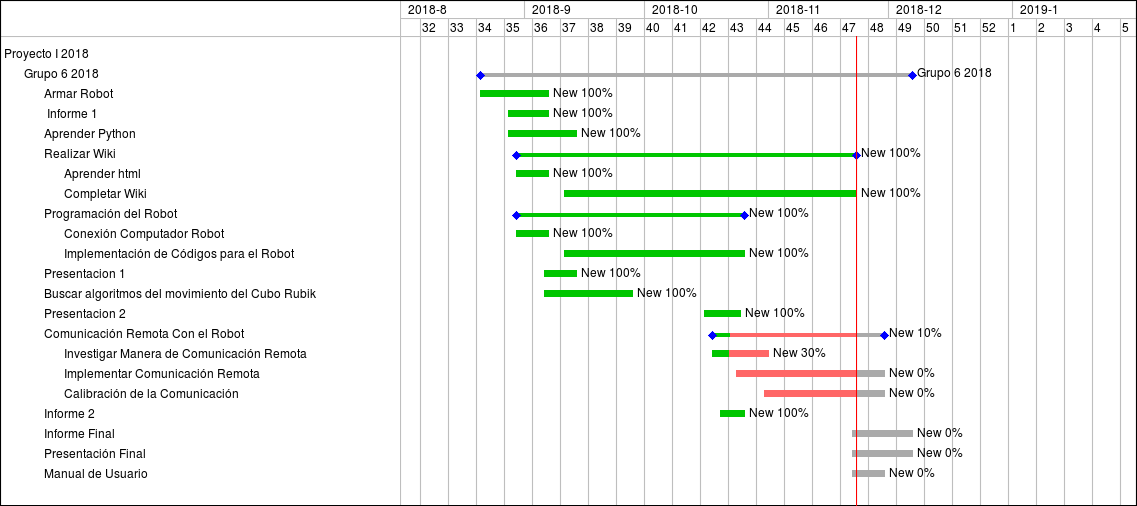
El mecanismo de comunicación entre los miembros del equipo es en el aula de clases en las cuales se conversara de avances, problemas surgidos, resolución de estos este medio de comunicación es bastante limitado adicionalmente hemos creado Google Drive (para realizar más rápido los entregables del proyecto ) poseemos grupo de WhatsApp y grupo en Facebook ante cualquier problema que pudiera surgir en determinado momento. Otro medio es REDMINE en el cual subimos los avances del proyecto y tenemos acceso a nuestra carta Gantt.

**Planificación del Proyecto**

**3.1. Actividades**: Dentro del proyecto de momento tenemos contempladas las siguientes actividades :

* **Armado del robot:** Construir el Robot a partir de un plano modificado  
   **Responsables:**-Nicolas Vargas  
   **Seguidores:** -Angelo Coriza  
  **Producto:** Armado completo del Robot mediante las instrucciones de Lego
* **Aprender Python:** Aprender lenguaje Python para la codificación de algoritmos de movimiento del robot   
   **Responsable:** -Nicolas Vargas  
   **Producto:** Conocimiento del lenguaje de programación Python
* **Realizar Wiki:** Crear y actualizar una wiki en la plataforma Redmine   
   **1. Aprender HTML:** Aprender lenguaje HTML para la creación de la wiki  
   **2. Completar wiki:** Actualizar la wiki  
   **Responsable:**-David Orellana  
   **Seguidores:**---  
   **Producto:** Wiki en la cual se promociona y se muestran avances realizados en el proyecto.
* **Programación del robot :** Programar mediante el lenguaje Python poder mover el robot.  
   **1. Conexión computadora robot: C**onexión vía SSH del Brick con el PC para poder enviar los archivos ejecutables los cuales permitirán que el Robot realice ciertos movimientos los cuales ayudaran a resolver un cubo de Rubik’s.  
   **2. Probar código del robot:** Se probarán los códigos o ajustarán para que realicen la tarea predeterminada de manera óptima.   
   **Responsable:** - Nicolas Vargas  
   P**roducto**: Código del robot
* **Informe 1:** Crear informe de planificación del proyecto **Responsable:**- Nicolas Vargas  
   **Seguidores:** Angelo Coriza y David Orellana **Producto:** Planificación del proyecto en formato Word.
* **Presentación 1:** Crear presentación de la planificación del proyecto **Responsable:** Nicolas Vargas  
   **Seguidores:** -Angelo Coriza, -David Orellana  
   **Producto:** Planificación del proyecto en formato Power Point
* **Buscar algoritmos del movimiento del Cubo:** Hallar algoritmos de movimiento de cubo Rubik’s de 3x3 y adaptarlo al Lenguaje de Programación Python.  
   **Responsable:**-David Orellana  
   **Producto:** Conocimiento de algoritmos y su codificación
* **Comunicación Remota Con el Robot:** Buscar e implementar comunicación remota con  
   el robot **1. Investigar Manera de Comunicación Remota:** Investigar múltiples formas de conectarse remotamente con el robot **2. Implementar Comunicación Remota:** Implementar una forma de comunicación previamente investigada.  **3. Calibración de la Comunicación:** Resolver problemas en la comunicación entre el robot y el dispositivo.  
   **Responsable:**-Angelo Coriza  
   **Seguidores:**---  
   **Producto:** comunicación remota con el robot
* **Informe 2:** Crear informe del avance del proyecto **Responsable:**-David Orellana  
   **Seguidores:** - Nicolas Vargas , Angelo Coriza  
   **Producto:** Informe con los avances realizados y la corrección del informe entregado anteriormente en formato WORD
* **Presentación 2:** Crear presentación del avance del proyecto   
   **Responsable:**-David Orellana  
   **Seguidores:** - Nicolas Vargas, Angelo Coriza  
   **Producto:** Presentación en formato Power Point

**3.2. Asignación de tiempo:**

****

**3.3. Personal-rol asignado:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Actividad** | **Descripción** | **Responsable** | **Seguidores** |
| Líder | Encargado de verificar el cumplimiento de los avances del proyecto en las fechas establecidas , velar por el trabajo en equipo , generar buenas relaciones entre los miembros del equipo . | Nicolas Vargas | --- |
| Programadores | Persona que se dedicara previamente a estudiar el lenguaje de programación Python para posteriormente codificar algoritmos de armado de cubo Rubik a dicho lenguaje de programación. | Nicolas Vargas | Angelo Coriza |
| Armadores | Persona encargada de arma el robot, conseguir piezas faltantes (en caso de de no estén dichas piezas) si están piezas no logran ser conseguidas se tratará de adaptar un mecanismo con las piezas que ya poseemos. | Nicolas Vargas | David Orellana – Angelo Coriza |
| Secretario | Persona encargada de subir las Bitácoras y/o avances del proyecto. | David Orellana | Nicolas Vargas – Angelo Coriza |
| Encargado de la wiki | Persona encargada de subir fotografías con avances para promocionar el proyecto que se está realizando . | David Orellana | --- |



**3.3. Gestión de Riesgos:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Riesgos** | **Probabilidad de Ocurrencia** | **Nivel de Impacto** | **Acción Remedial** |
| Falte tiempo para el desarrollo del proyecto | 70% | 3 | Se tratará de distribuir de manera mejor las actividades restantes o algún miembro del equipo deberá realizar más de alguna tarea con el fin de agilizar el avance del proyecto |
| Persona ajena al equipo manipule el Robot y lo dañe o use incorrectamente | 60% | 5 | Dejamos escondido en la sala del ayudando el robot del equipo con la finalidad de protegerlo de la manipulación de terceras personas ajenas al proyecto. |
| Algún miembro tenga problemas o abandone el equipo | 90% | 6 | Mantener una comunicación estable en el equipo. |
| Daño o pérdida de Tarjeta SD o Adaptador Wifi | 40% | 2 | Las tarjetas SD suelen ser muy frágiles a múltiples factores en caso de daño se harán copias de seguridad constantemente y se reemplazará el dispositivo dañado |
| Piezas faltantes en el armado del robot | 30% | 4 | Se deberá adaptar la pieza faltante los las piezas ya tenidas en el kit de Lego |



**4.1. Recursos Hardware-Software requeridos**

* **Hardware:** Para el exitoso desarrollo del proyecto se necesita de Un Kit de lego MindStorms con todos sus accesorios, un PC para codificar los algoritmos en Python , una tarjeta SD, un adaptador WIFI y un cubo de Rubik’s
* **Software:** Se requiere del sistema operativo (para cargador en el robot y enviar los códigos) el cual fue entregado en clases, el IDE Visual Studio Code , el lenguaje de Programación Python el cual puede ser descargado de su sitio WEB

**4.2. Estimación de Costos (Hardware, Software, Recursos Humanos)**

* **Hardware:** 1 notebook o PC para realizar la programación de algoritmos de armado de cubo Rubik’s , 1 kit de Lego Mindstorm, cubo rubik 3x3, tarjeta sd, dongle wi-fi .

* **Software:** Los software utilizados en este proyecto no poseen costo ya que puede ser descargado de la web sin ningún costo .
* **Recursos Humanos:** El conocimiento de cada miembro es esencial para el desarrollo de este proyecto ya que cada uno aporta su conocimiento para potenciar el éxito que tendrá en proyecto.
  + Valor Hora:$9.500
  + Horas de trabajo(en la semana):12 horas
  + Semanas de trabajo: 18
  + Total de costos(Por persona):$2.052.000

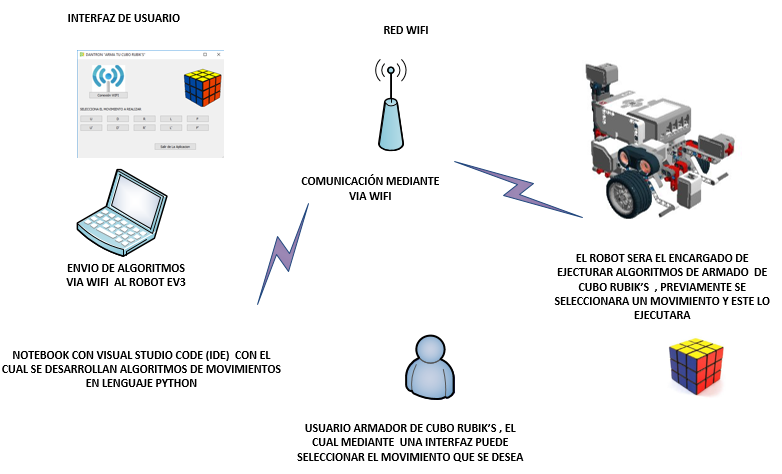
|  |  |
| --- | --- |
| **Productos** | **Costos(CLP)** |
| Robot EV3 | $490.209 |
| MicroSD | $6.000 |
| Adaptador | $9.000 |
| Cubo Rubik | $8.000 |
| Software | $0 |
| Valor equipo(3 personas) | $6.156.00 |
| **Total** | **$6.669.209** |

1. **Análisis-diseño**

**5.1 Especificación de Requerimientos (incluyendo método/algoritmos considerados para resolver el cubo Rubik)**  
  
Requerimientos:

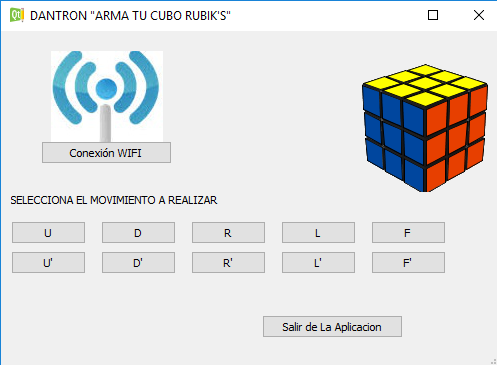
|  |  |
| --- | --- |
| **Funcional:** | **No Funcional:** |
| El robot ejecuta movimientos básicos y algunos de mayor complejidad.Los cuales son:  **Basicos:**R ,R’,L,L’,U,U’,D,D’,F,F’  **Complejos:**Sexy move,Cruz,Sexy move inverso,Cuatro vértices,Cambio Aristas | El robot ejecutará los movimientos básicos en un tiempo de 20 segundos y los complejos en 5 minutos |
| En ocasiones el cubo requerirá una calibración por parte del usuario | Se requiere una conexión WIFI estable para un uso más cómodo y no perder el vínculo entre los dispositivos. |
| Se recomienda que usuario sea un armador del cubo rubik,en caso contrario se requiere que lea el manual de usuario. | Se recomienda el uso del sistema operativo Windows. |
| El robot no armará el cubo Rubik’s de manera independiente | El robot solo armara un cubo rubik 3x3. |
| La conexión entre los dispositivos (Robot - PC) solo debe ser por WIFI ya que por otro método no es posible realizar dicha funcional | El sistema operativo del robot debe ser EV3dev S[tretch](https://www.google.com/search?client=ubuntu&hs=JRr&channel=fs&q=ev3dev+stretch&sa=X&ved=0ahUKEwitgMq4hejeAhVFgJAKHZSSAK4Q7xYIKigA) |

**5.2 Arquitectura Propuesta (incluyendo aspectos de comunicación)**



* **Computador:** En el computador está el programa que permite la ejecución de movimientos los cuales ayudan al armado de un cubo Rubik’s de 3 x 3 previamente estos códigos fueron codificados en lenguaje de programación Python y enviados al robot mediante la extensión de ev3 de Visual Studio Code.
* **Usuario:** El usuario tendrá la interfaz de usuario en la cual le aparecerán los movimientos que pueda realizar , dichos movimientos le ayudarán en el armado del cubo Rubik’s.
* **Robot Lego Mindstorms:** Será el encargado de realizar los movimientos que elija el usuario en la interfaz de usuario estará conectado mediante la aplicación ambos están vinculados mediante conexión WIFI.
* **WIFI :** Sería ideal que la conexión sea estable para no perder dirección IP y no entorpecer la comunicación entre los dispositivos (PC - Robot ) .

**5.3 Diseño de la Interfaz Usuario**



Ventana de la interfaz de usuario la cual en la parte superior tiene la opción para conectarse al robot Lego Mindstorms mediante una conexión WIFI mediante dicha conexión el robot actúa como servidor y el PC actúa como servidor al presionar los botones de movimiento el robot realizará dicho movimiento el cual le ayudará al usuario a armar un cubo RUBIK’S 3 x 3 mediante un algoritmo de armado .

1. **Implementación**

6.1 Descripción de los programas implementados (entradas, salidas, procesos)  
  
Los movimientos básicos creados para la implementación de de los movimientos del cubo son los siguientes :  
**CambiarCara()**: Este consiste en el movimiento del brazo para trasladar la cara frontal hacia arriba.  
**def** **BaseAH():** Mueve la base donde está el cubo 90° en sentido anti horario.  
**def** **BaseNH():** Mueve la base donde está el cubo 90° en sentido horario.  
**def** **Descanso():** Retrae el brazo del robot para un movimiento de la libre de la base .  
**def** **Accion():** El brazo vuelve a estar sobre el cubo.  
  
Las siguientes funciones están compuestas de las funciones anteriores para realizar los movimientos del cubo:  
**def** **Dn():**Realiza movimiento Down normal.  
   
**def Dp():** Realiza movimiento Down prima.  
 

**def Un():** Realiza movimiento Up normal.



**def Up():** Realiza movimiento Up prima.  
 

**def Fn():** Realiza movimiento Frontal normal.

  
**def Fp():** Realiza movimiento Frontal prima.  
 

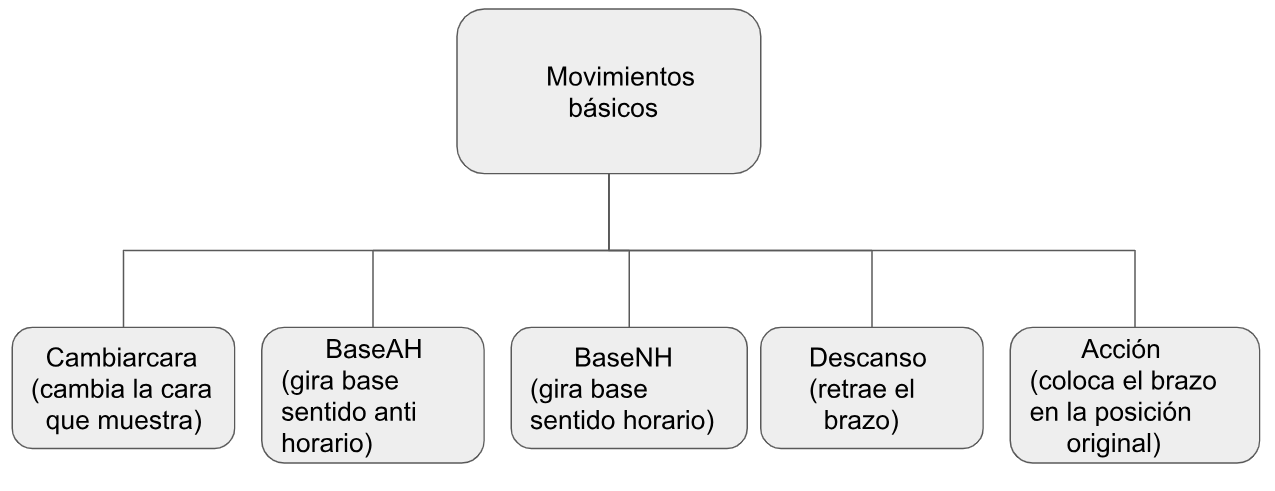
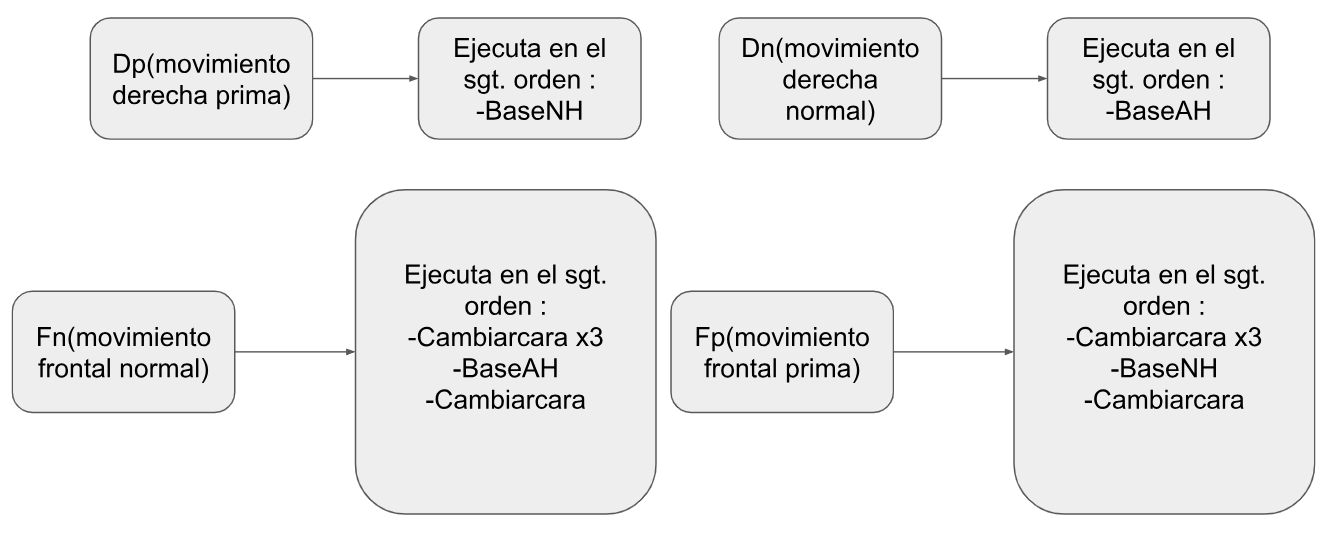
**def Rn():** Realiza movimiento Right normal.  
 

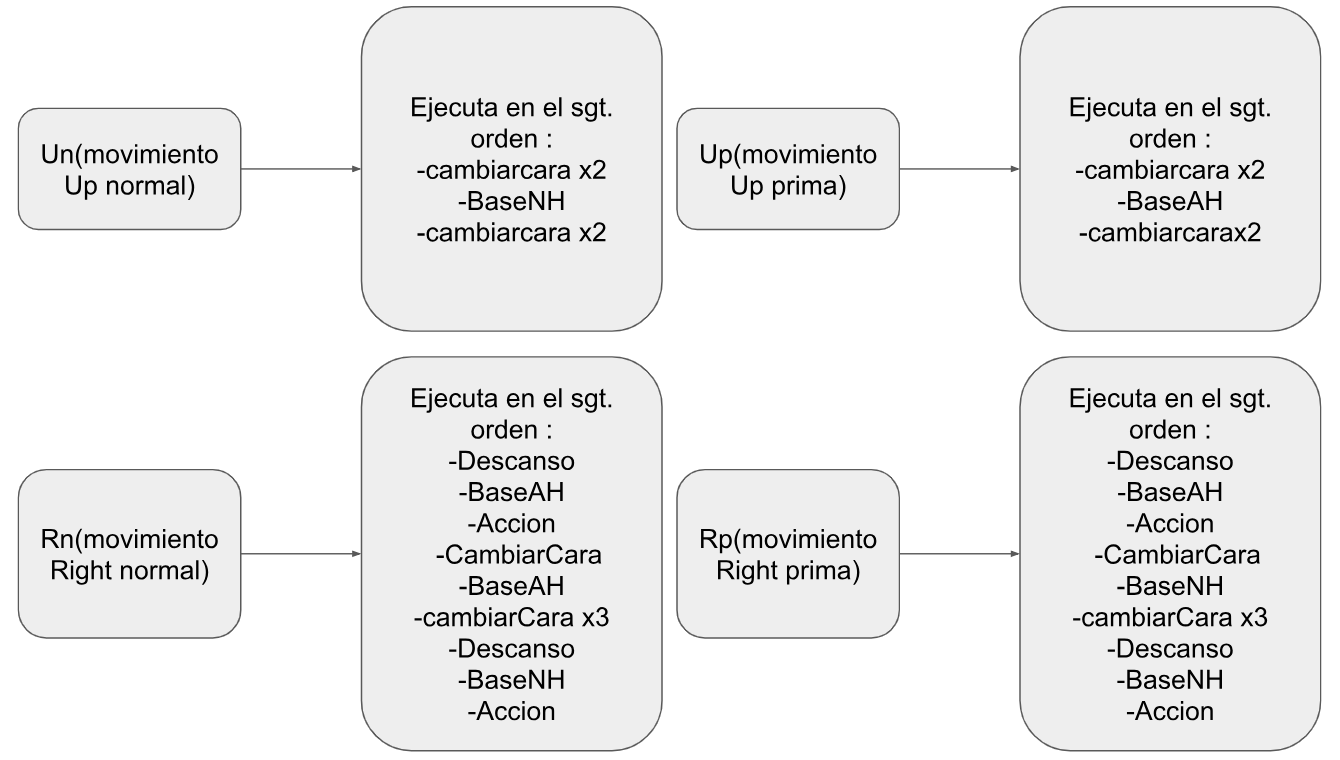
**def Rp():** Realiza movimiento Right prima.  
 

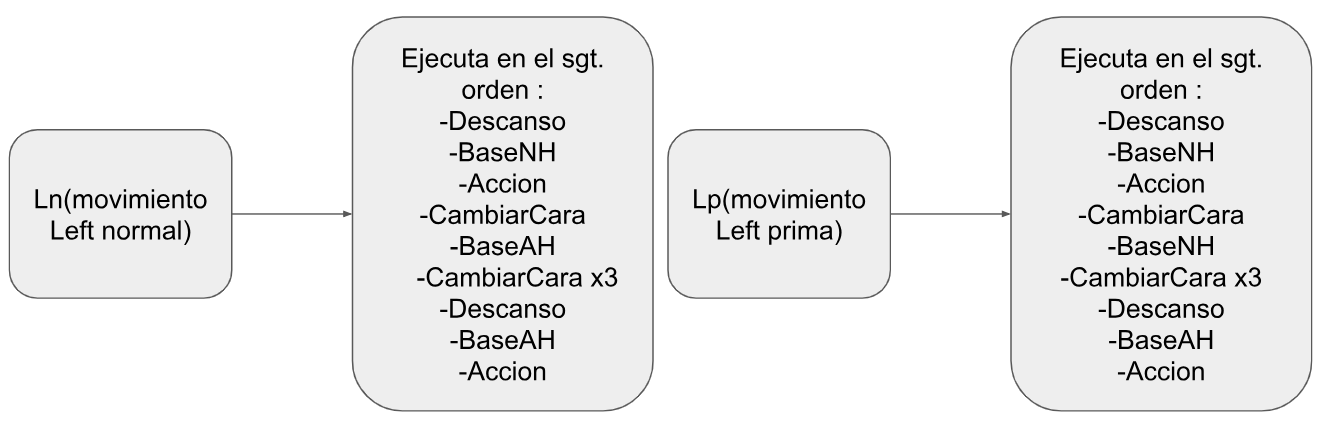
**def Ln():** Realiza movimiento Left normal.  
 

**def Lp():** Realiza movimiento Left prima.  
 

6.2 Diagrama de interacción entre programas

**  
  
  
**

****

****

1. **Resultados**   
   **7.1 Estado actual del proyecto**

El proyecto actualmente se encuentra en un 75%,ya que tenemos los códigos de movimiento a su vez estos fueron enviados al Robot mediante la extensión EV3 del IDE Visual Studio Code , este los ejecuta correctamente pero debido a la rigidez del cubo y a la descalibración del robot se debe calibrar al finalizar cada movimiento para evitar movimientos erróneos o que este realice un movimiento sin mover alguna cara del cubo , sólo quedaría pendiente la conexión mediante WIFI mediante una arquitectura cliente - servidor en lo que estamos trabajando como equipo para buscar una solución adecuada.

**7.2 Problemas encontrados y soluciones propuestas  
-Problemas en armado**   
El problema que se presentó con el armado del robot fue la falta de piezas con respecto a manual del robot EV3.La solucion propuesta para el problema fue que se nos facilitaron piezas Lego y hubieron piezas sobrantes del sensor que se omite para poder utilizarlas.

También surgió el problema de tener que adaptar la base giratoria, ya que, algunas de las piezas necesarias para su armado no se encontraban en el kit de Mindstorms. La solución planteada fue el crear desde cero la base con las piezas no utilizadas.  
  
Base rígida : al ejecutar los movimientos de el robot, a la base le costaba moverse por exceso de piezas al construirlo .La solución planteada fue crear una nueva base un poco más alta y con menos piezas.   
  
**-Problemas en programación**   
El primer problema encontrado fue el software del brick ev3 al instalar ev3dev instalando la versión “ev3dev-jessie” con el cual no pudimos crear códigos para el movimiento del robot por problemas con las librerías de esta versión. La solución propuesta a este problema fue instalar la versión “ev3dev-stretch” ,este problema nos quitó demasiado tiempo ya que estuvimos paralizados en busca de la solución al problema y después de muchos intentos sin éxito logramos solucionar el inconveniente presentado.

**7.3 Conclusiones**

Llegados a este punto del proyecto ya casi terminando logramos aprender mucho y reconocer muchos errores que tuvimos como equipo los cuales nos afectó principalmente en el tiempo y labores asignadas a cada miembro del equipo ,sin embargo, cada error nos sirvió para aprender y evitarlo en un futuro .

Mediante un análisis , implementación y una ejecución se logró desarrollar algoritmos codificados en python los cuales ya se encuentran en el robot y listos para ser ejecutados previamente mediante una conexión WIFI.

El proyecto ya se encuentra en su fase final donde solo nos falta realizar una conexión entre el pc -robot para que el usuario haga un uso cómodo del producto final,gracias a lo que logramos aprender en el tiempo del proyecto sabemos que debemos enfocar bien los tiempos como equipo para lograr superar este punto y dar fin al proyecto.

1. **Referencias (estándar IEEE)**

* **Sitio web de Lego Mindstorms :** Explora una galería de robots épicos de LEGO y LEGO MINDSTORMS creados por fanáticos, y haz clic para obtener todos los detalles, instrucciones de construcción y misiones de programación.

[**https://www.lego.com/en-us/mindstorms/build-a-robot**](https://www.lego.com/en-us/mindstorms/build-a-robot)

* **Python :** Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos

[**https://www.python.org/**](https://www.python.org/)

* **Visual Code Studio :** Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

[**https://code.visualstudio.com/**](https://code.visualstudio.com/)

* **Sitio de EV3DEV :** Ev3dev es un sistema operativo basado en Debian Linux que se ejecuta en varias plataformas compatibles con LEGO® MINDSTORMS, incluyendo LEGO® MINDSTORMS EV3 y BrickPi impulsado por Raspberry Pi.

[**https://www.ev3dev.org/**](https://www.ev3dev.org/)

**Anexo**

**Código fuente :**

**#!/usr/bin/env python3**

**from ev3dev.ev3 import \***

**from time import sleep**

**def CambiarCara():**

**mA = LargeMotor('outA')**

**i=1**

**if i == 1:**

**mA.run\_to\_rel\_pos(position\_sp=70, speed\_sp=750, stop\_action="hold")**

**time.sleep(1)**

**i=i+1**

**if i == 2:**

**mA.run\_to\_rel\_pos(position\_sp=-70, speed\_sp=750, stop\_action="hold")**

**time.sleep(1)**

**def BaseAH():**

**mB = LargeMotor('outB')**

**mB.run\_to\_rel\_pos(position\_sp=-285, speed\_sp=900, stop\_action="hold")**

**time.sleep(1)**

**def BaseNH():**

**mB = LargeMotor('outB')**

**mB.run\_to\_rel\_pos(position\_sp=285, speed\_sp=900, stop\_action="hold")**

**time.sleep(1)**

**def Descanso():**

**m = LargeMotor('outA')**

**m.run\_to\_rel\_pos(position\_sp=-100, speed\_sp=500, stop\_action="hold")**

**sleep(1)**

**def Accion():**

**m = LargeMotor('outA')**

**m.run\_to\_rel\_pos(position\_sp=100, speed\_sp=900, stop\_action="hold")**

**sleep(1)**

**def Dp():**

**BaseNH()**

**def Dn():**

**BaseAH()**

**def Fn():**

**i=0**

**while i!=3:**

**CambiarCara()**

**i=i+1**

**BaseAH()**

**CambiarCara()**

**def Fp():**

**i=0**

**while i!=3:**

**CambiarCara()**

**i=i+1**

**BaseNH()**

**CambiarCara()**

**def Un():**

**i=0**

**while i!=2:**

**CambiarCara()**

**i=i+1**

**BaseNH()**

**i=0**

**while i!=2:**

**CambiarCara()**

**i=i+1**

**def Up():**

**i=0**

**while i!=2:**

**CambiarCara()**

**i=i+1**

**BaseAH()**

**i=0**

**while i!=2:**

**CambiarCara()**

**i=i+1**

**def Rn():**

**Descanso()**

**BaseAH()**

**Accion()**

**CambiarCara()**

**BaseAH()**

**i=0**

**while i!=3:**

**CambiarCara()**

**i=i+1**

**Descanso()**

**BaseNH()**

**Accion()**

**def Rp():**

**Descanso()**

**BaseAH()**

**Accion()**

**CambiarCara()**

**BaseNH()**

**i=0**

**while i!=3:**

**CambiarCara()**

**i=i+1**

**Descanso()**

**BaseNH()**

**Accion()**

**def Ln():**

**Descanso()**

**BaseNH()**

**Accion()**

**CambiarCara()**

**BaseAH()**

**i=0**

**while i!=3:**

**CambiarCara()**

**i=i+1**

**Descanso()**

**BaseAH()**

**Accion()**

**def Lp():**

**Descanso()**

**BaseNH()**

**Accion()**

**CambiarCara()**

**BaseNH()**

**i=0**

**while i!=3:**

**CambiarCara()**

**i=i+1**

**Descanso()**

**BaseAH()**

**Accion()**