



UNIVERSIDAD DE TALCA  
FACULTAD DE INGENIERÍA  
ESCUELA DE INGENIERÍA CIVIL EN COMPUTACIÓN

**Smart Irrigation: Prototipo de riego inteligente de  
bajo costo para el uso eficiente del agua**

NICOLÁS GASTÓN MATORANA BARRIOS

Profesor Guía: BENJAMÍN INGRAM

Memoria para optar al título de  
Ingeniero Civil en Computación

Curicó – Chile  
Diciembre, 2015



UNIVERSIDAD DE TALCA  
FACULTAD DE INGENIERÍA  
ESCUELA DE INGENIERÍA CIVIL EN COMPUTACIÓN

**Smart Irrigation: Prototipo de riego inteligente de  
bajo costo para el uso eficiente del agua**

NICOLÁS GASTÓN MATORANA BARRIOS

Profesor Guía: BENJAMÍN INGRAM

Profesor Informante: PROFESOR INFORMANTE 1

Profesor Informante: PROFESOR INFORMANTE 2

Memoria para optar al título de  
Ingeniero Civil en Computación

El presente documento fue calificado con nota: \_\_\_\_\_

Curicó – Chile

Diciembre, 2015

*Dedicado a ...*

## **AGRADECIMIENTOS**

Agradecimientos a ...

## TABLA DE CONTENIDOS

	página
<b>Dedicatoria</b>	<b>I</b>
<b>Agradecimientos</b>	<b>II</b>
<b>Tabla de Contenidos</b>	<b>III</b>
<b>índice de Figuras</b>	<b>VI</b>
<b>índice de Tablas</b>	<b>VIII</b>
<b>Resumen</b>	<b>IX</b>
<b>1. Introducción</b>	<b>10</b>
1.1. Descripción de la propuesta . . . . .	10
1.1.1. Contexto del proyecto . . . . .	10
1.1.2. Trabajo relacionado . . . . .	11
1.1.3. Definición del problema . . . . .	11
1.1.4. Propuesta de solución . . . . .	12
1.2. Hipótesis . . . . .	13
1.3. Objetivos . . . . .	13
1.4. Alcances . . . . .	13
1.5. Metodología . . . . .	14
<b>2. Marco teórico</b>	<b>15</b>
2.1. Programación del riego . . . . .	15
2.1.1. Textura del suelo . . . . .	16
2.1.2. Como regar los suelos . . . . .	17
2.2. Problemas identificados . . . . .	18
2.3. Información disponible . . . . .	19
2.4. Sistema basado en reglas . . . . .	19
2.4.1. Modus Ponens . . . . .	22
2.4.2. Estrategias de inferencia . . . . .	22

2.5.	Servicios web meteorológicos . . . . .	23
2.5.1.	¿Qué es un servicio web? . . . . .	23
2.5.2.	Wunderground . . . . .	24
2.6.	Trabajos relacionados . . . . .	25
2.6.1.	Cultivar . . . . .	25
2.6.2.	Rachio IRO . . . . .	26
2.6.3.	Rain Bird . . . . .	27
2.6.4.	Tabla comparativa . . . . .	27
2.7.	Hardware . . . . .	28
2.7.1.	Raspberry pi . . . . .	28
2.7.2.	Arduino . . . . .	29
2.7.3.	Servidor de base de datos . . . . .	29
2.7.4.	Servidor web . . . . .	30
2.7.5.	Sensores de humedad . . . . .	30
2.7.6.	Sensor de flujo de agua . . . . .	31
2.7.7.	Relé . . . . .	32
2.7.8.	Válvula solenoide . . . . .	33
2.7.9.	GPS . . . . .	33
2.7.10.	Módem 3G . . . . .	34
2.7.11.	Cargador . . . . .	35
2.7.12.	Costo componentes . . . . .	35
2.8.	Software . . . . .	36
2.8.1.	Raspbian . . . . .	36
2.8.2.	Python . . . . .	36
<b>3.</b>	<b>Desarrollo</b> . . . . .	<b>38</b>
3.1.	Hardware . . . . .	38
3.1.1.	Arduino . . . . .	38
3.1.2.	Raspberry . . . . .	41
3.1.3.	GPS . . . . .	43
3.2.	Software . . . . .	45
3.2.1.	Información meteorológica . . . . .	45
3.2.2.	Modelo de riego . . . . .	46
3.2.3.	Servicio web de suelo . . . . .	48

3.2.4. Envío y consulta de datos, humedad y agua utilizada . . . . .	51
3.2.5. Aplicación web . . . . .	53
<b>4. Conclusión</b>	<b>55</b>
<b>5. Trabajo futuro</b>	<b>56</b>
<b>Glosario</b>	<b>57</b>
<b>Bibliografía</b>	<b>58</b>
<b>Anexos</b>	
<b>A: Trabajo realizado en FIA</b>	<b>61</b>
<b>B: Código fuente</b>	<b>63</b>
B.1. gps.py . . . . .	63
B.2. riego.py . . . . .	65
B.3. clima.py . . . . .	67
B.4. arduino.py . . . . .	68

## ÍNDICE DE FIGURAS

	página
2.1. Como y cuando regar . . . . .	15
2.2. Permeabilidad según textura del suelo . . . . .	17
2.3. Variación en la infiltración por textura del suelo . . . . .	18
2.4. Objetos y posibles valores para el ejemplo del cajero automático . . . . .	20
2.5. Ejemplos de reglas para sacar dinero de un cajero automático . . . . .	21
2.6. Arquitectura servicio web . . . . .	23
2.7. Comparación servicio meteorológico de Google v/s Wunderground para la localidad rural de Corcolén . . . . .	25
2.8. Tabla Comparativa . . . . .	28
2.9. Raspberry Pi modelo B+ . . . . .	29
2.10. Arduino . . . . .	29
2.11. Sensor de humedad . . . . .	31
2.12. Sensor de flujo de agua . . . . .	32
2.13. Relé de estado sólido . . . . .	33
2.14. Válvula solenoide . . . . .	33
2.15. Módulo GPS USB . . . . .	34
2.16. Módem 3G . . . . .	34
2.17. Cargador . . . . .	35
2.18. GPIO Raspberry Pi . . . . .	37
2.19. Pines GPIO . . . . .	37
3.1. Conexión sensores en Arduino . . . . .	40
3.2. Arduino en Raspberry . . . . .	41
3.3. Comunicación Raspberry Arduino . . . . .	42
3.4. Recibo de datos desde Arduino en Raspberry . . . . .	43
3.5. GPS en Raspberry, /dev/ttyUSB1 . . . . .	44
3.6. Datos GPS . . . . .	45
3.7. Respuesta del servicio wunderground al hacer una consulta . . . . .	46
3.8. Árbol reglas modelo de riego . . . . .	49
3.9. Registro almacenado en la base de datos . . . . .	50
3.10. Registros almacenados en la base de datos . . . . .	52

3.11. Humedad del suelo en tiempo real . . . . .	53
3.12. Consumo de agua mensual . . . . .	54
A.1. Exterior tríptico realizado en FIA . . . . .	61
A.2. Interior tríptico realizado en FIA . . . . .	62

## ÍNDICE DE TABLAS

	página
2.1. Tabla de precios . . . . .	35
3.1. Objetos y posibles valores para determinar un riego . . . . .	47
3.2. Variables predefinidas para la reglas del modelo . . . . .	47
3.3. Reglas modelo, cuando y como regar . . . . .	48

## **RESUMEN**

Un riego eficaz sólo es posible con un seguimiento periódico de las condiciones del agua del suelo y con la previsión de las futuras necesidades de agua. Retrasar el riego provoca estrés, o aplicar muy poca agua puede resultar en una pérdida sustancial. La aplicación de mucha agua se traducirá en costos adicionales y agua desperdienciada.

Lo que se busca en este proyecto es aumentar la eficiencia en los recursos hídricos de la zona en la que el dispositivo esté instalado entregando información relevante, que permita disminuir costos y mantener las áreas verdes a pesar del actual cambio climático que vive el planeta.

Esto se puede lograr con un dispositivo de riego de bajo costo, de fácil instalación, que no necesite configuraciones, autónomo e inteligente, que permita monitorear y controlar el riego de los jardines o áreas verdes manteniéndolos en condiciones ideales de humedad con el fin de ayudar al crecimiento y desarrollo saludable del césped.

Se les ofrecerá a los usuarios monitorear y controlar el riego, entregándoles información que les puede ser de gran utilidad, como por ejemplo, el agua utilizada en un día, todo esto a través de una aplicación web y/o móvil.

# 1. Introducción

---

## 1.1. Descripción de la propuesta

### 1.1.1. Contexto del proyecto

El proyecto consiste en desarrollar un sistema de riego autónomo e inteligente ideal para el uso en áreas verdes que utilizan sistemas de riego por ejemplo, por aspersión. Lo que se busca es reducir la intervención humana, aumentar la eficiencia en el uso del agua, ahorrar dinero y entregar datos estadísticos a los usuarios.

El sistema básicamente consta de sensores de humedad del suelo, un micro computador y válvulas. Cuando un sensor detecte que la humedad del suelo no es la adecuada para mantener el césped en buen estado, le enviará una señal a la Raspberry la que a su vez analiza datos meteorológicos y decide si es necesario o no hacer un riego. Con todo esto, se asegura que el jardín se encuentre en buen estado utilizando la menor cantidad de agua posible.

Para el proyecto se utiliza un micro computador el cual se programa en Python, y es el encargado de recibir los datos de entradas, que en este caso son los datos meteorológicos y los del sensor de humedad. Una vez que los datos son recibidos el micro computador los analiza y si se requiere un riego envía una señal la válvula solenoide que permitirá el paso del agua. También se utilizará un sensor de flujo para medir la cantidad de agua que se está utilizando y así entregar reportes al usuario.

Cabe destacar que este sistema no solo es útil para los jardines, sino que también se puede utilizar para los huertos y/o predios agrícolas que utilizan sistemas de riego automáticos.

### 1.1.2. Trabajo relacionado

Los trabajos que hoy en día son utilizados para resolver este problema cumplen su propósito general, que es el de automatizar el riego, pero en esta solución aun se requiere la intervención de las personas que tienen que configurar el dispositivo, encendido o apagado cuando sea necesario, esto hace referencia a los sistemas que utilizan temporizadores [13]. Estos dispositivos se activan cada cierto intervalo de tiempo sin tomar en cuenta variables relevantes como lo son la humedad del suelo o incluso si al momento de regar esta lloviendo.

También existen trabajos en los que se mide la humedad del suelo [11] pero aun no se utiliza eficientemente el agua. Por otro lado tampoco indican el consumo de agua que se ha utilizado en el riego y tampoco tienen sistema de prevención en caso de condiciones climáticas extremas, como puede ser el frío extremo que nos ayuden a mantener en buen estado nuestro jardín.

Estos dispositivos se pueden encontrar en tiendas de retail a costos elevados.

### 1.1.3. Definición del problema

Existen principalmente tres problemas que están asociados al consumo de los recursos hídricos:

Primero, la escasez de agua que existe hoy en día en gran parte de nuestro país, el año 2013 fue uno de los años más secos desde 1866. Existen déficit de precipitaciones [3] que van desde un 20 % llegando incluso hasta un 60 % entre las zonas de Copiapó y Talca al sur, estas son zonas de gran producción agrícola además entre los lugares mencionados son los que han tenido mayor inversión para poder satisfacer la demanda de agua dulce a las personas, a esto le sumamos que este tipo de agua también es utilizada para el riego agrícola, lo que hace este recurso aún más escaso.

Segundo, se utilizan técnicas de riego pobres en la agricultura (riego áreas verdes), es por ello que es muy importante y de suma urgencia idear nuevas técnicas de riego sustentables en las que se obtenga el mayor provecho del agua sin afectar la calidad del riego.

Por último, la falta de tecnologías innovadoras y baratas aplicadas al riego para el uso doméstico.

#### 1.1.4. Propuesta de solución

La solución al problema consiste en desarrollar un dispositivo de riego inteligente, automático, autónomo y de bajo costo que permita monitorear y controlar el riego de un jardín o área verde.

Este dispositivo consta principalmente de dos partes, la primera parte es el hardware que consiste en los sensores, válvulas, actuadores y microcomputador. La segunda parte es el software que tomará la información de los sensores y la información a través de internet con la que determinará cuándo y cómo aplicar los recursos hídricos al suelo.

Para el hardware se utilizarán piezas que se venden en el mercado y son de fácil acceso, por lo que no será un problema. Para el desarrollo del software, el caso ideal sería trabajar en conjunto con un Ingeniero Agrónomo experto en riego, pero en esta oportunidad no se hará así.

Cuando y cómo aplicar un riego es muy importante ya que de esta manera se está utilizando eficientemente el agua evitando escurrimiento, evaporación, salinización y otros problemas asociados al exceso de agua en el riego.

La conexión a internet se hará utilizando dispositivos 3G que tienen una amplia cobertura a través de todo el país gracias a la iniciativa del gobierno “Proyecto Bicentenario” que entrega acceso al 90 % de las localidades rurales.

El dispositivo al estar conectado a internet permitirá monitorear en línea la información que se recoge a través de los sensores, por ejemplo temperatura del lugar, humedad del suelo, cantidad de agua utilizada además se podrá controlar el riego para el caso de los usuarios no agricultores.

Otro punto importante consiste en alarmas, que alertarán a los usuarios cuando existan condiciones desfavorables para el jardín como lo son las heladas en invierno, el dispositivo generará una alarma que permitirá tomar medidas de forma rápida y oportuna disminuyendo los daños que se puedan ocasionar.

Como se espera que esto funcione de forma autónoma y utilice eficientemente el agua, no se realizarán riegos cuando exista alguna probabilidad de lluvia, por ejemplo si se espera para mañana una lluvia y los sensores de humedad indican que es necesario regar, el sistema no regará ese día o regará utilizando menos agua, esta es una característica que les será de mayor utilidad a los usuarios no agricultores ya que a ellos los que se quieren despreocupar totalmente del riego de áreas verdes.

Una variable que se considerará, distinguirá y hará diferente este dispositivo de otros existentes, es que al estar consciente de su ubicación actual podrá determinar el tipo de suelo en el que se encuentra, con esto se mejorará el impacto del riego en cada tipo de suelo, ya que permitirá saber la capacidad de absorción que éste tiene. Con esta información se podrá saber qué tipo de riego aplica mejor con el fin de utilizar el agua lo más inteligente y eficientemente posible.

## 1.2. Hipótesis

- El prototipo es realizable con la tecnología existente.
- Las personas se desocuparán por el riego de su jardín.
- El uso de este sistema cuidará el jardín en todas las estaciones del año.
- La utilización del dispositivo de riego disminuirá el consumo de agua.

## 1.3. Objetivos

### Objetivo general

- El objetivo es diseñar un prototipo inteligente de bajo costo que será utilizado en los jardines para automatizar y monitorear el riego de un jardín o áreas verdes.

### Objetivos específicos

- Diseñar un prototipo del dispositivo (Hardware).
- Diseñar un modelo que determinar cuando y como realizar un riego (Software).
- Diseñar un prototipo de aplicación web y/o móvil para los usuarios.

## 1.4. Alcances

- El sistema de riego (cañerías, aspersores, etc.) debe estar previamente instalado.

- El modelo de riego será básico y se podrá mejorar con más tiempo y personas especializadas en el área.
- En el monitoreo los usuario podrán ver la humedad del suelo y la cantidad de agua utilizada en el riego.
- Se diseñará una aplicación web y/o móvil que permita a los usuarios visualizar la humedad del suelo y la cantidad de agua utilizada.
- El dispositivo no se podrá configurar por parte del usuario.

## 1.5. Metodología

**Objetivo 1:** “Diseñar un prototipo del dispositivo (Hardware)”

- Determinar las piezas necesarias para construir el dispositivo.
- Leer documentación de las piezas.
- Juntar todas las partes para formar el dispositivo.
- Probar el dispositivo armado.

**Objetivo 2:** “Diseñar un modelo que determinar cuando y como realizar un riego (Software)”

- Determinar las variables que se utilizarán.
- Desarrollo del modelo (Desarrollo de software).
- Probar el modelo en el dispositivo.

**Objetivo 3:** “Diseñar un prototipo de aplicación web y/o móvil para los usuarios”

- Diseñar mockup de aplicación web y/o móvil.
- Desarrollo de la aplicación.
- Pruebas de la aplicación.

## 2. Marco teórico

---

En este capítulo se hablará acerca en que consiste la programación del riego, los distintos tipos de suelos; que afectan en como se debe realizar un riego. Se describirá que es un servicio web y se verá el servicio que se utilizará para este proyecto mencionando las características por la cual fue escogido, finalmente se hará un breve análisis de los trabajos relacionados que existen actualmente en el mercado con una tabla comparativa y se verán los componentes tanto de hardware como de software.

### 2.1. Programación del riego

La programación del riego como se ve en la Figura 2.1, es la planificación de cuándo y qué cantidad de agua aplicar con el fin de mantener el crecimiento saludable del césped. Es una práctica de gestión diaria esencial.



Figura 2.1: Como y cuando regar

El momento adecuado del riego es una decisión crucial para la aplicación:

- Satisfacer las necesidades de agua, para evitar daños debido a la escasez de agua.
- Maximizar la eficiencia del uso del agua de riego y conservar los recursos hídricos locales.

Un riego eficaz sólo es posible con un seguimiento periódico de las condiciones del agua del suelo y el desarrollo de lo que se está cuidando (campos de cultivos o jardines) y con la previsión de las futuras necesidades de agua. Retrasar el riego hasta provoca estrés, o aplicar muy poca agua puede resultar en una pérdida sustancial. La aplicación de mucha agua se traducirá en costos adicionales y agua desperdiciada.

Existen varias herramientas que están disponibles para ayudar a un administrador el riego: sondas de suelo, sensores de humedad del suelo, estaciones meteorológicas, etc.

Para la programación del riego se utilizarán sensores de humedad del suelo y datos obtenidos de alguna estación meteorológica que entregará la probabilidad de lluvia en el lugar, cuando sea posible y a causa de que existe probabilidad de lluvia, la cantidad de agua de riego aplicada debe ser algo menor que el déficit de agua en el suelo con el fin de proporcionar el resto con la lluvia que caerá en el lugar.

Otra variable que se tendrá en cuenta, es el tipo de suelo en el que se encuentra ubicado el sistema de riego, ya que permitirá saber la capacidad de absorción que éste tiene, con el conocimiento de esta información se podrá saber cuándo comenzar un riego con el fin de utilizar el agua de forma eficiente.

### 2.1.1. Textura del suelo

La textura indica el contenido relativo de partículas de diferente tamaño, como la arena, el limo y la arcilla, en el suelo. La textura tiene que ver con la facilidad con que se puede trabajar el suelo, la cantidad de agua y aire que retiene y la velocidad con que el agua penetra en el suelo y lo atraviesa.

Para conocer la textura de una muestra de suelo [4]:

- Arcilloso: Se adhiere bastante, es fácilmente moldeable, las partículas no son visibles y la superficie brilla levemente.
- Limoso: Se adhiere a los dedos, se moldea con dificultad, los dedos dan apariencia grasosa y las partículas son brillantes.
- Arenoso: No se pega en los dedos, no se moldea como una masa y sus partículas individuales son visibles.

Cada uno de estos tipos de suelos presentan una propiedad llamada permeabilidad; que es la característica que tiene el suelo de transmitir el agua y el aire y es

una de las cualidades más importantes que han de considerarse a la hora de realizar un riego. Mientras más permeable sea el suelo, mayor será la filtración.

Por regla general, como se muestra en la Figura 2.2, mientras más fina sea la textura del suelo, más lenta sera la permeabilidad:

Suelo	Textura	Permeabilidad
Suelos arcillosos	Fina	De muy lenta a muy rápida
Suelos limosos	Moderadamente fina	
Suelos arenosos	Moderadamente gruesa	
	Gruesa	

Figura 2.2: Permeabilidad según textura del suelo

### 2.1.2. Como regar los suelos

#### Suelo arenoso

Para un suelo arenoso, se debe aplicar agua con más frecuencia que los suelos limosos o arcillosos. El agua drena rápidamente a través de la arena, por lo que no estarán disponibles cuando las plantas lo necesitan, por lo que hay que regar con poca cantidad pero con más frecuencia.

#### Suelo limoso

Para un suelo limoso, se debe puede aplicar agua constantemente durante un periodo de tiempo determinado. El agua drena con una velocidad moderada.

#### Suelo arcilloso

Los suelos arcillosos drenan mal el agua, debido a la pequeñez de sus partículas. Por ello se encharcan. Se debe aplicar agua con menor frecuencia que los suelos limosos y arenosos.

En la Figura 2.3 se puede apreciar la reacción de cada tipo de suelo cuando se le aplica agua.

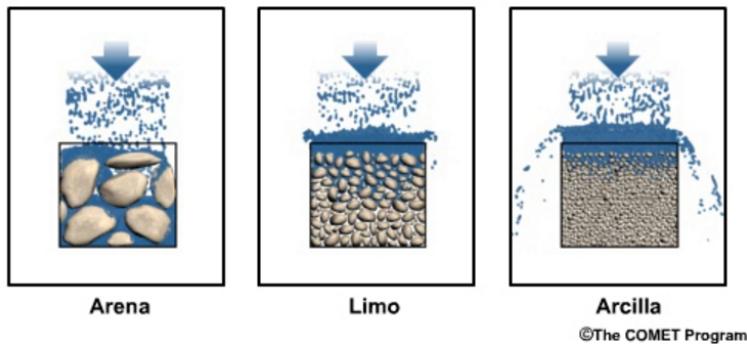


Figura 2.3: Variación en la infiltración por textura del suelo

## 2.2. Problemas identificados

Según Chuck Ingels [8], asesor agrícola de Extensión Cooperativa de la Universidad de California en el condado de Sacramento y Loren Oki, especialista de Extensión Cooperativa en el Departamento de Botánica de UC Davis y especialista en horticultura de paisajes, existen problemas en los sistemas de riego actuales que hoy en día son utilizados, ellos nombran principalmente cuatro problemas:

- “Apagar el sistema de riego por aspersores después de que llueve es el primer paso que se debe tomar para ahorrar agua”
  - “El riego de césped y jardines representa más de la mitad del agua usada en un hogar promedio”
  - “Ha llevado a cabo estudios sobre un problema común relacionado al riego de céspedes residenciales en California: el escurrimiento de agua”
  - “Mucha del agua que se aplica a los céspedes termina directamente en las alcantarillas. No solo se trata de agua desperdiciada; hemos descubierto también que el agua de escurrimientos arrastra contaminantes (incluyendo pesticidas y fertilizantes) hasta las alcantarillas y, eventualmente, a los canales y vías fluviales. El problema radica en que los sistemas de aspersores típicos aplican agua de forma más rápida que la que la tierra puede absorber, lo cual lleva al escurrimiento”

### 2.3. Información disponible

Para los problemas anteriormente mencionados existe información que será utilizado para planificar y determinar un riego, los podemos ver a continuación:

- Pronóstico del tiempo: se cuenta con esta información que permite saber en tiempo actual las condiciones climáticas de un punto en particular.
- Ubicación del dispositivo: se puede determinar la ubicación del dispositivo utilizando un GPS, si se utiliza en lugares con acceso a Internet en los que no se utilice el módem 3G, se podría determinar la ubicación a través de la dirección IP.
- Tipo de suelo: indica el tipo de suelo en el que el dispositivo está siendo utilizado, esta información se obtiene utilizando la ubicación del dispositivo.
- Humedad del suelo: indica la humedad del suelo utilizando sensores de humedad.

Existe más información que puede ser incluida para mejorar la calidad del riego aplicado al suelo, pero para este trabajo solo se considera las variables mencionadas anteriormente.

### 2.4. Sistema basado en reglas

En la vida diaria encontramos muchas situaciones complejas las que se rigen por reglas deterministas: sistemas de control de tráfico, sistemas de seguridad, transacciones bancarias, entre otros. Los sistemas basados en reglas [9] son una herramienta eficiente para tratar estos problemas.

Las reglas deterministas componen la más sencilla de las metodologías utilizadas en sistemas expertos. La base de conocimiento contiene las variables y el conjunto de reglas que definen el problema, y el motor de inferencia obtiene las conclusiones aplicando la lógica clásica a estas reglas. Por regla se entiende una proposición lógica que relaciona dos o más objetos e incluye dos partes, la premisa y la conclusión. Cada una de estas partes consiste en una expresión lógica con una o más afirmaciones objeto-valor conectadas mediante los operadores lógicos *y*, *o*, o *no*. Una regla se escribe normalmente como “Si premisa, entonces conclusión”.

Como ejemplo de problema determinista que puede ser formulado usando un conjunto de reglas, considérese una situación en la que un usuario (por ejemplo, un cliente) desea sacar dinero de su cuenta corriente mediante un cajero automático (CA). En cuanto el usuario introduce la tarjeta en el CA, la maquina la lee y la verifica. Si la tarjeta no es verificada con éxito (por ejemplo, porque no es legible), el CA devuelve la tarjeta al usuario con el mensaje de error correspondiente. En otro caso, el CA pide al usuario su número de identificación personal (NIP). Si el número fuese incorrecto, se dan tres oportunidades de corregirlo. Si el NIP es correcto, el CA pregunta al usuario cuánto dinero desea sacar. Para que el pago se autorice, la cantidad solicitada no debe exceder de una cierta cantidad límite diaria, además de haber suficiente dinero en su cuenta.

En este caso se tienen siete objetos, y cada objeto puede tomar uno y sólo un valor de entre sus posibles valores. La Figura 2.4 muestra estos objetos y sus posibles valores.

Objeto	Conjunto de posibles valores
Tarjeta	{verificada, no verificada}
Fecha	{expirada, no expirada}
NIP	{correcto, incorrecto}
Intentos	{excedidos, no excedidos}
Balance	{suficiente, insuficiente}
Límite	{excedido, no excedido}
Pago	{autorizado, no autorizado}

Figura 2.4: Objetos y posibles valores para el ejemplo del cajero automático

La Figura 2.5 muestra siete reglas que gobiernan la estrategia que el CA debe seguir cuando un usuario intenta sacar dinero de su cuenta.

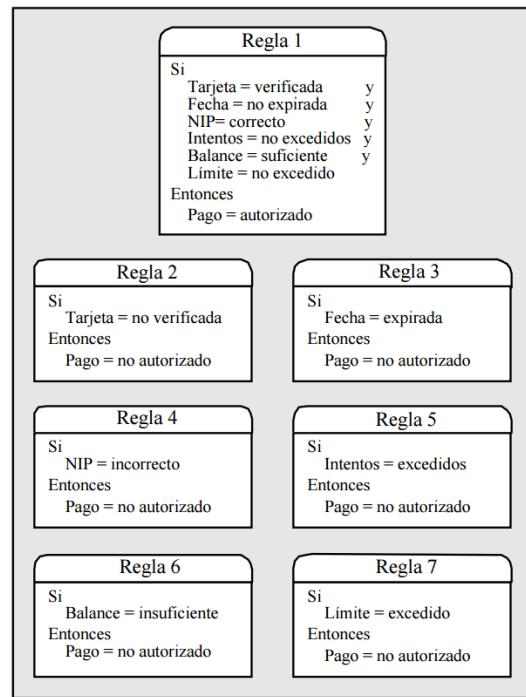


Figura 2.5: Ejemplos de reglas para sacar dinero de un cajero automático

Tal como se ha mencionado, hay dos tipos de elementos: los datos (hechos o evidencia) y el conocimiento (el conjunto de reglas almacenado en la base de conocimiento).

Para obtener conclusiones, los expertos utilizan diferentes tipos de reglas y estrategias de inferencia y control:

- Modus Ponens
- Modus Tollens

y las estrategias de inferencia:

- Encadenamiento de reglas
- Encadenamiento de reglas orientado a un objetivo

#### **2.4.1. Modus Ponens**

El Modus Ponens es quizás la regla de inferencia más comúnmente utilizada. Se utiliza para obtener conclusiones simples. En ella, se examina la premisa de la regla, y si es cierta, la conclusión pasa a formar parte del conocimiento. Por ejemplo, suponga que se tiene la regla, “Si A es cierto, entonces B es cierto” y que se sabe además que “A es cierto”. La regla Modus Ponens concluye que “B es cierto.”

#### **2.4.2. Estrategias de inferencia**

Una de las estrategias de inferencia más utilizadas para obtener conclusiones compuestas es el encadenamiento de reglas. Esta estrategia puede utilizarse cuando las premisas de ciertas reglas coinciden con las conclusiones de otras. Cuando se encadenan las reglas, los hechos pueden utilizarse para dar lugar a nuevos hechos, este proceso se repite sucesivamente hasta que no pueden obtenerse más conclusiones. El tiempo que consume este proceso hasta su terminación depende, por una parte, de los hechos conocidos, y, por otra, de las reglas que se activan.

Existen dos tipos de encadenamiento: progresivo y regresivo:

## Encadenamiento progresivo

Este tipo de encadenamiento se produce cuando el objetivo propuesto hace que se ejecute una regla, y la conclusión obtenida permite que se ejecute otra, y así sucesivamente hasta llegar a una respuesta, positiva o negativa. El punto final se detecta cuando no se pueden producir más encadenamientos.

## 2.5. Servicios web meteorológicos

### 2.5.1. ¿Qué es un servicio web?

Los servicios web son sistemas de software que permiten el intercambio de datos y funcionalidad entre aplicaciones sobre una red. Están soportados en diferentes estándares que garantizan la interoperabilidad de los servicios [10].

En la Figura 2.6, se puede apreciar la arquitectura de un servicio web, se ve que el cliente realiza una petición al servidor y éste le envía una respuesta con lo solicitado.

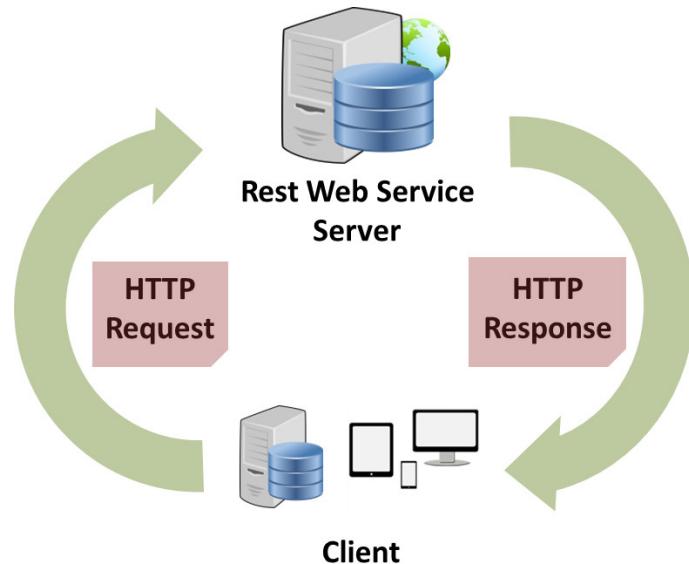


Figura 2.6: Arquitectura servicio web

Un servicio web REST, es un servicio implementado usando HTTP y que entrega como resultado en formato XML o JSON de los cuales se puede obtener la información solicitada.

### 2.5.2. Wunderground

Weather Underground es un servicio que entrega información meteorológica que ha desafiado las convenciones en torno a cómo la información del tiempo se comparte con el público desde 1993. Se ha creado con el fin de mejorar el acceso de las personas a los datos meteorológicos significativos de todo el mundo. Como servicio meteorológico primero de Internet, son considerados pioneros en su campo y están en constante búsqueda de nuevos conjuntos de datos y tecnologías que ayuden a compartir más datos con más gente.

Wunderground no da la posibilidad de consultar el clima dado las coordenadas de la posición en la que un punto se encuentra, estas coordenadas pueden ser obtenidas de forma automática utilizando un GPS o ingresándolas manualmente. Para utilizar el servicio, es necesario registrarse para obtener un llave que se utiliza para realizar las consultas meteorológicas.

Las características de este servicio web y las razones por las que se decidió utilizarlo son las siguientes:

- Las consulta para obtener la información meteorológica son gratuitas.
- Se adapta perfectamente a las necesidades del proyecto, tanto en los datos requeridos como en la forma de solicitar éstos.
- Se pueden realizar hasta 24 consultas por día.
- Los resultados obtenidos son bastante parecidos entre lo que entrega Google y Wunderground como se puede ver en la Figura: 2.7 y para fines de este proyecto es suficiente (Suponiendo que Google entrega resultados confiables).

Para obtener más información y/o registrarse para utilizar el servicio visitar la página web.<sup>1</sup>

---

<sup>1</sup><http://www.wunderground.com/weather/api/d/docs?d=index>

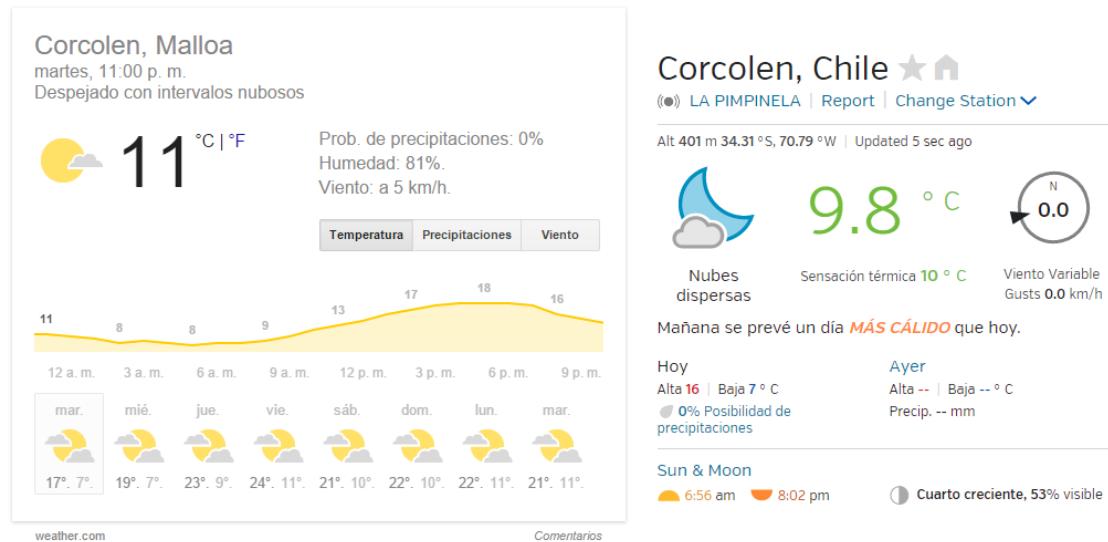


Figura 2.7: Comparación servicio meteorológico de Google v/s Wunderground para la localidad rural de Corcolén

## 2.6. Trabajos relacionados

A continuación se mencionan algunos de los dispositivos que se utilizan actualmente para automatizar el riego de los jardines.

### 2.6.1. Cultivar

Cultivar está desarrollando y fabricando por Raincloud [2]. Raincloud administra de forma conveniente e inteligente la gestión del agua, con un sistema de riego conectado a la web que permite monitorear en todo momento el estado del sistema. Raincloud vincula los dispositivos móviles permitiendo activar válvulas de agua y consultar sensores de humedad utilizando Wi-Fi.

Ventajas:

- Utiliza sensores de humedad de suelo.
- Se controla a través de un teléfono móvil.
- Consumo la menor cantidad de agua posible.
- Almacena los datos en la nube.

- Entrega estadísticas del consumo de agua.

Desventajas:

- Necesita configuración previa.
- Se necesita Wi-Fi en el hogar para su óptimo funcionamiento.
- Tiene un costo muy elevado (\$344.000 pesos).
- No toma en cuenta datos meteorológicos.

### 2.6.2. Rachio IRO

Iro [14] es un controlador de riego inteligente que permite a las personas de manera fácil, mantener sus jardines en buen estado. Se integra de manera fácil al Wi-Fi de su hogar, entregando un completo control a través de su computador o teléfono móvil.

Rachio se ajusta automáticamente a los datos meteorológicos del lugar en el que se encuentra, estación del año, tipo de zona y características de la región.

Ventajas:

- Se controla a través de un teléfono móvil o una computador.
- Puede funcionar de forma automática.
- Se ajustara automáticamente a el clima.
- Consume la menor cantidad de agua posible.
- No necesita programarse.
- Entrega estadísticas del consumo de agua.

Desventajas:

- No utiliza sensores de humedad de suelo.
- Se necesita Wi-Fi en el hogar para su óptimo funcionamiento.
- Tiene un costo muy elevado (\$180.000 pesos).

### 2.6.3. Rain Bird

El Programador 6 Estaciones Digital Rain Bird [15] (temporizador)logra, gracias a su particular panel con diversas opciones, controlar y personalizar tu sistema de riego. Si bien tiene claras opciones para potenciar el jardín, este producto exterioriza sus habilidades para centrarse en distintos planos y superficies. Con el fin de que tus zonas verdes comprendan el mismo cuidado que el interior de tu casa, este producto te da total control de lo que deseas lograr, entregándote oportunidades específicas de acción que se ajustan a las necesidades diarias.

Entre las características del Programador 6 Estaciones Digital Rain Bird se puede encontrar su tablero altamente definido que te da la posibilidad, de manera sencilla, de regular y hacer mejor uso del suministro de agua, debido al despliegue uniforme y sistemático de las acciones de los riegos. Al poseer un esquema de operación para tu jardín, podrás lograr una constancia en los procesos de crecimiento de tu extensa flora, demostrando que la sencillez de este producto es una cualidad para considerar.

Ventajas:

- Programación basada en zonas, que permite asignar programas independientes a cada zona.
- Puede funcionar de forma automática.

Desventajas:

- No utiliza sensores de humedad de suelo.
- No tiene control remoto.
- No entrega información del consumo de agua.
- No tiene información meteorológica, por lo que puede hacer un riego cuando esté lloviendo.
- Tiene un costo muy elevado (\$65.990 pesos).

### 2.6.4. Tabla comparativa

En la Figura: 2.8 se muestra las características de cada uno de los dispositivos que actualmente se utilizan en el mercado y además se incluye el trabajo que se esta

realizando en esta memoria. Las características descritas en la tabla son necesarias para realizar un riego eficiente y se puede ver que los dispositivos incluyen algunas de estas, pero no están todas incluidas en uno solo y es lo que se espera lograr con este trabajo.

También se puede apreciar que ningún dispositivo toma en cuenta el tipo de suelo que es regando, una variable a considerar si se desea maximizar el uso del agua.

	Automatización de riego	Aplicación Web/Móvil	Consulta datos Meteorológicos	Detección de tipo suelo	Almacenamiento de datos	Sensores de Humedad
Cultivar	✓	✓	✗	✗	✓	✓
Rachio IRO	✓	✓	✓	✗	✓	✓
Rain Bird	✓	✗	✗	✗	✗	✗
Memoria	✓	✓	✓	✓	✓	✓

Figura 2.8: Tabla Comparativa

## 2.7. Hardware

Todas las partes físicas de un sistema que tiene componentes: eléctricos, electrónicos, electromecánicos y mecánicos. Para la construcción del sistema de riego se necesitarán una serie de componentes, sensores, actuadores y computadores para su correcto funcionamiento.

### 2.7.1. Raspberry pi

Raspberry Pi [7] es un microcomputador del tamaño de una tarjeta de crédito a bajo costo que se conecta a un monitor o un televisor, utiliza un teclado y un ratón estándar. Es un dispositivo que permite a las personas de todas las edades explorar la computación y aprender a programar en lenguajes como Python. Es capaz de hacer todo lo que espera de una computadora de escritorio, desde navegar por Internet y reproducir videos en alta definición, trabajar en hojas de cálculo, procesadores de texto, y jugar.

Raspberry Pi tiene la capacidad de interactuar con el mundo exterior, y se ha utilizado en una amplia gama de proyectos digitales, máquinas de música y detectores en las estaciones meteorológicas.

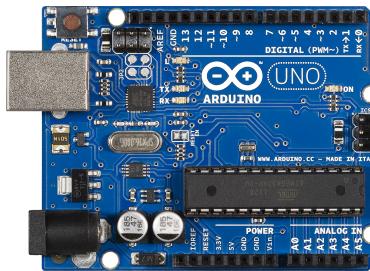
La función de la Raspberry es obtener toda la información de los sensores a través del arduino, obtener datos meteorológicos, obtener los tipos de suelo y utilizando toda esta información determinar cuándo se debe realizar un riego, activando un relé. También envía la información de humedad de suelo y agua utilizada a un servidor.



Figura 2.9: Raspberry Pi modelo B+

### 2.7.2. Arduino

Arduino [1] es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, como hobby y para cualquiera interesado en crear objetos o entornos interactivos. Arduino puede sentir el entorno mediante la recepción de entradas desde una variedad de sensores y puede afectar a su alrededor mediante el control de luces, motores y otros artefactos.



sobre millones de elementos a los que acceden al mismo tiempo miles de usuarios. El servidor gestiona las actualizaciones de datos, permite el acceso simultáneo de muchos servidores o usuarios web y garantiza la seguridad y la integridad de los datos.

Así como sus funciones básicas, el software de servidores de bases de datos ofrece herramientas para facilitar y acelerar la administración de bases de datos. Algunas funciones son la exportación de datos, la configuración del acceso de los usuarios y el respaldo de datos.

Se utilizará PostgreSQL como sistema de gestión de base de datos que es libre y tiene un complemento llamado PostGIS que añade soporte de objetos geográficos a la base de datos, esta característica será utilizada en conjunto con el GPS determinar el tipo de suelo en el que el dispositivo se encuentra.

#### 2.7.4. Servidor web

Básicamente, un servidor web sirve contenido estático a un navegador, carga un archivo y lo sirve a través de la red al navegador de un usuario. Este intercambio es mediado por el navegador y el servidor que hablan el uno con el otro mediante HTTP.

Su función en este proyecto es almacenar todos los datos que envía el dispositivo. También se almacenará una aplicación web para que los usuarios puedan consultar información relacionada al riego.

#### 2.7.5. Sensores de humedad

Existen varios tipos de sensores de humedad [16]

- Mecánicos: aprovechan los cambios de dimensiones que sufren cierto tipos de materiales en presencia de la humedad.
- Basados en sales higroscópicas: deducen el valor de la humedad en el ambiente a partir de una molécula cristalina que tiene mucha afinidad con la absorción de agua.
- Por conductividad: la presencia de agua en un ambiente permite que a través de unas rejillas de oro circule una corriente. Ya que el agua es buena conductora de corriente. Según la medida de corriente se deduce el valor de la humedad.

- Capacitivos: se basan sencillamente en el cambio de la capacidad que sufre un condensador en presencia de humedad.
- Infrarrojos: estos disponen de 2 fuentes infrarrojas que lo que hacen es absorber parte de la radiación que contiene el vapor de agua.
- Resistivos: aplican un principio de conductividad de la tierra. Es decir, mientras más agua hay en la muestra, más alta es la conductividad de la tierra.

En este proyecto se utilizarán sensores resistivos, ya que son baratos y se pueden encontrar en cualquier parte.

La desventaja es que su tiempo de vida útil es corto.

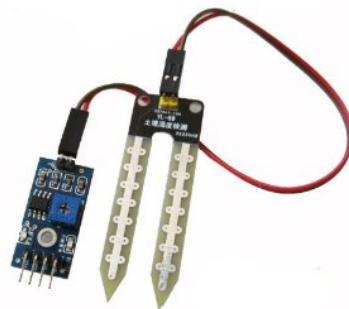


Figura 2.11: Sensor de humedad

#### 2.7.6. Sensor de flujo de agua

Un sensor de flujo de agua consiste en una válvula de cuerpo plástico, un rotor de agua y un sensor de efecto hall. Cuando el agua fluye a través del rotor, el rotor gira. Su velocidad cambia con diferentes ritmos de flujo. El sensor de efecto hall tiene como salida la señal de pulso correspondiente.

Será utilizado para medir la cantidad de agua utilizada en riego para que los usuarios manejen esta información.



Figura 2.12: Sensor de flujo de agua

#### 2.7.7. Relé

Un relé de estado sólido es un dispositivo interruptor electrónico que commuta el paso de la electricidad cuando una pequeña corriente es aplicada en sus terminales de control. Los relé de estado sólido consisten en un sensor que responde a una entrada apropiada (señal de control), un interruptor electrónico de estado sólido que commuta el circuito de carga, y un mecanismo de acoplamiento a partir de la señal de control que activa este interruptor sin partes mecánicas. El relé puede estar diseñado para commutar corriente alterna o continua. Hace la misma función que el relé electromecánico, pero sin partes móviles.

La principal ventaja de utilizar un relé de estado sólido es que utilizan una menor tensión de trabajo, se activan desde 1,5V o menos lo que lo hace ideal para trabajar con la Raspberry, que sus salidas de corrientes son de 3V - 5V.

El relé es utilizado como interruptor para activar la válvula solenoide. No se usa la Raspberry directamente ya que para activar la válvula se necesitan 12V.



Figura 2.13: Relé de estado sólido

### 2.7.8. Válvula solenoide

Una válvula solenoide está diseñada para controlar el paso de un fluido por un conducto o tubería. La válvula se mueve mediante una bobina solenoide. Generalmente no tiene más que dos posiciones: abierto y cerrado, o todo y nada. Las electroválvulas se usan en multitud de aplicaciones para controlar el flujo de todo tipo de fluidos.

La válvula se activa al aplicarle un voltaje de 12V.

Se utilizará para permitir el paso del agua al sistema de riego del jardín.



Figura 2.14: Válvula solenoide

### 2.7.9. GPS

GPS [12] es un sistema que tiene como objetivo la determinación de las coordenadas espaciales de puntos respecto de un sistema de referencia mundial. Los puntos pueden estar ubicados en cualquier lugar del planeta, pueden permanecer estáticos o

en movimiento y las observaciones pueden realizarse en cualquier momento del día. Para la obtención de coordenadas el sistema se basa en la determinación simultánea de las distancias a cuatro satélites (como mínimo) de coordenadas conocidas. Estas distancias se obtienen a partir de las señales emitidas por los satélites, las que son recibidas por receptores especialmente diseñados. Las coordenadas de los satélites son provistas al receptor por el sistema.

El GPS se utilizará para determinar la ubicación exacta en la que el dispositivo se encuentra instalado, con esta información se determinará el tipo de suelo y se aplicará el riego más conveniente.



Figura 2.15: Módulo GPS USB

#### 2.7.10. Módem 3G

Módem 3G es un terminal de red inalámbrica basada en la tecnología CDMA2000. Es la mejor opción para los usuarios que necesitan acceder a Internet desde cualquier lugar en cualquier momento. El módem está conectado al micro computador mediante una interfaz USB.

El módem se utilizará para comunicar el dispositivo con el servidor donde se almacenarán los datos, consultar los datos meteorológicos y obtener el tipo de suelo.



Figura 2.16: Módem 3G

### 2.7.11. Cargador

Un cargador es un dispositivo utilizado para suministrar corriente eléctrica a un dispositivo, que en este caso se tratará de la Raspberry Pi. El cargador debe funcionar a 5V y entregar una corriente de 700mA.

Será utilizado para entregarle energía a la Raspberry para que pueda funcionar correctamente en conjunto con los sensores y válvulas.



Figura 2.17: Cargador

### 2.7.12. Costo componentes

En el Cuadro 2.1 se detallan los costos de cada componente necesario para la construcción, esto tiene un total de \$ 50.110 pesos que es lo que cuesta construir el prototipo.

Cuadro 2.1: Tabla de precios

Componente	Precio (pesos)
Raspberry Pi	\$ 28.400
Arduino	\$ 3.500
Sensor de Humedad de Suelo	\$ 700
Sensor de Flujo de Agua	\$ 2.700
Relé de Estado Solido	\$ 3.500
Valvula Solenoide	\$ 3.500
Gps	\$ 7.100
Cargador	\$ 710
<b>Total</b>	<b>\$ 50.110</b>

## 2.8. Software

### 2.8.1. Raspbian

Raspbian [6] es un sistema operativo libre basado en Debían optimizado para el hardware Raspberry Pi. Raspbian no tiene relación con la Fundación Raspberry Pi, fue creado por un pequeño y dedicado equipo de desarrolladores que son fans del hardware Raspberry Pi<sup>2</sup>.

Raspbian ofrece más que un sistema operativo; viene con más de 35.000 paquetes, software pre-compilado en un formato que hace más fácil la instalación en su Raspberry Pi que permiten hacer muchas cosas diferentes en distintos ámbitos .

### 2.8.2. Python

Python es un claro y poderoso lenguaje de programación orientado a objetos, comparable con Perl, Ruby o Java.

Algunas de las características de Python<sup>3</sup> son:

- Python ideal para el desarrollo de prototipos y otras tareas de programación, sin comprometer la capacidad de mantenimiento.
- Utiliza una sintaxis clara, por lo que los programas son fáciles de leer.
- Viene con una biblioteca estándar que soporta muchas tareas comunes de programación, tales como la conexión a los servidores web, la búsqueda de texto con expresiones regulares, leer y modificar archivos.
- Es software libre.
- Tiene librerías que facilitan el uso de la interfaz GPIO de la Raspberry Pi.

### Librería GPIO

Una de las características de gran alcance de la Raspberry Pi es la fila de GPIO [5] (Entrada/Salida de Propósito General) alfileres a lo largo del borde de la placa, junto a la salida de vídeo socket amarilla como se puede apreciar en la Figura 2.18:

---

<sup>2</sup><https://www.raspbian.org/RaspbianInstaller>

<sup>3</sup><https://wiki.python.org/moin/BeginnersGuide/Overview>

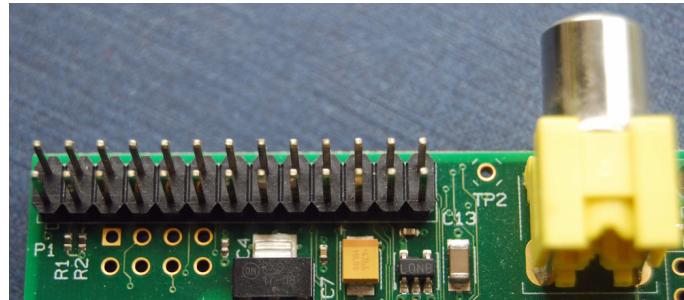


Figura 2.18: GPIO Raspberry Pi

Estos pines son una interfaz física entre el la Raspberry y el mundo exterior. Al nivel más simple, se puede pensar en ellos como interruptores que puede activar o desactivar ya sean las entradas como las salidas. Diecisiete de los veinte y seis pines son GPIO; los otros son pines de alimentación o de tierra como se ve en la Figura 2.19:

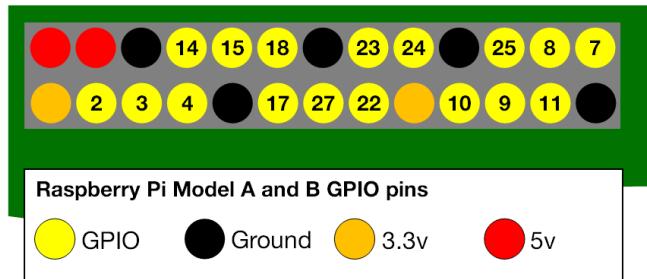


Figura 2.19: Pines GPIO

¿Qué se puede hacer con esto?

Se puede programar los pines de maneras asombrosas para interactuar con el mundo real. Las entradas no tienen que venir de un interruptor físico; podría ser la entrada de un sensor o una señal de otro ordenador o dispositivo, por ejemplo. La salida también puede hacer cualquier cosa, desde encender un LED con el envío de una señal o de datos a otro dispositivo. Si la Raspberry está en una red, puede controlar los dispositivos que están conectados a él desde cualquier lugar. Conectividad y control de los dispositivos físicos a través de Internet es algo muy poderoso y emocionante, y la Raspberry es ideal para esto.

# **3. Desarrollo**

---

En este capítulo se mostrarán los pasos necesarios para construir y hacer funcionar el dispositivo de riego. Primero veremos todo lo relacionado con el hardware juntando todas las partes, luego se trabajará en el desarrollo del modelo que corresponde a la parte de software del dispositivo. También se mostrará brevemente el desarrollo de la aplicación web y finalmente veremos las variables necesarias para determinar un helada y como se hará para notificar a los usuarios.

## **3.1. Hardware**

Raspberry es una plataforma muy útil para el desarrollo de aplicaciones domóticas y de control. Sin embargo. Raspberry tiene una limitación en cuanto a entradas/salidas disponibles y no puede trabajar con modulación por ancho de pulsos.

Una solución a este problema puede ser utilizar un Beaglebone black o se puede combinar el Raspberry con Arduino y dejar todo el control de hardware al Arduino y solo utilizar la Raspberry como controlador. En esta sección se mostrará como conectar la Raspberry con un Arduino a través de USB a través de comunicación serie.

### **3.1.1. Arduino**

#### **Programación Arduino**

Lo primero que se hará es crear un programa para Arduino que leerá la información de un sensor de humedad y la información que envía un sensor de flujo

utilizando el puerto serie del Arduino. El programa que utilizamos se muestra a continuación, para ello se debe instalar el IDE de Arduino<sup>1</sup>

Instalado el programa, conectar al Arduino a la computadora, ejecutar el programa y ya se está listo para trabajar.

En el Listing 3.1 se muestra el código con el cual se puede leer información desde los distintos tipos de sensores utilizados.

Listing 3.1: Código Arduino

```

volatile int frecuencia_sensor; // Mide los pulsos de flujo
unsigned int litrosHora; // Calcula los litros por hora
unsigned int humedad_sensor; // Mide la humedad del suelo
unsigned char pinFlujo = 2; // Pin de flujo
unsigned char pinHumedad = A0; // Pin de humedad
unsigned long tiempoActual;
unsigned long tiempoCiclo;

void flujo () // Interruot function
{
    frecuencia_sensor++;
}

void setup () {
    pinMode(pinFlujo, INPUT);
    Serial.begin(9600);
    attachInterrupt(0, flujo, RISING); // Setup Interrupt
    sei(); // Enable interrupts
    tiempoActual = millis();
    tiempoCiclo = tiempoActual;
}

void loop () {
    humedad_sensor = analogRead(pinHumedad);
    //Serial.print(humedad_sensor);
    //Serial.println(" Humedad");
    tiempoActual = millis();
    //Cada segundo, calcula e imprime litros por hora
    if(tiempoActual >= (tiempoCiclo + 1000))

```

---

<sup>1</sup><https://www.arduino.cc/en/Main/Software>

```

{
    tiempoCiclo = tiempoActual; // Actualiza tiempoCiclo
    litrosHora = (frecuencia_sensor * 60 / 7.5); // (Pulse frequency x
        60 min) / 7.5Q = L/hour
    frecuencia_sensor = 0; // Resetea el contador
    //Serial.print(litrosHora); // Imprime litro por hora
    //Serial.println(" L/hour");
    //Serial.print(humedad_sensor);
    //Serial.println(" Humedad");
    Serial.println(String(humedad_sensor) + " " + String(litrosHora));
}
}

```

Una vez escrito el código en el IDE, se verifica y se sube al Arduino. Es importante destacar que para enviar información a las Raspberry se utiliza la linea de código `Serial.println("mensaje")`.

### Conexión de sensores

Como se ha mencionado anteriormente, los sensores que se utilizarán son de humedad de suelo y flujo de caudal, y se mostró el código necesario para que estos funcionen. De acuerdo a ese código se es que se realizó la conexión de los sensores al Arduino, se puede apreciar claramente en la Figura 3.1:

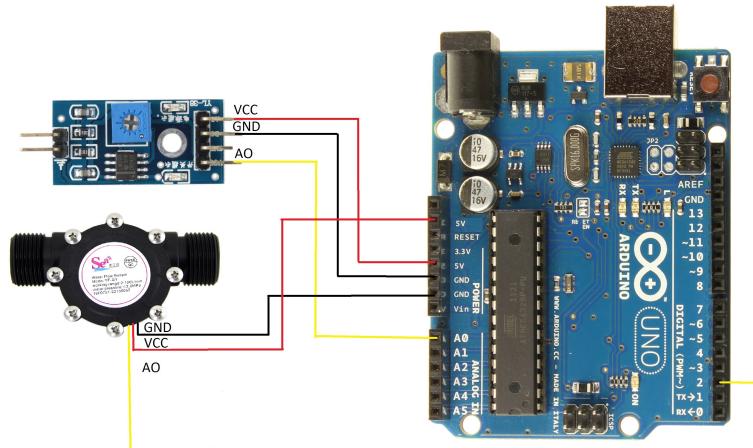


Figura 3.1: Conexión sensores en Arduino

VCC: Alimentación positiva del circuito.

GND: Alimentación negativa del circuito.

AO: Salida analógica del sensor o válvula.

### 3.1.2. Raspberry

Para configurar la Raspberry lo primero que hay que hacer es instalar las librerías para el control del puerto serie desde Python. Para hacer esto ejecutamos el siguiente comando desde la consola en nuestra Raspberry:

```
sudo apt-get install python-serial
```

Con esto instalado ya esta lista la Raspberry para comunicarse al Arduino utilizando el puerto serie. Lo que hay que determinar es que puerto utiliza el Arduino cuando esta conectado al Raspberry. Para saber el puerto primero se conecta el Arduino a la Raspberry por USB y luego se ejecuta el siguiente comando:

```
ls /dev/tty*
```

Una vez ejecutado el comando se verá un listado de diferentes dispositivos. El Arduino en este caso aparece en el último lugar llamado /dev/ttyUSB0 como se ve en la Figura 3.2:

```
pi@raspberrypi: ~/Desktop $ ls /dev/tty*
/dev/tty      /dev/tty20  /dev/tty33  /dev/tty46  /dev/tty59
/dev/tty0     /dev/tty21  /dev/tty34  /dev/tty47  /dev/tty6
/dev/tty1     /dev/tty22  /dev/tty35  /dev/tty48  /dev/tty60
/dev/tty10    /dev/tty23  /dev/tty36  /dev/tty49  /dev/tty61
/dev/tty11    /dev/tty24  /dev/tty37  /dev/tty5   /dev/tty62
/dev/tty12    /dev/tty25  /dev/tty38  /dev/tty50  /dev/tty63
/dev/tty13    /dev/tty26  /dev/tty39  /dev/tty51  /dev/tty7
/dev/tty14    /dev/tty27  /dev/tty4   /dev/tty52  /dev/tty8
/dev/tty15    /dev/tty28  /dev/tty40  /dev/tty53  /dev/tty9
/dev/tty16    /dev/tty29  /dev/tty41  /dev/tty54  /dev/ttyAMA0
/dev/tty17    /dev/tty3   /dev/tty42  /dev/tty55  /dev/ttyprintk
/dev/tty18    /dev/tty30  /dev/tty43  /dev/tty56  /dev/ttyUSB0
/dev/tty19    /dev/tty31  /dev/tty44  /dev/tty57
/dev/tty2     /dev/tty32  /dev/tty45  /dev/tty58
pi@raspberrypi: ~/Desktop $
```

Figura 3.2: Arduino en Raspberry

Una vez que el Arduino esta identificado ya podemos comenzar a comunicarnos mediante un programa en Python. En este caso se mostrará un ejemplo de como recibir la información en la Raspberry enviada desde el Arduino, en la Figura 3.3 se puede ver el código utilizado.



The screenshot shows a terminal window titled "pi@raspberrypi: ~/Desktop" running "GNU nano 2.2.6". The file being edited is "arduino.py". The code in the editor is:

```
from serial import Serial

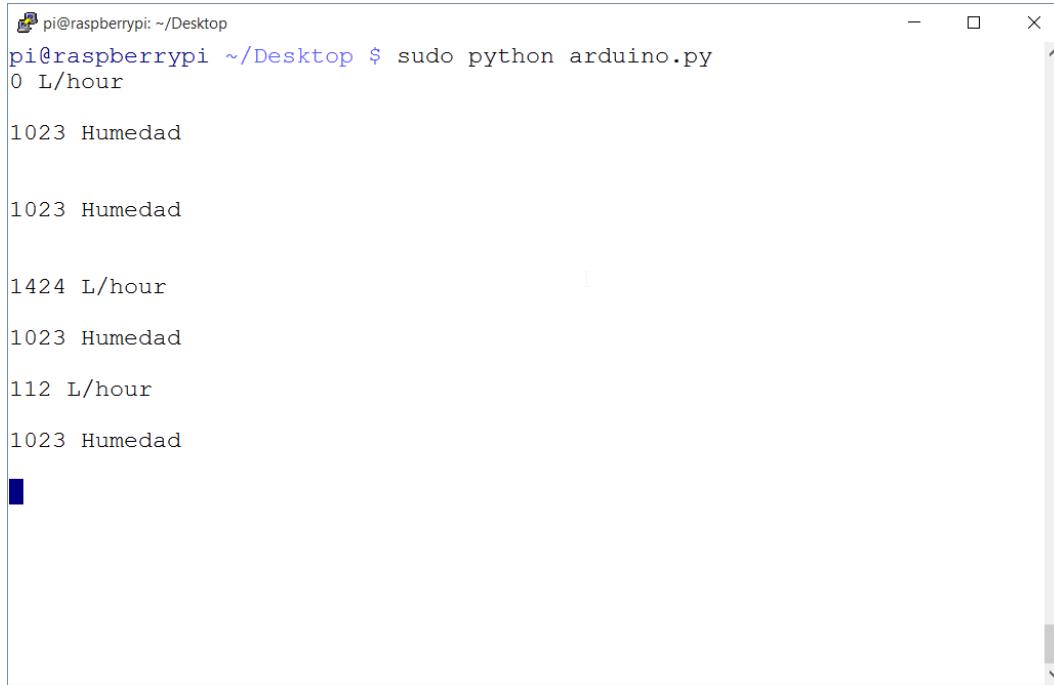
ardu = Serial('/dev/ttyUSB0', 9600, timeout=1)
while True:
    try:
        a = ardu.readline()
        print a
    except KeyboardInterrupt:
        break
ardu.close()
```

The terminal window has a menu bar at the bottom with the following options:

- File: arduino.py
- Get Help
- WriteOut
- Read File
- Prev Page
- Cut Text
- Cur Pos
- Exit
- Justify
- Where Is
- Next Page
- UnCut Tex
- To Spell

Figura 3.3: Comunicación Raspberry Arduino

Con este programa en Python ya se puede recibir datos desde el Arduino en la Raspberry, para probarlo se ejecuta el programa con la siguiente linea de código *sudo python nombre\_programa.py*. Los resultados obtenido en la Figura 3.4

A screenshot of a terminal window titled "pi@raspberrypi: ~/Desktop". The window contains the following text:

```
pi@raspberrypi:~/Desktop$ sudo python arduino.py
0 L/hour
1023 Humedad
1023 Humedad
1424 L/hour
1023 Humedad
112 L/hour
1023 Humedad
```

The terminal has a standard Linux-style interface with a title bar, a scroll bar on the right, and a dark background.

Figura 3.4: Recibo de datos desde Arduino en Raspberry

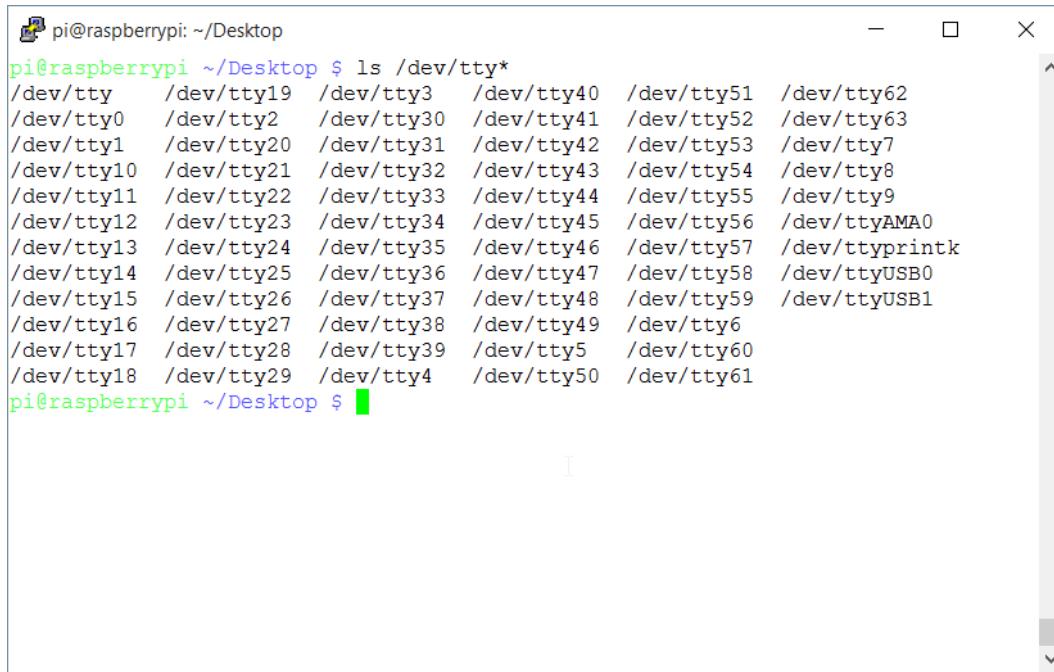
### 3.1.3. GPS

El GPS es una parte importante del dispositivo, ya que determina la posición en la que se aplicará el riego, gracias a esto se puede determinar el tipo de suelo en el que el dispositivo se encuentra y realizar un riego especial a cada lugar.

El GPS se conectará a la Raspberry ya que utiliza conexión USB.

Pasos para el funcionamiento del GPS en la Raspberry:

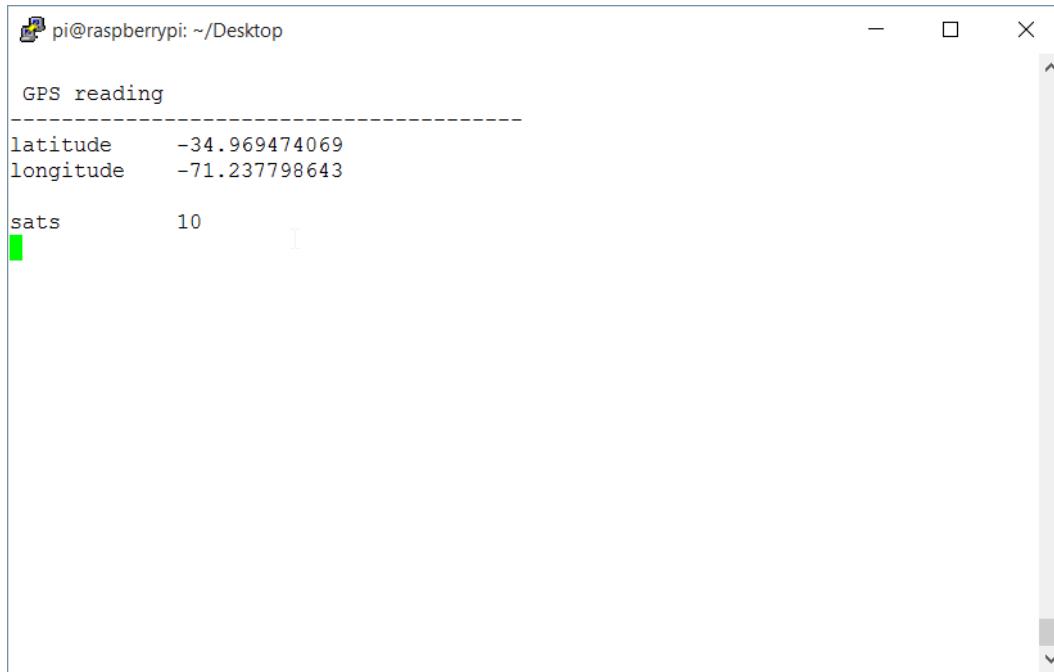
- Conectar el GPS a un puerto USB de la Raspberry.
- En la terminal de Raspberry ejecutar *ls /dev/tty\** como se ve en la Figura 3.5:



```
pi@raspberrypi: ~/Desktop $ ls /dev/tty*
/dev/tty      /dev/tty19  /dev/tty3    /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty0     /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/ttyUSB0
/dev/tty15    /dev/tty26  /dev/tty37   /dev/tty48  /dev/tty59  /dev/ttyUSB1
/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: ~/Desktop $
```

Figura 3.5: GPS en Raspberry, /dev/ttyUSB1

- *sudo gpsd -N -D3 /dev/ttyUSB1*
- *sudo nano gps.py*
- En el archivo creado anteriormente copiar el código del siguiente link:  
<https://gist.github.com/wolfg1969/4653340#file-gpsddata-py>
- Ejecutar *sudo python gps.py* con esto obtenemos como resultado lo que se aprecia en la Figura 3.6:

A screenshot of a terminal window titled "pi@raspberrypi: ~/Desktop". The window displays a "GPS reading" message followed by a dashed line. Below the line, it shows the following data:

```
latitude      -34.969474069
longitude     -71.237798643
sats          10
```

A small green progress bar is visible on the left side of the terminal window.

Figura 3.6: Datos GPS

## 3.2. Software

### 3.2.1. Información meteorológica

La estructura de la url para realizar la consulta de la siguiente:

`http://api.wunderground.com/api/e7d949410a7481dc/forecast10day/lang:SP/q/-34.40819094134256,-71.00087251514196.json`

- `http://api.wunderground.com/api/` es la dirección donde se hace la consulta.
- `e7d949410a7481dc` es la llave que se obtiene al registrarse y es necesaria para utilizar el servicio.
- `forecast10day` retorna datos meteorológicos de los siguientes 10 días.
- `lang:SP` se utiliza para obtener los datos en español.
- `q/-34.40819094134256,-71.00087251514196` son las coordenadas del lugar que se quieren obtener los datos.

- .json es el formato en el que se devuelve la información, también puede ser .xml.

Los datos obtenidos se pueden apreciar en la Figura 3.7 y la información más relevante es “pop” que informa de la probabilidad de lluvia que hay en el lugar.

```
{
  "period":0,
  "icon":"clear",
  "icon_url":"http://icons.wxug.com/i/c/k/clear.gif",
  "title":"Martes",
  "fcttext":"Cielo mayormente despejado. Mínima de 51 F.",
  "fcttext_metric":"Cielo mayormente despejado. Mínima de 10 C.",
  "pop":"0"
},
{
  "period":1,
  "icon":"nt_clear",
  "icon_url":"http://icons.wxug.com/i/c/k/nt_clear.gif",
  "title":"Noche del martes",
  "fcttext":"Cielo mayormente despejado. Mínima de 51 F. Vientos del OSO de 5 a 10 milla/h.",
  "fcttext_metric":"Cielo mayormente despejado. Mínima de 10 C. Vientos del OSO de 10 a 15 km/h.",
  "pop":"0"
}
```

Figura 3.7: Respuesta del servicio wunderground al hacer una consulta

### 3.2.2. Modelo de riego

La principal función del modelo de riego es determinar cuando hacer un riego en base a los datos que tiene disponible, también indicará que tipo de riego se debe hacer en base al tipo de suelo en el que el dispositivo se encuentra instalado. Esto se hará mediante un sistema basado en reglas que ya fue explicado anteriormente.

¿Cuando se ejecutará el modelo? El modelo de riego se ejecutará cada mañana a las 6 a.m y el tiempo de riego será de 20 min. La hora fue escogida para evitar la evaporación del agua y daños que se le puedan dañar el césped a causa del sol. El código que realiza esta acción se ve en el Listing 3.2:

Listing 3.2: Control de riego

```

while True:
    hora = datetime.datetime.now().time()
    if (hora>=datetime.time(6,00) and hora<=datetime.time(6,20)):
        if suelo == "arenoso":
            #RiegoArenoso
        else if suelo == "limoso":
            #RiegoLimoso
        else:
            #RiegoArcilloso

```

Los datos disponibles para determinar un riego son los que se aprecian en el Cuadro 3.1, también se pueden ver los valores asociados a cada objeto.

Cuadro 3.1: Objetos y posibles valores para determinar un riego

Objeto	Conjunto de posibles valores
Humedad de Suelo	{ 0 - 1024 }
Tipo de Suelo	{ Arcilloso, Limoso, Arenoso }
Probabilidad de lluvia	{ 0 - 100 }
Riego	{ RiegoArcilloso, RiegoLimoso, RiegoArenoso, NoRegar }

A partir de estos datos se generaron reglas que determinan cuando realizar un riego.

Para generalizar un poco las reglas del modelo a ha creado una tabla con las valores predefinidos como se ve en el Cuadro 3.2:

Cuadro 3.2: Variables predefinidas para la reglas del modelo

Valores
Humedad = H = 512
Probabilidad de lluvia = P = 70 %

Como esto es un prototipo, estos valores se pueden modificar para mejorar la calidad del riego, esto se podría hacer consultando a un experto en el área de riego de suelo y dejarlos fijos sin que vuelvan a cambiar. Otra opción es utilizar “elicitation technique”; que es una técnica usada para obtener datos y reunir conocimiento o información desde las personas, por lo que al momento de que el dispositivo esté instalado se pueden configurar estas variables, dejando a las personas que tomen la decisión que consideren más apropiada para su jardín, esto también podría ser

Cuadro 3.3: Reglas modelo, cuando y como regar

<b>Regar</b>
If Humedad <H and Probabilidad <P and Suelo == “Arcillo” then RiegoArcilloso,
If Humedad <H and Probabilidad <P and Suelo == “Limoso” then RiegoLimoso,
If Humedad <H and Probabilidad <P and Suelo == “Areno” then RiegoArenoso
<b>NO regar (Alta probabilidad de lluvia)</b>
If Humedad <H and Probabilidad >P then No regar
<b>NO regar (Humedad de suelo suficiente)</b>
If Humedad >H then No regar

modificado a través del tiempo para ir ajustándolo gradualmente y obtener el mayor beneficio posible (ahorro de agua v/s césped verde).

Incluso el dispositivo podría aprender ajustando sus valores automáticamente como puede ser el caso de la probabilidad de lluvia. También se puede automatizar la humedad del suelo preguntándole periódicamente al usuario el estado del césped para ajustar el tiempo de riego.

Estas reglas se pueden mejorar y aumentar su complejidad a medida que se agreguen nuevas variables al modelo, utilizando eficientemente el agua sin afectar la calidad del riego, en el Cuadro 3.3 se pueden apreciar las reglas que se definieron para el modelo de riego.

Gráficamente estas reglas se pueden ver como árbol tal como lo muestra la Figura 3.8, además queda claro que nunca se dará el caso en el que pueda haber como resultado dos tipos de riego diferentes, esto se debe a que el espacio de soluciones existentes es finito. El código fuente del modelo de riego se puede encontrar en el Anexo 1.

### 3.2.3. Servicio web de suelo

Para determinar el tipo de suelo en el que el dispositivo se encuentra, se creó un servicio web que dado las coordenadas geográficas entregará la información requerida.

El servicio consta de dos partes:

- El dispositivo determina el tipo de suelo y lo almacena en la base de datos.

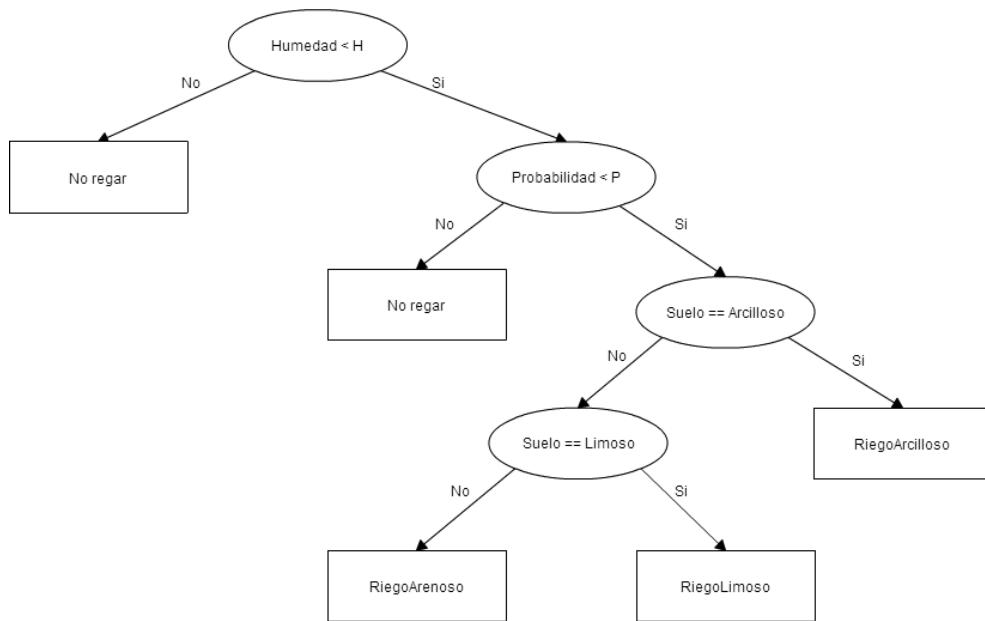


Figura 3.8: Árbol reglas modelo de riego

- El dispositivo consulta el tipo de suelo para ser utilizado en el modelo de riego.

### Determinando tipo suelo

Para determinar el tipo de suelo y guardarla en la base de datos se utilizan dos programas, uno en Python y un servicio web en PHP.

El programa en Python se ejecuta en la Raspberry y utilizando el GPS obtiene las coordenadas geográficas, se puede ver en el Anexo X. Cuando se tienen las coordenadas se llama al servicio web para que este lo guarde.

El servicio web solo recibe como parámetros las coordenadas, es éste quien con la información recibida determina el tipo de suelo y recién es cuando se guarda la información en la base de datos.

El código utilizado se ve en el Listing 3.3:

Listing 3.3: Servicio web que almacena el tipo de suelo

```

id = $_POST["id"];
$lat = $_POST["lat"];
$lon = $_POST["lon"];
$selectSuelo = "SELECT domsoi FROM dsmw WHERE st_contains(geom ,
    ST_GeomFromText('POINT(".$lon." ".$lat.")',4326))";
$result = pg_query($dbconn3, $selectSuelo);
  
```

```
$row = pg_fetch_row($result);
$suelo = $row[0];
$insert = "INSERT INTO coordenadas(id, lat, lon, suelo) VALUES('".$id."
', '".$lat."', '".$lon."', '".$suelo."')";
$result = pg_query($dbconn3, $insert);
```

Luego consulta por el tipo de suelo a una base de datos espacial<sup>2</sup> con la siguiente consulta SQL que se muestra en el Listing 3.4:

Listing 3.4: Servicio web que obtiene el tipo de suelo

```
SELECT domsoi FROM dsmw WHERE st_contains(geom, ST_GeomFromText('POINT
(".$lon." ".$lat.")', 4326))
```

Finalmente inserta un id (de la Raspberry Pi), las coordenadas y el tipo de suelo a la base de datos. El resultado queda como en la Figura 3.9

Acciones	id	lat	lon	suelo
<a href="#">Editar</a> <a href="#">Eliminar</a>	00000000a362b1d3	-34.965555026	-71.232560955	Je

Figura 3.9: Registro almacenado en la base de datos

### Consultando tipo suelo

Cuando el modelo de riego necesita saber el tipo de suelo, consulta por el registro que se mostró anteriormente, esto se hace desde la Raspberry Pi.

El código en Python que lo haces es el que se en el Listing 3.5:

Listing 3.5: Obtener el tipo de suelo desde la Raspberry Pi

```
serial = getserial()
url = "http://bri2. utalca . cl/~nmaturana/getCoordenada . php?id=" + serial
response = urllib . urlopen(url)
data = json . loads(response . read())
suelo = data["suelo"]
```

El servicio en PHP recibe como paramento el id del dispositivo y retorna el tipo de suelo se puede apreciar en el Listing 3.6:

---

<sup>2</sup><http://data.fao.org/map?entryId=446ed430-8383-11db-b9b2-000d939bc5d8>

Listing 3.6: Servicio web que retorna el tipo de suelo

```
$id = $_GET["id"];
$result = pg_query($dbconn, "Select _suelo _from _coordenadas _where _id _=_
". $id . " ");
$arr = pg_fetch_array($result, NULL, PGSQL_ASSOC);
echo json_encode($arr);
```

### 3.2.4. Envío y consulta de datos, humedad y agua utilizada

#### Envío de datos

El envío de datos se hace en la Raspberry la que a su vez llama a un servicio web que almacena los datos en la base de datos.

Los datos son enviados cada 30 segundos, como lo muestra en el Listing 3.7:

Listing 3.7: Envío de datos desde la Raspberry Pi hacia el servidor

```
while True:
    fecha = datetime.datetime.now()
    entradas = arduino.readline()
    datos = entradas.split(" ")
    humedad = datos[0]
    caudal = datos[1]
    mydata=[('humedad', humedad), ('caudal', caudal), ('fecha', fecha)]
    mydata=urlencode(mydata)
    path='http://bri2.utalca.cl/~nmaturana/insertData.php'
    req=urllib2.Request(path, mydata)
    req.add_header("Content-type", "application/x-www-form-
    urlencoded")
    page=urllib2.urlopen(req).read()
    time.sleep(30)
```

La humedad y cantidad de agua utilizada se leen desde el Arduino y son pasadas al servicio web “insertData.php” que basicamente inserta la información recibida en la base de datos como se ve en la Listing 3.8:

Listing 3.8: insertData.php

```
$humedad = $_POST["humedad"];
$caudal = $_POST["caudal"];
$fecha = $_POST["fecha"];
```

```
$insert = "INSERT INTO registros (humedad , caudal , fecha ) VALUES( '' .
    $humedad . '' , '' . $caudal . '' , '' . $fecha . '' )";
$result = pg_query($dbconn3 , $insert);
```

Y el resultado de todo este proceso se ve en la Figura 3.10. La primera columna corresponde a la humedad, la segunda a la cantidad de agua utilizada y la tercera es la fecha.

1021	0	2015-11-05 23:49:48.811118
1023	0	2015-11-05 23:49:53.964913
1023	0	2015-11-05 23:49:59.123227
1023	32	2015-11-05 23:50:04.300711
1023	568	2015-11-05 23:50:09.484097
1021	1048	2015-11-05 23:50:14.665617
1023	1240	2015-11-05 23:50:19.870196
1023	520	2015-11-05 23:50:25.035644
1023	216	2015-11-05 23:50:30.207851
1020	680	2015-11-05 23:50:35.396651
1023	1056	2015-11-05 23:50:40.580829
1023	1080	2015-11-05 23:50:45.756097
1023	320	2015-11-05 23:50:50.922634
1023	64	2015-11-05 23:50:56.088228
131	0	2015-11-06 14:11:16.236601
129	0	2015-11-06 14:11:21.418925
130	0	2015-11-06 14:11:26.571319
129	0	2015-11-06 14:11:31.732779

Figura 3.10: Registros almacenados en la base de datos

## Consulta de datos

Estos datos son consultados por la aplicación web que utilizando JQUERY obtiene los datos. El Listing 3.9 obtiene los datos de humedad del suelo y los almacena en dps para luego mostrar esta información a los usuarios.

Listing 3.9: JQuery humedad de suelo

```
$.getJSON("http:// bri2 . utalca . cl/~ nmaturana/getRegistros . php" , function
(result) {
    yVal = parseInt( result [ "humedad" ] );
    dps . push( { x: xVal , y: yVal } );
    xVal++;
});
```

El servicio web “getRegistros.php” retorna el ultimo registro insertado en la base de datos como se muestra en el Listing 3.10:

Listing 3.10: getRegistros.php

```
$result = pg_query($dbconn , "SELECT_*_FROM_ registros _ORDER_BY_fecha_ DESC LIMIT 1");
$arr = pg_fetch_array($result , NULL, PGSQL_ASSOC);
echo json_encode($arr);
```

### 3.2.5. Aplicación web

La aplicación web que básicamente muestra la humedad actual del suelo y el consumo de agua mensual fue desarrollada utilizando PHP, HTML y JAVASCRIPT.

La Figura 3.11 muestra la humedad registrada por el sensor de humedad y además indica la hora en el que el registro fue capturado.

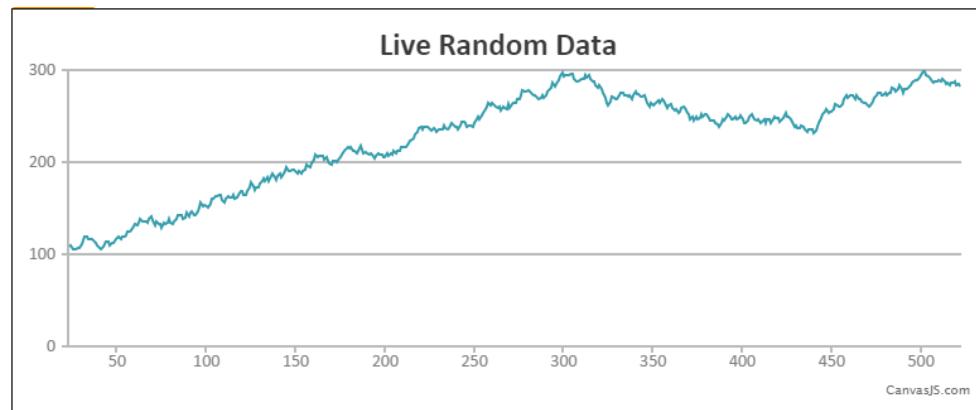


Figura 3.11: Humedad del suelo en tiempo real

La Figura 3.12 muestra el consumo de agua durante los últimos seis meses como se muestra en la boleta de consumo de agua potable que llega nuestro hogar.

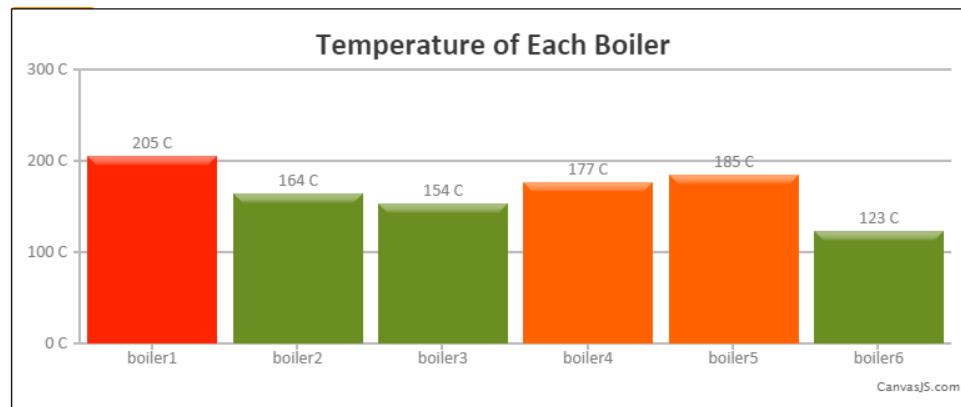


Figura 3.12: Consumo de agua mensual

## 4. Conclusión

---

Se puede concluir que el desarrollo de este dispositivo puede generar un beneficio económico para las personas y a la vez ayuda a combatir el cambio climático que hoy nos afecta, al ser un dispositivo de bajo costo puede ser adquirido masivamente.

Los dispositivos que se venden en la actualidad tienen un costo muy elevado o requieren de mucha intervención de las personas, por lo que muchas veces se dejan de utilizar o no se utiliza correctamente.

Este dispositivo se puede ampliar al uso agrícola, uno de los grandes problemas identificados por el Gobierno es la falta de tecnología aplicada en la agricultura. Es dispositivo como se encuentra en este momento puede ser fácilmente utilizado para este fin, sin duda que se le pueden agregar más sensores y mejorar el modelo para cada tipo de cultivo.

Este trabajo aparte de la investigación, escritura de este documento y desarrollo de un prototipo del dispositivo de riego incluyó también trabajo externo en un concurso de emprendimiento llevado a cabo por Fundación para la Innovación Agraria, el concurso llamado “Proyectos de Emprendimiento Innovador: Jóvenes Innovadores”. En este concurso el dispositivo uno de los finalistas los cuales tuvieron una capacitación en Santiago durante dos semanas con profesores de la Universidad de Oxford, en esas dos semana se vio principalmente gestión y financiamiento de proyectos, en el Anexo X se puede ver parte del trabajo realizado durante este periodo.

Queda claro que es una idea muy viable, que en la actualidad existe un problema que no está solucionado, existe el mercado potencial para hacer económicamente sustentable esta idea y con este trabajo se ha demostrado que se puede hacer algo bueno que genere un gran impacto en la sociedad.

## 5. Trabajo futuro

---

Esta idea tiene un gran potencial y sin duda quedan muchas cosas por mejorar.

En el modelo de riego recomiendo trabajar junto a un experto en el área, además para hacer el dispositivo más inteligente sería ideal que se aplicara un técnica de inteligencia artificial y que fuera aprendiendo que variables de humedad, probabilidad de lluvia, entre otras se debiera utilizar. El diseño del dispositivo se hizo de tal manera que se pueda ingresar un nuevo modelo de riego y que este funcione correctamente, por lo que se puede profundizar mucho más el tema y mejorar el funcionamiento del dispositivo en general.

Este trabajo de enfoco principalmente en el diseño del prototipo del dispositivo y el modelo de riego, el desarrollo de la aplicación para el usuario no fue la mejor, solamente se muestran dos gráficos que entregan información a los usuarios, se le podría agregar mas funciones para que el usuario interactué más con el dispositivo.

También quedo pendiente el desarrollo de una aplicación móvil para los usuarios, lo que haría mucho más interésate el uso del dispositivo.

El hardware que se utilizo se podría reemplazar, disminuyendo la cantidad de componentes que se utilizaron. Existe un microcomputador llamado “beaglebone black” que podría reemplazar a la Raspberry Pi y el Arduino disminuyendo la cantidad de componentes.

# Glosario

**El primer término:** Este es el significado del primer término, realmente no se bien lo que significa pero podría haberlo averiguado si hubiese tenido un poco mas de tiempo.

**El segundo término:** Este si se lo que significa pero me da lata escribirlo...

# Bibliografía

- [1] Arduino.cl. ¿que es arduino? <http://arduino.cl/que-es-arduino/>, 2011. Ultimo acceso: 21-09-2015.
- [2] Cultivar. Raincloud project. <http://ecultivar.com/rain-cloud-product-project/>. Ultimo acceso: 23-05-2015.
- [3] Asociaci n Nacional de Empresas de Servicios Sanitarios. Informe de gesti n de la sequia 2014. <http://www.andess.cl/descargas/noticias/201401-INFORME-GREMIAL-SEQUIA-ANDESS-ENERO-2014.pdf>, 2013. Ultimo acceso: 21-09-2015.
- [4] L. Rucks-F. Garc a-A. Kapl n-J. Ponce de Le n-M. Hill. Propiedades f sicas del sueloweb. <http://www.fagro.edu.uy/edafologia/curso/Material/fisicas.pdf>, Facultad de Agronom a Universidad de la Rep blica, 2004. Ultimo acceso: 23-05-2015.
- [5] Raspberry Pi Foundation. Gpio. <https://www.raspberrypi.org/documentation/usage/gpio/>, University of Minnesota. Ultimo acceso: 23-05-2015.
- [6] Raspberry Pi Foundation. Raspbian. <https://www.raspbian.org/>. Ultimo acceso: 23-05-2015.
- [7] Raspberry Pi Fundation. What is a raspberry pi? <https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>. Ultimo acceso: 23-05-2015.
- [8] Myriam Grajales-Hall. Ahorremos agua usando estrategias probadas de riego para jardines. <http://ucanr.edu/sites/Spanish/Noticias/?uid=5813&ds=199>, Extensi n Cooperativa de la Universidad de California, 2014. Ultimo acceso: 30-06-2015.

- [9] Jose Manuel Gutierrez. Sistemas expertos basados en reglas. <http://personales.unican.es/gutierjm/cursos/expertos/Reglas.pdf>, Dpto. de Matematica Aplicada. Universidad de Cantabria, 1995. Ultimo acceso: 01-11-2015.
- [10] Carlos Andres Morales Machuca. Estado del arte: Servicios web. <http://www.bivica.org/upload/doc1.pdf>, Universidad Nacional de Colombia, 2011. Ultimo acceso: 23-05-2015.
- [11] Rafael Muñoz-Carpena and Michael D. Dukes. Automatic irrigation based on soil moisture for vegetable crops. <http://edis.ifas.ufl.edu/pdffiles/ae/ae35400.pdf>, Department of Agricultural and Biological Engineering, UF/IFAS Extension, 2005. Ultimo acceso: 23-05-2015.
- [12] Eduardo Huerta Aldo Mangiaterra Gustavo Noguera. Gps posicionamiento satelital. [http://www.cartografia.cl/download/libro\\_gps.pdf](http://www.cartografia.cl/download/libro_gps.pdf), Universidad Nacional de Rosario, 2005. Ultimo acceso: 23-05-2015.
- [13] Marco Antonio Bello Maria Teresa Pinto. Programadores de riego. <http://www2.inia.cl/medios/biblioteca/boletines/NR25634.pdf>, Gobierno de Chile, Ministerio de Agricultura, 2000. Ultimo acceso: 23-05-2015.
- [14] Rachio. Rachio iro. <https://rachio.com/>. Ultimo acceso: 23-05-2015.
- [15] Rainbird.es. Programador rainbird. <http://www.rainbird.es/productos/programadores/programadores-de-la-serie-esp-rzx>, 2015. Ultimo acceso: 22-10-2015.
- [16] Jerry Wright. Irrigation water management considerations for sandy soils in minnesota. <http://www.extension.umn.edu/agriculture/water/irrigation-water-management-considerations/>, University of Minnesota, 2008. Ultimo acceso: 23-05-2015.

# **ANEXOS**

## A. Trabajo realizado en FIA

---



Figura A.1: Exterior tríptico realizado en FIA



Figura A.2: Interior tríptico realizado en FIA

## B. Código fuente

---

### B.1. gps.py

Listing B.1: gps.py

```
#!/usr/bin/python
# Written by Dan Mandle http://dan.mandle.me September 2012
# License: GPL 2.0

import os
from gps import *
from time import *
from id import getserial
import time
import threading
import sys
import urllib2, urllib

gpsd = None #seting the global variable
os.system('clear') #clear the terminal (optional)

class GpsPoller(threading.Thread):
    def __init__(self):
        threading.Thread.__init__(self)
        global gpsd #bring it in scope
        gpsd = gps(mode=WATCHENABLE) #starting the stream of info
        self.current_value = None
        self.running = True #setting the thread running to true

    def run(self):
```

```

global gpsd
while gpss .running:
    gpsd.next() #this will continue to loop and grab EACH set of gpsd
info to clear the buffer

if __name__ == '__main__':
    gpss = GpsPoller() # create the thread
    try:
        gpss.start() # start it up
        while True:
            #It may take a second or two to get good data
            #print gpsd.fix.latitude , ' ', gpsd.fix.longitude , ' Time: ', gpsd.
            utc
            os.system('clear')
            if gpsd.fix.latitude != 0.0 and not (math.isnan(gpsd.fix.
                latitude)):
                mydata=[( 'id' ,getserial() ),( 'lat' ,gpsd.fix.latitude ),( 'lon' ,
                    gpsd.fix.longitude )]
                mydata=urllib .urlencode(mydata)
                path='http://bri2. utalca. cl/~nmaturana/insertCoor. php'
                req=urllib2 .Request(path , mydata)
                req.add_header("Content-type" , "application/x-www-form-
                    urlencoded")
                page=urllib2 .urlopen(req) .read()
                print page
                gpsd.running = False
                sys.exit(0)
                time.sleep(5) #set to whatever
            except (KeyboardInterrupt , SystemExit): #when you press ctrl+c
                print "\nKilling_Thread..."
                gpss.running = False
                gpss.join() # wait for the thread to finish what it's doing
                print "Done.\nExiting."

```

## B.2. riego.py

Listing B.2: riego.py

```

#!/usr/bin/python
import datetime
import time
from id import getserial
from clima import lluvia
from serial import Serial
import RPi.GPIO as GPIO
import urllib, json

#BCM -> maneja los pines por los numeros de GPIO17
#BOARD -> maneja los pines por numero de secuencia 1, 2, 3...
GPIO.setmode(GPIO.BCM)
GPIO.cleanup()
GPIO.setwarnings(False)
GPIO.setup(27,GPIO.OUT)

#probabilidad de lluvia
#lluvia(latitud, longitud)

#obtener serial
#getserial()

serial = getserial()
url = "http://bri2.ugal.cl/~nmaturana/getCoordenada.php?id=" + serial
response = urllib.urlopen(url)
data = json.loads(response.read())
suelo = data["suelo"]

arduino = Serial('/dev/ttyUSB0', 9600, timeout=1)

while True:
    hora = datetime.datetime.now().time()
    #Se prende desde las 6 a.m hasta las 23.59 p.m
    if (hora >= datetime.time(6,00) and hora <= datetime.time(6,20)):
        entradas = arduino.readline()
        datos = entradas.split(" ")

```

```
if len(datos) == 2:  
    humedad = datos[0]  
    if humedad < 512:  
        if suelo == "arena":  
            #Riego arenoso  
            GPIO.output(17,GPIO.HIGH)  
            time.sleep(1200)#Riego por 20  
            min  
        else if suelo == "limoso":  
            #Riego limoso  
            GPIO.output(17,GPIO.HIGH)  
            time.sleep(1200)#Riego por 20  
            min  
    else:  
        #Riego Arcilloso  
        GPIO.output(17,GPIO.HIGH)  
        time.sleep(1200)#Riego por 20  
        min  
else:  
    GPIO.output(17,GPIO.LOW)  
    time.sleep(60)
```

### B.3. clima.py

Listing B.3: clima.py

```
#!/usr/bin/python
import urllib2
import json
import time

#f = urllib2.urlopen('http://api.wunderground.com/api/e7d949410a7481dc/
#    forecast10day/lang:SP$'
#json_string = f.read()
#parsed_json = json.loads(json_string)

def lluvia(lat, lon):
    f = urllib2.urlopen('http://api.wunderground.com/api/
        e7d949410a7481dc/forecast10day/lang:SP/q/'+ str(lat) +' , '+
        str(lon) +'.json')
    json_string = f.read()
    parsed_json = json.loads(json_string)
    location = parsed_json[ 'forecast' ][ 'txt_forecast' ][ 'forecastday'
        ][0][ 'title' ]
    temp_c = parsed_json[ 'forecast' ][ 'txt_forecast' ][ 'forecastday' ,
        ][0][ 'fcttext_metric' ]
    pop = parsed_json[ 'forecast' ][ 'txt_forecast' ][ 'forecastday' ,
        ][0][ 'pop' ]
    f.close()
    return pop
```

## B.4. arduino.py

Listing B.4: arduino.py

```
#!/usr/bin/python
import urllib2, urllib
from serial import Serial
import datetime
import time

arduino = Serial('/dev/ttyUSB1', 9600, timeout=1)

while True:
    hora = datetime.datetime.now()
    entradas = arduino.readline()
    datos = entradas.split("\u00a0")
    if len(datos) == 2:
        humedad = datos[0]
        caudal = datos[1]
        print "Humedad:\u00a0" + humedad
        print "Caudal:\u00a0" + caudal
    print hora
    time.sleep(5)
```