

**UNIVERSIDAD MAYOR DE SAN SIMON**

**FACULTAD DE CIENCIAS Y TECNOLOGIA**



**TRABAJO DE INVESTIGACIÓN**

**SISTEMA DE MONITOREO Y CONTROL DE ROBOT**

Presentado por :

Univ. Alexander Mamani Y.  
FCyT  
Cocambamba  
akeymy4@gmail.com

Univ. Adan F. Toledo  
FCyT  
Cocambamba  
AdanT5@gmail.com

Univ. Juan C. Veizaga  
FCyT  
Cocambamba  
carlos12@gmail.com

Director :

Lic. Yony Richard Montoya Burgos

Cochabamba, 2018

Univ. Alexander Mamani Y.  
FCyT  
Cochabamba  
akeymy4@gmail.com

Univ. Adan F. Toledo  
FCyT  
Cochabamba  
AdanT5@gmail.com

Univ. Juan C. Veizaga  
FCyT  
Cochabamba  
carlos12@gmail.com

## Resumen

Los vehículos no tripulados terrestres han demostrado que tienen múltiples usos según el concepto que se les de, dichos usos pueden ser beneficiosos para la sociedad buscando la mejoría de su calidad de vida, pero como siempre es muy difícil que todo sea bueno, y así como hay impactos positivos los hay negativos.

Estos vehículos no tripulados se están empleando en el sector portuario para realizar tareas de mantenimiento, seguridad, gestión del tráfico y control del transporte de mercancías entre otras labores. La reducción de costes es una de sus ventajas ya que elimina los accidentes y mejora los procesos al poder actuar de forma autónoma y sin poner en riesgo a los trabajadores.

Entonces podemos apreciar la gran importancia de estos tipos de vehículos no tripulados.

En el presente trabajo se pretende implementar un Software de que nos permita manejar, monitorear y mostrar datos estadísticos de un vehículo terrestre no tripulado, mediante una interfaz gráfica que facilite al usuario el manejo del mismo, así también describiremos cada módulo del software y patrones de diseño utilizados para su creación y posterior implementación.

---

**Palabras clave:** Dron terrestre, Arduino, Sistema de manejo y monitoreo de dron terrestre.

## Índice

Índice	2
1. INTRODUCCIÓN	3
2. DISEÑO DEL SOFTWARE	3
2.1. Arquitectura . . . . .	3
2.2. View . . . . .	4
2.3. Controller . . . . .	4
2.4. Model . . . . .	5
3. ALGORITMIA	6
4. METODOLÓGICA Y FUENTE	8
4.1. Composicion de grupo . . . . .	8
5. GLOSARIO DE TÉRMINOS	9
5.1. Diccionario . . . . .	9
6. BLIBLIOGRAFIA	10
Referencias	10

## 1. INTRODUCCIÓN

Arduino es una compañía open source y open hardware, así como un proyecto y comunidad internacional que diseña y manufactura placas de desarrollo de hardware para construir dispositivos digitales y dispositivos interactivos que puedan sensar y controlar objetos del mundo real. Arduino se enfoca en acercar y facilitar el uso de la electrónica y programación de sistemas embebidos en proyectos multidisciplinarios.

Teniendo en cuenta los problemas planteados con que cuenta nuestra sociedad como ser el sector portuario, y entre otros.

Se pretende desarrollar módulos que nos permitan manipular el hardware de un dron terrestre, como así también monitorear y extraer datos estadísticos del mismo, para después implementar en un Robot que tiene como microcontrolador Arduino.

La manipulación de estos módulos se llevará a cabo a través de una interfaz Gráfica, que reaccionará a ciertos eventos.

El Robot(hardware), cuenta con 3 Servo Motores dos para las ruedas y una para una pinza que nos permite agarrar objetos, además de un módulo bluetooth que nos permitirá la transmisión y recepción de datos, un sensor ultrasonido que es usado para medir la distancia entre el Robot y un objeto.

Las tecnologías usadas para el desarrollo e implementación del proyecto son las siguientes:

- como Lenguaje de Programación se usó Python en su versión 3.
- como almacenamiento de datos se usó la estructura XML.

## 2. DISEÑO DEL SOFTWARE

### 2.1. Arquitectura

Como base para el desarrollo del software se usó la Arquitectura de software MVC, ya que el mismo nos permite separar el modelo de datos del software de la interfaz gráfica del usuario, así mismo nos ayudó a la mejor estructuración del software y a ser más escalable al momento de implementar nuevas funcionalidades.

Los módulos que se desarrollaron se encuentran en los siguientes paquetes:

- **View.-** Paquete que contiene módulos que nos permitan manipular la interfaz gráfica de nuestro Software, separando la interfaz del usuario con el modelo del negocio.

- **Controller.-** Paquete que contiene modulos que nos permitiran capturar los eventos producidos en nuestro View, y en base a estos desencadenar acciones que seran reflejadas en nuestra Interfaz Grafica
- **Model.-** Paquete que contiene modulos que nos permitiran manipular los datos generados por nuestro software, como ser guardar los datos al momento en que se graba una movimiento del Robot

## 2.2. View

En este Paquete se encuentran los modulos encargados de manejar la interfaz grafica, con la cual un Usuario cualquiera podra interactuar de manera intuitiva con nuestro software.

Este paquete cuenta con las siguientes clases :

- **Ventana.-** Clase que se encarga de graficar todos los componentes de la interfaz grafica como ser: label, botones, Robot,.. etc
- **Robot.-** Clase Singleton que nos permite dibujar en la Ventana todos los comportamientos del robot como ser Avanzar, Retroceder, girar a la izquierda, girar a la derecha, Detener, abrir y cerrar pinza
- **Grafica.-** Clase que despliega distintos tipos de graficos estadisticos , con los datos proporcionados por la Clase Model.Estadistica.

## 2.3. Controller

En este Paquete se encuentran los modulos encargados de capturar los eventos producidos en el View, y en base a estos realizar una accion, ya sea de grabar o reproducir un accion previamente grabada.

Este paquete cuenta con las siguientes clases y modulos :

- **Evento.-** Clase parent de clase Ventana, el cual captura eventos de teclado y los asocia con una funcion, tambien cuenta con un metodo que captura eventos de tipo clickeado de un componente button, a estos de igual manera los asocia a una funcion.
- **acciones.-** Modulo que tiene funciones, las cuales nos permite realizar acciones, como ser :

- **grabar movimiento.-** funcion que graba movimientos del robot, en intervalos de tiempo que dependeran que si hubo un cambio de estado del robot, de esta manera evitamos almacenar mucha informacion en nuestro archivos XML
  - **reproducir movimiento.-** funcion que se encarga de reproducir un movimiento previamente guardado, recibe como parametro la ruta del archivo XML del cual vamos a reproducir.
  - **monitorear robot.-** funcion que se encarga de monitorear las acciones que realiza el robot, para despues grabar en un archivo XML y reproducirlo cuando el usuario lo vea conveniente.
  - **reportes Estadisticos.-** funcion que se encarga darnos los reportes estadisticos acerca de nuestro Robot.
- **Conexion.-** Clase singleton que nos permitira la recepci3n y transmisi3n de datos con el robot.

## 2.4. Model

En este Paquete se encuentran los modulos encargados de la manipulacion de datos de nuestro software.

Como estructura de almacenamiento tomamos a las estructuras XML, ya que nos permite guardar y acceder a su informacion de un manera muy sencilla, y dada su portabilidad podemos acceder a ella desde cualquier ruta en la que este alojada.

Este paquete cuenta con las siguientes clases y datos :

- **Data.-** Clase que extiende de la clase etree.Element que nos permitira manipular datos de estructuras de tipo XML.

La estructura XML es la siguiente:

- **Objeto datos.-** Objeto que almacena el punto de partida desde que se empezo a grabar el movimiento del robot
- **Objeto movimiento.-** Objeto que almacena datos acerca de un movimiento grabado, tiene los siguiente atributos:
  - **tiempo.-** Atributo que contiene un timer en milisegundos que es tiempo que se ejecuto un movimiento "X"
  - **movimiento.-** Atributo que contiene el tipo de movimiento que se hizo, ya se que avanzo, se detuvo o si giro.  
El valor de este atributo dependera de la Clase Evento, puesto que esta es la que maneja los eventos de teclado, y cada evento esta relacionado

a una acción, es aquí donde se puede apreciar, ya que este atributo representa una acción que realizó la clase Evento.

- **pinza.-** Atributo que contiene el estado de si pinza, es decir si este la tiene abierta o cerrada "T", si esta abierta y "N en caso contrario".

- **Estadística.-** Clase la cual nos proporciona datos estadísticos del robot, desde que se inicio el programa, como ser cuantos movimiento hizo, cuales fueron los movimientos que mas se realizó, ..etc

### 3. ALGORITMIA

Un problema que se tuvo fue la lectura y escritura de los datos al momento de grabar y reproducir el movimiento del robot. Esto se dio ya que se estaba almacenando los estados de los movimientos del robot cada cierto intervalo de tiempo "X", lo cual implicaba que la escritura de los datos fuera bastante.

Un ejemplo fue que se grababa los estados del robot cada 100 milisegundos, por lo cual si queríamos grabar 5 minutos de movimientos, implicaría escribir 3000 movimientos. A esto aumentar que si queríamos movimientos mas exactos habría que reducir el intervalo de tiempo, lo cual implica mas datos y mas propenso a perder los mismo.

También como contábamos con varios datos esto constituía un gran uso de la memoria RAM para reproducir todos estos movimientos, y sobre esto agregar que la conexión al arduino por el puerto serial es Asíncrona, lo cual implica que si perdíamos un dato o varios de estos, nuestra gráfica ya no será tan exacta como esperamos.

A continuación se muestra los algoritmos mencionados:

---

#### **Algorithm 1** Algorithm to store robot movements

---

**input** : root of a file XML

**output:** file XML with the data of robot movements

Open file XML;

**while** *is recording* **do**

    write movements;

    sleep(100);

**end**

close file XML;

---

---

**Algorithm 2** Algorithm to reproduce robot movements

---

**input** : root of a file XML

**output:** play robot movements

Open file XML;

**while** *there is data* **do**

    data = read movements;

    reproduce( data );

    sleep(100);

**end**

close file XML;

---

Para solucionar este problema, puesto que nuestro algoritmo no era eficiente y menos eficaz, se implemento el siguiente algoritmo que acoparacion del anterior, este no guarda datos en cada intervalo de tiempo, sino que guarda datos cada vez que se detecta cambios en el robot, es decir que si se detecta un cambio de estado en el robot, como ser girar, mover, abrir o cerrar pinza, este guarda el tiempo que tarda en realizar esta accion, de esta manera optimizamos los recursos de nuestro computador (Eficiencia Temporal) como ser memoria (RAM), puesto que ya no tiene que leer datos cada cierto intervalo de tiempo sino que cuando sea necesario y ademas tenemos movimientos mas exactos, puesto que reducimos la perdida de datos al momento de reproducirlo, ya que trabajamos con tiempos de reloj y no asi con intervalos.

Acontinuacion se muestra el algoritmo mencionado:

---

**Algorithm 3** New algorithm to store robot movements

---

**input** : root of a file XML

**output:** file XML with the data of robot movements

open file XML;

write initial data;

**while** *is recording* **do**

**if** *is firts interaction* **then**

        time = get Time Of Computer;

        lastMovement = Robot movement ;

**else if** *was a change in the robot* **then**

        time = get Time Of Computer - time;

        write lastMovement and time;

        lastMovement = Robot movement;

**end**

close file XML;

---



---

**Algorithm 4** New algorithm to reproduce robot movements

---

**input** : root of a file XML

**output:** play robot movements

Open file XML;

read initial data;

**while** *there is data* **do**

    data = read movements;

    time = read time;

    reproduce( data, time );

**end**

close file XML;

---

## 4. METODOLÓGICA Y FUENTE

Las fuente de información que tomamos como base para desarrollar el proyecto son principalmente publicas y bibliográficas

Tambien para el desarrolo del proyecto se trabajo con la metodologia agil Scrum, puesto que era la que mas se adaptaba mas a nuestra comodidad.

Como Sistema de Control de Versiones se uso GitHub, ya que nos permite el trabajo colaborativo y no presencial(<https://github.com/akey96/TallerDeProgramacion>)

Para el desarrollo de la documentacion se uso el formato ACM.

### 4.1. Composicion de grupo

Esta parte falta definir, puesto que recién implementaremos modulos faltantes.....

## 5. GLOSARIO DE TÉRMINOS

### 5.1. Diccionario

- **Arduino.-** Arduino es una plataforma de prototipos electrónica de código abierto (open – source) basada en hardware y software flexibles y fáciles de usar. Está pensado e inspirado en artistas, diseñadores, y estudiantes de computación o robótica
- **Concurrencia.-** evento cuando mas de un proceso intentan acceder al mismo recurso
- **MVC .-** Arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.
- **Singleton.-** Es un patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto.Su intención consiste en garantizar que una clase solo tenga una instancia y proporcionar un punto de acceso global a ella.
- **Eficiencia Espacial.-** Cantidad de recursos espaciales ( de almacén) que un algoritmo consume o necesita para su ejecución
- **SCV.-** Software que administra el acceso a un conjunto de ficheros, y mantiene un historial de cambios realizados. El control de versiones es útil para guardar cualquier documento que cambie con frecuencia, como código fuente, documentación o ficheros de configuración.
- **XML.-** XML proviene de eXtensible Markup Language (“Lenguaje de Marcas Extensible”). Se trata de un metalenguaje (un lenguaje que se utiliza para decir algo acerca de otro) extensible de etiquetas que fue desarrollado por el Word Wide Web Consortium (W3C)
- **Comunicacion Asincrona.-** Es la conexión que se establece entre el cliente y el servidor que permite la transferencia de datos no sincrónica, o sea el cliente puede realizar varias peticiones al servidor sin necesidad de esperar por la respuesta de la primera.

## 6. BIBLIOGRAFIA

### Referencias

- [1] mundoerp <http://mundoerp.com/blog/Singleton>
- [2] arduinodhtics <https://arduinodhtics.weebly.com/iquiestqueacute-es.html>
- [3] definicion <https://definicion.de/xml/>
- [4] ecured <https://www.ecured.cu/Comunicaci%C3%B3n-as%C3%ADncrona>
- [5] mundoerp <http://mundoerp.com/blog/sistemas-de-control-de-versiones/>