ASISTENCIA REMOTA PARA EL CONTROL DE DISPOSITIVOS EN PREDIOS, A TRAVÉS DEL SISTEMA OPERATIVO ANDROID

MARIO ENRIQUE AVILA PARADA

UNIVERSIDAD MILITAR NUEVA GRANADA
FACULTAD DE INGENIERIA
INGENIERIA EN MECATRONICA
BOGOTA DC
2014

ASISTENCIA REMOTA PARA EL CONTROL DE DISPOSITIVOS EN PREDIOS, A TRAVÉS DEL SISTEMA OPERATIVO ANDROID

MARIO ENRIQUE AVILA PARADA
CODIGO 1801554

Informe de investigación realizado para optar al título de ingeniero en mecatrónica

DIRECTORES
Ing. OLGA LUCIA RAMOS SANDOVAL. Ms.c
Ing. DARÍO AMAYA HURTADO. Ph.D

UNIVERSIDAD MILITAR NUEVA GRANADA
FACULTAD DE INGENIERIA
INGENIERIA EN MECATRONICA
BOGOTA DC
2014

NOTA DE ACEPTACIÓN

Informe final de investigación del proyecto para el control de dispositivos en predios, a través del sistema operativo Android, aprobado por el Comité de Grado en cumplimiento de los requisitos exigidos por la Universidad Militar Nueva Granada para optar al título de Ingeniero en Mecatrónica.

Director

Co-Director

Firma Jurado 1

Firma Jurado 2

Firma Jurado 3

BOGOTÁ - Julio de 2014

DEDICATORIA

A mis padres, hermana
y todos aquellos que me brindaron
su apoyo y colaboración a lo largo
del desarrollo de mi carrera.

AGRADECIMIENTO

En primer lugar agradezco a Dios quien me permitió desarrollar una carrera exitosa, brindándome la sabiduría para afrontar las diferentes situaciones que se presentaron.

A mis padres por el esfuerzo, apoyo y colaboración incondicional que me brindaron en todo momento para poder culminar mis estudios de pregrado, una de las tantas metas por cumplir.

A todos los profesores que día a día me guiaban con sus conocimientos a ser una mejor persona tanto moral como intelectualmente. A mis tutores Olga y Dario, quienes me guiaron y apoyaron en el desarrollo de este trabajo.

A aquellos compañeros de equipo que en algún momento de la carrera tuvimos que sacar adelante proyectos y evaluaciones, gracias por su apoyo y colaboración.

LISTA DE FIGURAS

pág.

Figura 1. Esquema del sistema desarrollado para la comprobación de gases vía Smartphone 13
Figura 2. Arquitectura del sistema de monitoreo de temperatura inalámbrico 14
Figura 3. Aplicación web para el sistema de exploración y control usando un smartphone 15
Figura 4. Formas de ventilación 20
Figura 5. Arquitectura Xively 28
Figura 7. Esquema del proyecto 29
Figura 8. Comparación voltaje de salida con diferentes resistencias 32
Figura 9. Conexión sensores 33
Figura 10. Circuito electrónico de acoplamiento 35
Figura 11. Modelo mecánico en SolidWorks 38
Figura 12. Obtencion del modelo mediante Ident 40
Figura 13. Respuesta variable temperatura 40
Figura 14. Respuesta Variable iluminancia 41
Figura 15. Comparación modelos estimados con los datos reales 41
Figura 16. Simulación lazo de control temperatura 42
Figura 17. Respuesta escalón a la temperatura 42
Figura 18. Respuesta escalón a la iluminación 43
Figura 19. Dispositivo web en la plataforma xively 45
Figura 20 Diagrama de flujo tarjeta controladora 48
Figura 21. SDK Manager de Andorid 50
Figura 22. AVD manager de Android 51
Figura 23. Simulación de un dispositivo Android con el AVD 51
Figura 24. Ventana principal Eclipse - Android 52
Figura 25. Carpetas del proyecto en estructura árbol 53
Figura 26. Paleta y parte grafica de la aplicación 54
Figura 27. Edicion de la parte grafica en codigo XML 55
Figura 35. Diagrama de caso uso de la aplicación 57
Figura 28. Creación proyecto en Eclipse 57
Figura 29. Diagrama Clase MainActivity 59
Figura 30. Diagrama clase HttpServiceConnection 61
Figura 31. Diagrama flujo de objeto para las variables 62
Figura 32. Diagrama de clase Hilo 63
Figura 33. Diagrama clase Temperatura e iluminación 64
Figura 34. Diagrama Clase calidad de Aire y Humedad 65
Figura 36. Ventanas de la aplicación 66

CONTENIDO

1. INTRODUCCIÓN 9

1.2 JUSTIFICACIÓN 10
1.3 OBJETIVOS 10
1.3.1 General 10
1.3.2 Específicos 10
1.4 ALCANCE 11

2. MARCO TEÓRICO 12

2.1 ESTADO DEL ARTE 12
2.2 MARCO CONCEPTUAL 15
2.2.1 Dispositivos móviles 15
2.2.2 Variables físicas en un predio 17
2.2.3 Sensores 20
2.2.4 Actuadores 23
2.2.5 Controlador 24
2.2.6 Comunicación inalámbrica 25
2.2.7 Internet de las cosas (IoT) 27

3. INGENIERÍA DEL PROYECTO 29

3.1 DISEÑO ELECTRÓNICO 30
3.1.1 Tarjeta de control 30
3.1.2 Sensores 31
3.1.3 Controlador y actuadores 33
3.1.4 Circuito de acoplamiento 34
3.1.5 Módulo de comunicación 36
3.2 DISEÑO MECÁNICO 37
3.3 OBTENCIÓN MODELO SISTEMA Y CONTROL PID SIMULADO 39
3.4 CONFIGURACIÓN DISPOSITIVO EN XIVELY 44
3.5 PROGRAMACIÓN TARJETA CONTROLADORA 45
3.6 PROGRAMACIÓN ANDROID 49
3.6.1 Configuraciones iniciales del entorno de desarrollo 49
3.6.2 Secciones y estructura de un proyecto Android 51
3.6.3 Desarrollo de la aplicación 55

4. CONCLUSIONES 67

5. BIBLIOGRAFÍA 68

RESUMEN

En el siguiente trabajo es desarrollado un sistema de monitoreo y control de variables en un predio a través de un dispositivo móvil conectado a la web. El sistema se compone de un prototipo que contiene los sensores correspondientes a las variables a ser monitoreados así como los actuadores de las que serán controladas. También, contiene una tarjeta controladora que además de capturar y controlar las variables se encarga de enviar los datos a una plataforma en internet a través de comunicación inalámbrica WiFi, para posteriormente ser observados en una aplicación móvil desarrollada para los dispositivos con sistema operativo Android. En el trabajo se describe el desarrollo de cada uno de los componentes, partes mecánicas, electrónicas y desarrollo de software, tanto en Arduino (tarjeta controladora) como en Android.

ABSTRACT

In this report it is developed a system of monitoring and control of variables in a property through a mobile device connected to the web. The system consists of a model that contains sensors for the variables that will be monitored and actuators which are controlled, it also contains a controller card in addition to capture and control variables is responsible for sending data to a platform Internet via wireless communication WiFi, later to be showed in a developed for devices with Android operating system mobile application. The work discribed the development of each of the components, mechanical, electronic and software development in Arduino (controller card) and software development for Android.

1. Introducción

El avance de la tecnología ha permitido desarrollar diferentes dispositivos con el fin de facilitar las tareas cotidianas, además de brindar seguridad y confort, siendo responsables del cambio de estilo de vida de los individuos. Algunos de esos dispositivos se convierten en parte esencial de la vida de las personas ya que satisfacen alguna necesidad especifica de esa persona o una comunidad, cambiando radicalmente sus costumbres, formas de pensar y de vivir (quees.info, 2013).

Uno de los dispositivos que se ha convertido en parte indispensable para la gran mayoría de personas son los teléfonos móviles inteligentes o *smartphones* (revistaitnow, 2014), que además de permitir hacer llamadas permiten hacer uso de múltiples aplicaciones gracias a la capacidad de procesamiento de datos que poseen. Una propiedad de estos dispositivos y que aumenta su utilidad comprende los diferentes protocolos de comunicación que soportan, dándoles flexibilidad para integrarse con diferentes sistemas (Rita Conde, 2013).

En este trabajo es desarrollado un sistema de monitoreo para cuatro variables a través de una aplicación móvil para sistema operativo Android empleando una comunicación WiFi con conexión a internet. En el capítulo 2 se desarrolla el marco teórico, donde se mencionan algunos proyectos desarrollados en diversas universidades que tienen por objeto el control de algún sistema a través de un Smartphone empleando diferentes formas de comunicación, también se tratan los aspectos técnicos a tener en cuenta para la realización del proyecto así como la descripción de los componentes que serán empleados.

En el capítulo 3 se describe la ingeniería del proyecto, donde se presenta un esquema general de la arquitectura del proyecto y a continuación se van desarrollando los subtemas correspondientes, como son aspectos electrónicos, mecánicos y de programación para la tarjeta controladora y la aplicación móvil.

En los aspectos electrónicos se tratan características de sensores, actuadores, las condiciones de trabajo de la tarjeta controladora y el diseño de circuitos adicionales. En los aspectos mecánicos se describe el diseño de la maqueta en la que se ubicaran los sensores y actuadores. Por último, la parte de programación es desarrollada para dos entidades, tarjeta controladora y dispositivo Android, para lo cual es necesario el uso de dos lenguajes de programación, donde se describe algunos aspectos básicos para el uso de cada compilador y la lógica de programación empleada entidad.

1.2 Justificación

El continuo avance de la tecnología ha permitido al ser humano desarrollar dispositivos y herramientas que facilitan y reducen considerablemente el tiempo de ejecución de diferentes actividades, brindando al usuario confort y seguridad, además de que permiten hacer uso óptimo de los recursos energéticos. Quizá, los dispositivos que han llegado a tener mayor impacto social en los últimos años sean los dispositivos portátiles, entre los que se encuentran los computadores personales, los *Smartphones* o teléfonos inteligentes y las *Tablets,* entre otros. Dichos dispositivos permiten tener al usuario el acceso a la información prácticamente desde cualquier lugar, con una característica principal, son de fácil transporte.

El desarrollo de un sistema que permita integrar el control de diferentes variables con la monitorización de las mismas de forma remota, ya sea por medio de un pc o un dispositivo móvil como una tableta o teléfono móvil, permitiría aumentar los usos de los dispositivos ya mencionados, además de brindarle confort al usuario.

Un sistema de estos podría ser implementado para proyectos domóticos, inmóticos, controles de temperatura en hornos y refrigeradores, apagado y encendido de luces, regulación de intensidad de luz, detección de fugas químicas; en aspectos militares como exploración, desactivación remota de armas; en agricultura el control de riegos, entre muchas otras aplicaciones.

1.3 Objetivos

1.3.1 General

Diseñar y desarrollar una plataforma de control a través de la web de dispositivos en predios, utilizando el sistema operativo Android.

1.3.2 Específicos

- Desarrollar una estructura de comunicación para interconectar sensores de temperatura, humedad, calidad de aire e iluminación, en un predio.
- Diseñar y desarrollar una estrategia de control para las variables de temperatura e iluminación, y la medición de humedad y calidad de aire.
- Diseñar y desarrollar un sistema de información que registre la información del estado de las variables a controlar.

- Desarrollar una aplicación en Android, para acceder a los datos de los sensores a través de la web, permitiendo definir los puntos de trabajo de las variables y la visualización de las mismas.

1.4 Alcance

CONCEPTUAL

Permitir la visualización y control de cuatro variables en un predio a través del acceso a la web en una aplicación de un dispositivo móvil.

GEOGRÁFICA

El proyecto se llevara a cabo en la ciudad de Bogotá D.C. en las instalaciones de la Universidad Militar Nueva Granada, haciendo uso laboratorios de ingeniería, ubicados en la calle 100 con carrera 11.

TEMPORAL O CRONOLÓGICO

El proyecto se realizara en el trascurso del primer semestre del 2014, en el periodo comprendido entre los meses de Enero a Junio.

2. Marco teórico

En este capítulo se explican y se exponen los antecedentes, las teorías y las investigaciones que se tomaron como base para iniciar la contextualización y posterior desarrollo de este trabajo.

2.1 Estado del arte

La aparición de dispositivos móviles ha significado un gran cambio en la sociedad, aparatos electrónicos como los reproductores de música, consolas de juegos, PDAs y teléfonos. Los teléfonos (llamados *smartphones o teléfonos inteligentes*), han llegado a convertirse en algo muy parecido a ordenador personal, a pequeña escala, ya que permite la comunicación inalámbrica y acceso a la web, además de poseer múltiples funciones de procesamiento de datos, siendo esta una característica que ha abierto un mundo de las posibilidades para nuevos desarrollos e investigaciones utilizando estos dispositivos (Weiss, 2002).

Debido a la versatilidad de estos dispositivos (*smartphones*), son múltiples las aplicaciones que se han desarrollado, desde unas muy simples, hasta unas más complejas que incluyen la interconexión con otros dispositivos y/o sistemas. Desarrollos de diferentes instituciones han empleado los smartphones como una herramienta de supervisión de diferentes procesos.

En la Universidad Americana de ciencia y tecnología (AUST) se desarrolló un sistema de detección de gases, el cual hacia uso de un Smartphone con sistema operativo Android, una placa de arduino, una cámara usb, un OMAP3530, un adaptador usb-WiFi y dos sensores de gas. El diseño estaba orientado a la exploración de zonas contaminadas por gases y contaba con la siguiente estructura (Figura 1): la placa Ardunio recibía las consignas dadas por los dos sensores de gas y los enviaba a la placa OMAP3530, que además procesaba la imagen de la cámara y los enviaba todos los datos a través de comunicación inalámbrica (WiFi) al teléfono móvil, que por medio de una aplicación Android permitía al usuario visualizar los estados de los sensores, la imagen capturada por la cámara y permitía controlar el movimiento de un vehículo tipo oruga que soportaba la cámara (R, y otros, 2013).

Figura 1. Esquema del sistema desarrollado para la comprobación de gases vía Smartphone

Fuente: IEEE/Gas Ascertainment via Smartphone

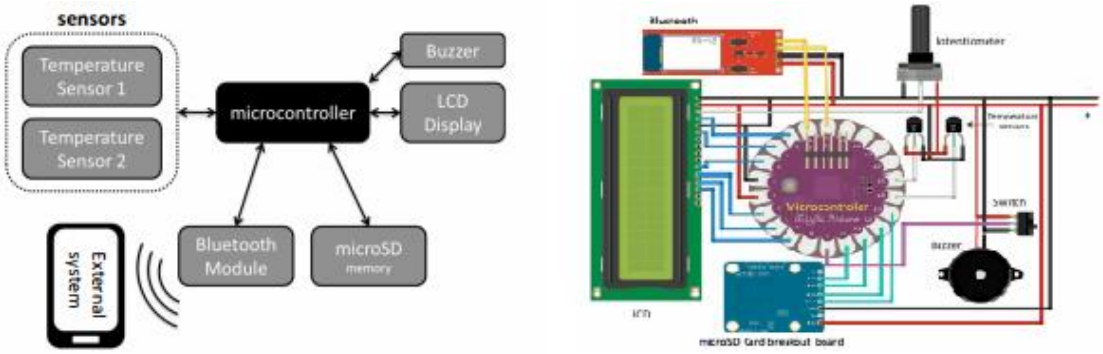
Otra investigación, muy similar a la anterior, tenía por objeto desarrollar un dispositivo móvil pequeño que fuera capaz de detectar gases altamente peligrosos para el ser humano. Para esto hicieron uso de cinco sensores que permitían detectar las concentraciones de CO y NO2 en el ambiente. La captura de los datos se hacía a través de una tarjeta arduino uno, que con la ayuda de un módulo bluetooth permitía la conexión con smartphones, tabletas y computadores personales que soportaran esta forma de comunicación, y contando con que dicho dispositivo contara con una conexión a internet se enviaran esos datos a una base de datos (Heng, y otros, 2012).

También se han desarrollado trabajos relacionados con la monitorización de temperatura y humedad. En la universidad de ciencia y tecnología de Kunming desarrollaron un sistema para la monitorización de la temperatura y la humedad de tres puntos de almacenamiento farmacéutico. Hacen uso de sensores SHT15, módulos zigBee CC2530 para el procesamiento y envió de los datos a través de WiFi, además de que permiten la visualización del estado de cada punto en una aplicación Android y el almacenamiento de datos en una base de datos SQLite (Ge, Wenqing; Lai, Hua; Qi, Mingjie; Cui, Linlin, 2012)

En la Universidad de ciencia de Kalifha desarrollaron un sistema portable para la monitorización remota de la temperatura corporal de un bebe, para cumplir el objetivo hacen uso de dos sensores de temperatura LM35 debido a su alta linealidad, una tarjeta arduino lylipad, un módulo bluetooth, una pantalla LCD, un zumbador, un potenciómetro y un dispositivo móvil con sistema operativo Android. El sistema permitía establecer un valor de referencia de temperatura, a través del potenciómetro o podía ser configurado desde el smartphone, y emitía una alerta sonora si los niveles se salían de determinado rango, también establecido por el

usuario. Todo el procesamiento de la información era elaborado por el arduino lilypad y enviada por medio del módulo bluetooth al smarthpone (Mohammad, y otros, 2013). EL diagrama de la arquitectura de este sistema puede observarse en la *Figura 2*.

Figura 2. Arquitectura del sistema de monitoreo de temperatura inalámbrico



Fuente: IEEE/ Portable Wireless Biomedical Temperature Monitoring System: Architecture and Implementation

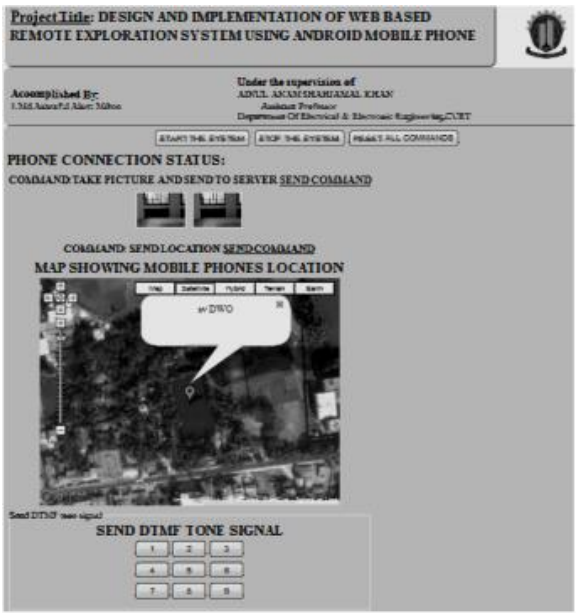
Por otra parte en la universidad de Oriente desarrollan un sistema que permitía la supervisión y control de la temperatura de un horno eléctrico haciendo uso de dispositivos móviles con sistema operativo Android. El sistema desarrollado se compone por un micro controlador encargado del control, un bus de campo que permite la comunicación un computador, una aplicación desarrollada en Java que permite obtener los datos del micro controlador a través de una interfaz RS232 y trasmitirla vía WiFi a un dispositivo móvil, que cuenta con una aplicación desarrollada para Android donde se monitorea y cambia el valor de referencia de temperatura del horno (R.A, 2013).

Otra aplicación que se le puede dar a un smartphone, y en la que se ha investigado es en la exploración remota, ya simplifica y resuelve diferentes inconvenientes que se puedan presentar en un sistema de estos gracias a que integra múltiples componentes necesarios. Integra la parte de visualización (cámara) y cuenta con variadas formas de comunicación (WiFi, GSM), lo que facilitan el control y monitoreo aun a grandes distancias, dándole al sistema una cobertura de exploración prácticamente igual a la de un teléfono móvil (Thompson, y otros, 2013).

Un sistema complejo desarrollado en la universidad de ingeniería y tecnología de Chittagong, hacía uso de una aplicación web, un servidor web, una red GSM, una base

de datos y un móvil con sistema operativo Android. El teléfono móvil era controlado desde la aplicación web (Figura 3), y este a su vez controlaba diferentes componentes eléctricos empleados para la exploración. La aplicación del móvil hacía uso del GPS y la cámara, los cuales permitían mostrar datos e imágenes reales de la posición en que se encontraba el móvil a través de la aplicación web utilizando como medio de comunicación la red GSM y el acceso a internet, lo cual le permitía al sistema tener una cobertura prácticamente igual a la de un teléfono celular (Milton, y otros, 2012).

Figura 3. Aplicación web para el sistema de exploración y control usando un smartphone



Fuente: IEEE/ Web Based Remote Exploration and Control System Using Android Mobile Phone

2.2 Marco conceptual

2.2.1 Dispositivos móviles

Es un dispositivo que permite su fácil traslado de un lugar a otro (portabilidad), por lo cual debe ser pequeño, con capacidad de procesamiento, con conexión a una red, con una determinada cantidad de memoria y diseñado con una función principal pero a la vez versatilidad para realizar otras tareas. Generalmente su uso es personal. Entre muchos de los dispositivos portátiles se pueden encontrar reproductores de audio, navegadores GPS, teléfonos móviles, PDA (Personal Digital Assistant) o llamados

comúnmente tablets, consolas de juegos, entre otros. Aunque un dispositivo móvil puede asemejarse a un computador hay que tener en cuenta que tienen menor funcionabilidad (Rodríguez, y otros, 2012)

Uno de los dispositivos móviles más utilizado en la actualidad es el "Smartphone" o teléfono inteligente. Aunque los teléfonos inicialmente fueron diseñados para la comunicación voz a voz inalámbrica actualmente son utilizados para realizar tareas similares a las de un computador (además de agrupar una gran variedad de dispositivos móviles en un solo equipo), ya que permite la instalación de programas que lo hacen incrementar el procesamiento de datos y conectividad, permitiéndole tener mayor utilidad. Dichos programas o aplicaciones son desarrolladas por los fabricantes del dispositivo o por un tercero, todo dependiendo del sistema operativo que haya sido instalado inicialmente en el dispositivo.

En el mercado actual se encuentran gran cantidad de empresas fabricantes de Smartphones, los cuales ofrecen dispositivos de diferentes referencias, características y sistemas operativos. Entre los fabricantes más sobresalientes a nivel mundial se encuentran Samsumg, Nokia, Apple, LG, ZTE, Hawei, Alcatel, Sony Mobile, Blackberry entre otros (Xataka, 2013).

Sistema operativo para teléfonos móviles

Entre los sistemas operativos para dispositivos móviles preferidos por los fabricantes se encuentran: Android, Windows Phone, IOS y Blacberry IOs. De los cuales Android es preferido por la mayoría de los fabricantes, debido es un software de código abierto o libre (iphoneandord.com, 2013). Android está basado en Linux, creado para una gran cantidad de dispositivos. El principal objetivo de Android es crear una plataforma de software libre disponible para fabricantes de equipos originales y desarrolladores, con el fin de hacer que las ideas se conviertan en realidad mejorando la experiencia móvil de los usuarios (Android.com, 2014).

Lenguaje de programación

Para desarrollar aplicaciones para Android existen diferentes lenguajes entre los que se encuentran:

- Basic4Android: su lenguaje básico de programación es VisualBasic. Necesita licencia.
- Mono para Android: su ambiente de desarrollo es VisualStudio C# y .NET, para lo cual es necesario instalar el SDK de Android. Necesita licencia.
- AppInventor: Herramienta desarrollada por el MIT, permite el desarrollo de aplicaciones sin tener conocimiento de ningún lenguaje de programación,

pues se hace a través de bloques con nombres fáciles de conocer. Licencia gratuita.

- LiveCode: plataforma que permite desarrollar para Android, iOS, Windows, Linux, iPhone, iPad, Web y para Servidores con una sola plataforma de trabajo. El lenguaje utilizado es programación orientada a eventos, que consiste en llevar elementos al área de trabajo y programar eventos que estén asociados a él. Necesita licencia.
- In Desing c6: herramienta desarrollada por Adobe orientada a aquellas personas que escriben libros, revistas, etc, y las publican continuamente. Necesita Licencia
- Java: plataforma nativa de Android y por lo tanto la información suministrada es directamente por Android, los ejemplos y tutoriales publicados se encuentran en este lenguaje. El entorno de programación más utilizado es eclipse y para el cual se debe instalar el SDK de Android de licencia gratuita (Schwarz, y otros, 2011)

2.2.2 Variables físicas en un predio

Son muchas las variables a la hora de realizar un control para obtener cierto grado de confort en un predio, entre estas se pueden encontrar y de las cuales se van a tratar en este trabajo, la humedad, temperatura, iluminación y calidad de aire.

Humedad

Representa el vapor de agua presente en el aire de la atmosfera. Si el aire no contiene vapor de agua es llamado *aire seco,* sin embargo es conveniente tratar el aire como una mezcla de vapor de agua y aire, ya que aunque la cantidad de *aire seco* que contiene la atmosfera permanece relativamente constante, la cantidad de vapor de agua varía dependiendo de la evaporación de océanos, ríos y fuentes de agua. A pesar de que la cantidad de vapor de agua es pequeña, desempeña un papel importante en la comodidad del ser humano (Cengel, 2012). Esa cantidad de vapor presente en el aire puede determinarse de varias maneras:

Humedad absoluta o específica (w): representa la cantidad de masa presente de vapor de agua en una unidad de masa de aire seco dada por la ecuación 1.

$$w = \frac{m_v}{m_a}\ (kg\ vapor\ agua/kg\ aire\ seco) \qquad (1)$$

Debido a que la presión de saturación del agua a temperaturas por debajo de los 50°C permanece constante (12.3 KPa), el vapor de agua puede tratarse como un gas ideal por lo tanto aplicando la ley de los gases ideales se obtiene la ecuación 2.

$$w = 0.622\frac{P_v}{P_a} = 0.622\frac{P_v}{P-P_v} \qquad (2)$$

Considerando que se tiene 1 Kg de vapor seco (humedad especifica cero o sin vapor de agua) y se empieza a añadir vapor de agua, la humedad especifica empieza a crecer hasta tal punto que el aire no pueda recibir más humedad, a este punto se le llama aire saturado.

Humedad relativa (ϕ): Representa la cantidad de humedad que el aire contiene (m_v), respecto a la cantidad máxima de humedad que el aire puede contener a la misma temperatura (m_g) (Ecuación 3)

$$\phi = \frac{m_v}{m_g} = \frac{\rho_v}{\rho_g} = \frac{P_v}{P_g} \qquad (3)$$

Donde $P_{\mathrm{g}} = P_{\mathrm{sat\ a\ T}}$

La humedad relativa varía entre 0 para aire seco y 1 para aire saturado. Es importante tener en cuenta que la cantidad de humedad que puede contener el aire depende de su temperatura, por lo tanto la humedad relativa del aire varia con la temperatura aunque la humedad especifica se mantenga constante. Si se multiplica por 100 la ecuación 3 se expresa la humedad relativa en valor porcentual.

La humedad relativa recomendable esta entre el 40% y 50%. Una humedad relativa alta (entre 60% y 70%) con calor ambiental produce sudoración, pero en ambiente húmedo el sudor no puede evaporarse y aumenta la sensación de calor. Una humedad relativa menor del 30% produce sequedad de la piel y dermatitis, dolores de cabeza, aumento de susceptibilidad a infecciones y sensación de falta de aire (InstitutoSindicaldeTrabajo, 2007).

Temperatura

Se trata de una medida del “calor” o “frio” de un objeto. De forma indirecta, es una propiedad que permite determinar la dirección del flujo de calor, ya que el calor fluye de manera espontánea de una sustancia a mayor temperatura (Brown, y otros, 2004). Existen varias escalas que permiten tener una base común para las mediciones de temperatura, las más utilizadas son la Celsius (°C) y la Kelvin (K), pero también se

encuentran las escalas Fahrenheit (°C) y Rankine (°R). La escala Celsius toma como base el punto de congelación del agua, mientras que la escala Kelvin toma como punto mínimo el cero absoluto (Cengel, 2012).

La escala Kelvin se relaciona con la Celsius mediante la ecuación 4.

$$T(K) = T(°C) + 273.15 \qquad (4)$$

Calidad de aire

Permite indicar que cantidad de aire está exento de contaminantes para determinar si es apto para ser respirado. Esos contaminantes son emitidos por diversas fuentes que se subdividen en dos, exteriores e interiores. Entre las fuentes exteriores se encuentran las industrias y los vehículos, mientras que entre las fuentes interiores de contaminación sobresalen sistemas de calefacción, cocinas, pinturas, ambientadores, humo de cigarrillo, los cuales están emitiendo constantemente diferentes óxidos (NOx, CO, SO2) y materiales en partículas al aire atmosférico (Farrás, 2012).

Para el tener el control de la calidad del aire pueden utilizarse técnicas como el aumento de volumen de aire fresco con el objetivo de disminuir las concentraciones de contaminantes, reducción de la humedad relativa para evitar el desarrollo de microorganismos o el suministro intermitente de aire exterior o ventilación (Figura 4). La ventilación puede realizarse mediante la extracción del aire, la impulsión de aire limpio hacia adentro, o una mezcla de los dos sistemas (Martínez, y otros, 2005).

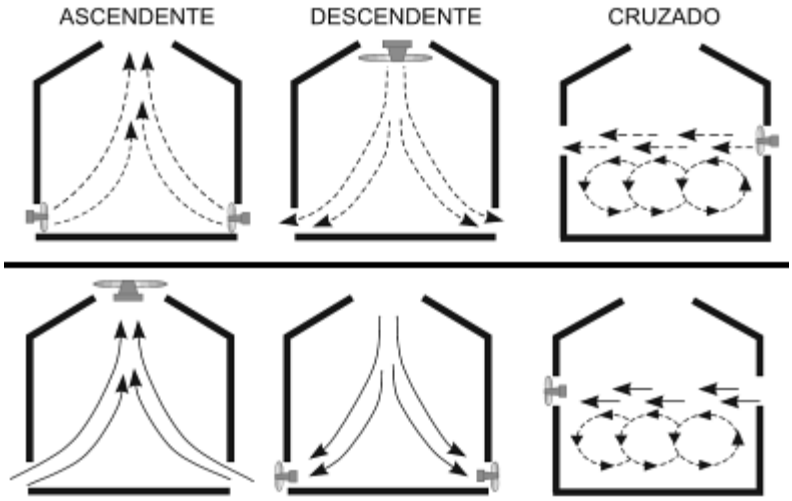
Iluminancia

Representada mediante la letra ***E*** representa la luz o flujo luminoso (ϕ) que llega a determinada superficie. Su expresión matemática está dada por la ecuación 5. Sus unidades son los lux (lumen/m2).

$$E = \frac{\phi}{A} \qquad (5)$$

Un factor que se tiene en cuenta a la hora de iluminar un lugar está dado por la iluminancia media, se establece en función del uso que se pretenda dar al lugar u objeto a iluminar, algunos usos cotidianos se presentan en la Tabla 1. Tomado de (Rodrigo, 2013)

Figura 4. Formas de ventilación



Tomado de google: Calidad de ambientes interiores

Tabla 1. Iluminancia según uso de la instancia

Alumbrado	Lux
Cocinas	100-300
Baños y Aseos	150-300
Alumbrado General en las salas de estar	50-100
Dormitorios	150-300
Pasillos y escaleras	50-100
Garajes	50-100

2.2.3 Sensores

Un sensor es un dispositivo que a partir de la energía del medio en que se mide, proporciona una señal (generalmente eléctrica) que es función de la magnitud que se quiere medir. Entre las diferentes variables a medir se pueden encontrar, temperatura, iluminación, humedad, presión, distancia, fuerza, entre otras.

Sensor de temperatura

Transforma los cambios de temperatura en energía eléctrica, se desarrollan a partir de tres tipos de transductores que se encuentran disponibles en el mercado (medirtemperatura, 2013):

- Termistores: fabricados de semiconductores, aprovechando el comportamiento de la resistencia en estos. Pueden ser de dos tipos negativos (NTC) o positivos (PTC), dependiendo del “signo” el comportamiento de la resistencia será directa o inversamente proporcional a la temperatura. No son lineales.
- RTD: basados en la resistencia de un conductor, son altamente lineales. Entre los metales usados para su elaboración se encuentran el platino, cobre, níquel y molibdeno.
- Termopares: está formado por dos metales, puede ser, hierro y constantano, cobre y constantano o antimonio y bismuto, al generarse una diferencia de temperatura se genera una diferencia de potencial eléctrica del orden de los milivotltios.

Sensor de temperatura LM35

Es un sensor lineal de temperatura con una escala calibrada en grados centígrados de 10mV/°C, lo que significa 10°C equivalen a 100mV. Tiene una exactitud garantizada de 0.5°C para temperaturas mayores a 25°C. Trabaja para un rango entre -55°C a 150°C. Opera entre 4V y 30V de alimentación. Debido a su baja impedancia de salida y salida lineal los circuitos de lectura son sencillos.

Sensor Fotorresistencia LDR

Transforma los cambios de intensidad de luz en energía eléctrica. El transductor más utilizado es el LDR (Resistor dependiente de la luz, por sus siglas en inglés) que consiste en un resistor que cambia su valor dependiendo de la intensidad de luz a la cual está expuesto.

Está constituida por una célula de Sulfuro de Cadmio, altamente estable, encapsulada con una resina epoxi transparente. La respuesta espectral es similar a la del ojo humano. Su nivel de resistencia aumenta cuando el nivel de luz disminuye, con una constante de -9.9Ω/Lux teóricamente.

Sensor de humedad

Permite detectar los cambios de concentración de agua en el aire o una mezcla gaseosa mediante cambios eléctricos. Son desarrollados bajo diferentes conceptos y/o propiedades de los materiales entre los que se encuentran, sensores mecánicos, de bulbo húmedo y seco, sensor por condensación, sensor por electrolisis y sensores tipo capacitivos.

Sensor de humedad y temperatura RHT03

Se trata de sensor de humedad y temperatura tipo capacitivo, que permite obtener las consignas de las variables en forma digital, mediante la conexión de un solo pin. Para obtener el valor de las variables es necesario hacer una comunicación simplex con un microcontrolador. Desde el microcontrolador es necesario enviar al sensor una señal de activación, una vez el sensor recibe esa señal, hace la medición y retorna una trama de 32 bits que divide en dos, la parte alta contiene el valor binario de la temperatura expresada en grados Celsius y la parte baja el valor de la humedad relativa expresado en porcentaje.

El fabricante garantiza la estabilidad del sensor, calibrando el dispositivo en una cámara de calibración precisa, donde obtiene un coeficiente de calibración que lo almacena en una memoria ROM del sensor, para que posterior al tomar una consigna se pueda calcular el valor adecuado de las variables mediante un método matemático.

Sensor de Gas

Se trata de un elemento que sufren un cambio físico o químico, reversible, al estar expuestos a determinado gas, entregando una señal para activar una alarma o realizar algún proceso. En la industria se encuentran diseñados utilizando diferentes propiedades físicas y químicas, entre las que se encuentran los sensores electroquímicos, sensores fabricados por semiconductores, sensores por conductividad térmica, sensores catalíticos y sensores infrarrojos (Pantoja, 2012).

Sensor de gas MQ135

Se trata de un sensor semiconductor que permite detectar la concentración de diferentes sustancias en el aire (Amoniaco NH3, Óxidos de nitrógeno NOx, alcohol, benceno, humo de cigarrillo, CO2, entre otras), es utilizado en interiores a temperatura ambiente. Para realizar la medición es necesario conectar una resistencia de carga que dependiendo de la aplicación tomara diferentes valores. Si se desea medir un gas en específico es necesario calibrarlo con una resistencia que varía entre 10KΩ y 40KΩ (conociendo la concentración del gas para variar la sensibilidad), si por el contrario se utiliza medir cualquier gas o detector de calidad aire es necesario calcular el valor de una resistencia que permita obtener un voltaje determinado al estar expuesto a un *aire limpio.*

2.2.4 Actuadores

Son los encargados de realizar acciones físicas sobre la variable de un al recibir la orden de algún sistema de control, su nombre está asociado a la fuente de energía (eléctrica, hidráulica, neumática) que le permita realizar la acción. Dicha acción física puede estar dada por un movimiento lineal o rotacional.

Ventilador

Maquina eléctrica que tiene como función principal generar una corriente de gas permanente, para llevar a cabo esta tarea se vale de un rodete que contiene aspas que al girar generan una diferencia de presiones, lo que hace fluir el gas en determinada dirección. Son empleados tanto industrialmente como residencialmente para hacer circular aire en un lugar cerrado y brindar comodidad a los ocupantes al disminuir olores, proporcionar una cantidad de oxigeno suficiente y la disminución de la sensación térmica (definicionabc, 2008).

Debido a que los actuadores pueden llegar a tener un consumo eléctrico mayor al suministrado por las tarjetas de control, es necesario el uso de dispositivos que permitan acoplar el circuito de control con el de potencia. Ese acople puede realizarse fácilmente a través del uso de transistores mediantes algunas configuraciones básicas.

Transistor

Dispositivo electrónico construido con materiales semiconductores, empleados para para producir una señal de respuesta a una señal de entrada. Tiene dos formas básicas de trabajo, la de amplificación de señales y la de conmutación, pasando o cortando las señales eléctricas de utilizado para producir una señal de salida a partir de una pequeña señal de control.

Existen dos tipos básicos de transistores, transistores bipolares (BJT) controlados por corriente, y los transistores de efecto de campo (FET) controlados por voltaje. Para utilizar el transistor como conmutador es necesario colocarlo en la región de corte y saturación, una de las ecuaciones útiles a la hora de calcular la resistencia de base *(Rb)* para un transistor BJT está dado por le ecuación 6.

$$Rb = hfe\left(\frac{Vi - Vbe}{Ic}\right) \qquad (6)$$

Donde *hfe* es el factor de ganancia de corriente y *Vbe* es el voltaje consumido entre base- emisor, valores que son entregados por la hoja de datos, *Vi* es el voltaje de

entrada o señal de control y *Ic* es la corriente de colector o la corriente que consume la carga.

2.2.5 Controlador

Es el componente principal de un sistema de control, detecta la diferencia existente entre un valor medido por un sensor y un punto de referencia dado (set point) para dar una señal de corrección a un actuador que corrija el estado de la variable medida. Puede construirse de forma análoga o digital, siendo los controladores digitales más versátiles a la hora de ser implementados suponiendo una amplia cantidad de ventajas sobre los análogos.

Microcontrolador

Se trata de un microprocesador incluido todo en un chip, CPU, memoria y elementos periféricos de forma que puede realizar todo un sistema de control conectando únicamente elementos exteriores (Santamaria, 1993). Se han desarrollado para cubrir diversidad de aplicaciones, juguetes, equipos médicos, electrodomésticos, equipos industriales, etc. Concebidos para realizar tareas específicas, donde debe realizar un pequeño número de tareas al menor costo posible, las cuales realiza mediante un programa almacenado en su memoria, que interactúa con datos almacenados temporalmente y con el exterior mediante los periféricos de entrada y salida.

Dependiendo de los diseños realizados por los fabricantes, en el mercado se encuentran microcontroladores con diferentes características y tamaños, que se seleccionan dependiendo de la aplicación para la que serán utilizados. Características tales como numero de pines de entrada y salida, velocidad de procesamiento, tamaño de los registros y funciones especiales como timers, módulos de conversión análoga digital y digital análoga, módulos de captura y comparación, PWM, módulos de comunicación, etc (Areny, y otros, 2007).

Arduino

Es una plataforma de electrónica abierta para la creación de prototipos en software y hardware flexibles y fáciles de usar, con el fin de facilitar el uso de la electrónica en diferentes proyectos. Su hardware se basa en una placa con un microcontrolador Atmel y un entorno de desarrollo. Al igual que un microcontrolador una placa arduino cuenta con puerto de entrada y salida y funciones especiales propias de un microcontrolador. El microcontrolador se programa bajo el lenguaje de programación Arduino y cuenta con su propio entorno desarrollador.

En la actualidad Arduino cuenta con una variedad de tarjetas, shields o módulos especiales externos y accesorios que pueden ser seleccionadas dependiendo de la aplicación para la cual vayan a ser utilizados. Las tarjetas varían de tamaño, forma y capacidad dependiendo del microcontrolador utilizado, por lo tanto se pueden encontrar tarjetas como el Arduino Uno, Arduino Ethernet, el Arduino Leonardo, Arduino Lilypad, el Arduino Mega, entre otras (Arduino, 2014).

También disponen de shields para diferentes formas de comunicación, Arduino GSM shield, Arduino Ethernet shield, Arduino WiFi shield y para el control de motores DC como el Arduino Motrol Shield. Además adaptadores serial/usb y pantallas lcd gráficas.

Arduino UNO

Se basa en un microcontrolador ATmega 328 que cuenta con 14 pines de entrada y salida digital, de los cuales seis pueden ser utilizados como salida análoga (PWM a una frecuencia de 500Hz por defecto), además cuenta con seis canales de conversión análoga-digital, un pin para cambiar el voltaje de referencia en la conversión análoga-digital y dos para comunicación I2C (SCL y SDA). También dispone de cuatro diodos led, dos que permiten visualizar el estado de la comunicación serial, uno que va conectado al pin 9 del microcontrolador que puede ser usado con múltiples aplicaciones y uno que indica el estado encendido de la tarjeta (Arduino, 2014).

2.2.6 Comunicación inalámbrica

La comunicación se trata del envió de información de una entidad a otra, por lo cual se hace imprescindible en todo proceso de control, ya sea en una sola dirección (simplex) o en dos o más direcciones (halfduplex, fullduplex y full-fullduplex). Una forma de comunicación que toma cada día mayor fuerza en equipos y procesos está dada por la comunicación inalámbrica (sin cables), que dependiendo de la aplicación, cubre desde pequeñas distancias (centímetros) hasta grandes distancias (varios kilómetros). Diferentes tecnologías y protocolos de comunicación inalámbrica se han desarrollado para diversas aplicaciones, por ejemplo GSM, 3G, LTE para telefonía móvil, WIFI, WIMAX para redes inalámbricas con acceso a internet, Bluetooth, Ultrawideband como sistemas de conexión sin hilo de periféricos.

WIFI

Se trata de una tecnología que permite interconectar gran cantidad de equipos sin la necesidad de utilizar cables. También es llamada WLAN (wirless lan) o estándar IEEE 802.11, debido a que es el estándar que garantiza que los equipos con WiFi puedan

trabajar sin problemas, independientemente del fabricante del dispositivo, además establece que la velocidad máxima de transmisión.

En el año 1999 se estableció la "norma original" (IEEE 802.11b) que permitía una velocidad máxima de transmisión de datos es 11 Mbps utilizando un ancho de banda de 2.4GHz. En ese mismo año también se estableció el estándar IEEE 802.11a que utilizaba un ancho de banda de 5GHz lo que permitía alcanzar velocidades mayores (hasta 108Mbps).

Para el 2003 se desarrolló el estándar IEEE 802.11g la cual buscaba aumentar la velocidad de transmisión (54 Mbps) sin renunciar a las ventajas que proporcionaba la banda de los 2.4GHz (algunas versiones alcanzan los 100Mbps).

En el 2009 aparece el estándar IEEE 802.11n, con este nuevo estándar se habla de velocidades de 300Mbps y alcances mucho mayores en comparación con los otros estándares, además de que es compatible con cualquiera de los anteriores. Una de las características externas es que posee varios canales para poder utilizarlos simultáneamente (Falcon, 2010).

Cabe resaltar que para que dos dispositivos se puedan comunicar a una velocidad alta deben ambos contar con el mismo estándar, de lo contrario la velocidad de comunicación será baja, por ejemplo si se conectan dos dispositivos 802.11g podrán alcanzar velocidades de 54Mbps, pero si uno es 802.11a y el otro 802.11g la velocidad máxima de comunicación será 11Mbps (aulaclic, 2005).

Una de las principales ventajas de WiFi es que permite la libertad de movimientos de los dispositivos conectados a una red sin la necesidad de estar extendiendo cables, lo que reduce costos a la hora de realizar la instalación de una red. A demás una red WiFi es flexible ya que solo con ingresar un equipo al área de cobertura puede formar parte de la red. También es escalable ya que permite ingresar nuevos dispositivos sin tener que hacer ningún cambio físico en la red, con tan solo hacer unos click y un par de instrucciones se puede agregar un nuevo dispositivo. A demás utiliza el protocolo de comunicación TCP/IP, protocolo que también es usado por internet y que lo hace compatible con redes cableadas.

De las desventajas, en comparación con las redes cableadas, resaltan las interferencias ya que el protocolo 802.11 usa el espectro de frecuencias de 2.4GHz, banda que no requiere licencia administrativa para ser usada, por lo que es usada por muchos dispositivos electrónicos comerciales (teléfonos, microondas…), incluyendo redes WiFi vecinas que utilizan la misma frecuencia. Esas interferencias hacen que el

entorno radioeléctrico este contaminado y por ende la calidad de la comunicación baje.

2.2.7 Internet de las cosas (IoT)

El termino IoT está asociado a un concepto de interconexión de dispositivos existentes a través de internet (Sensors empower the "Internet of Things", 2010). Lo que busca este concepto es expandir un poco más el campo de aplicaciones del internet (donde inicialmente era posible solo el acceso a portales y servicios), para permitir la interacción entre personas, procesos y objetos a partir de sensores, redes de comunicaciones y una unidad que interprete y distribuya los datos (José Miguel Roca Chillida, 2013).

Este concepto ha revolucionado el mundo de tal forma que son gran cantidad de empresas que se dedican al desarrollo de dispositivos que permitan la conexión de sus equipos a internet (Esteban Dávila Náder, 2014). De igual forma hay empresas dedicadas a la creación de plataformas que permitan la conexión de sensores y dispositivos por parte de desarrolladores (conex-lab.com, 2012), entre las que se encuentran Xively y Exosite (Ojeda, 2013) que destacan por ser plataformas gratuitas.

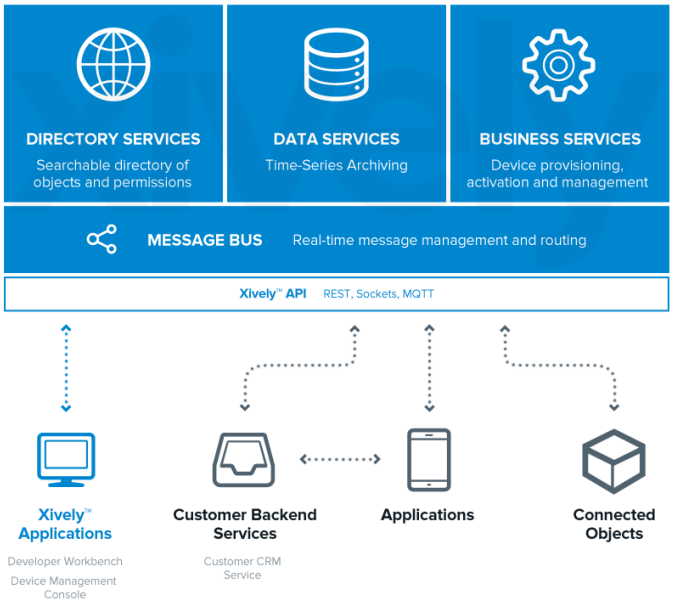
Xively

Se define como una “plataforma como un servicio” (PaaS), que se especializa en recibir y trasmitir información de sensores conectados a través de internet para potencializar los desarrollos relacionados con IoT. A demás es una plataforma abierta, con amplio soporte de bibliotecas, que le permite ser integrado con gran cantidad de lenguajes de programación en los que se incluye java, javascript, C, Ruby, entre otros, y haciéndolo compatibles con plataformas como Arduino y Raspberry (xively, 2014).

La plataforma también ofrece librerías para la comunicación con dispositivos con sistema operativo Android, característica que lo hace sobresaliente sobre otras plataformas.

La arquitectura del de la plataforma se presenta en la Figura 5. En la parte más baja se encuentran todos los dispositivos físicos conectados (sensores, actuadores, etc), en la siguiente capa se sitúa la interfaz de programación y protocolos de comunicación. La tercera capa permite la administración y enrutamiento de los datos en tiempo real. La etapa de la parte superior describe los diferentes servicios que ofrece la plataforma, permisos de administrador o de cliente, entre otros. (xively, 2014).

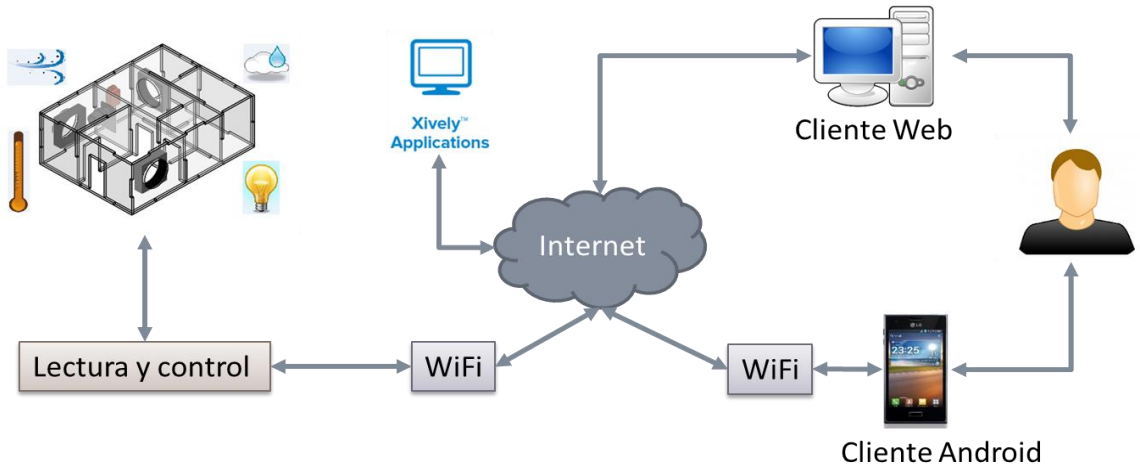
Figura 5. Arquitectura Xively



3. Ingeniería del proyecto

El proyecto tiene como fin el monitoreo de variables en un predio a través de un dispositivo móvil con sistema operativo android, en la Figura 6 se muestra el esquema de lo que se intenta desarrollar. En la primera parte se muestrean y controlan variables seleccionadas en un predio, para luego ser visualizadas por parte del usuario final en un dispositivo con sistema operativo android a través de la web, enviando los datos a la web de forma inalámbrica a través de comunicación WiFi y recibiéndolos de la misma forma en el dispositivo móvil, permitiendo al usuario visualizar el estado de cada una de ellas, así como establecer puntos de referencia en dos de ellas (temperatura e iluminación).

Figura 6. Esquema del proyecto



Las variables seleccionadas para observar y/o controlar son:

- La temperatura y la humedad relativa, que permiten al usuario tener cierto grado de confort o comodidad al estar en un lugar, al evitar casos extremos de temperaturas muy bajas o muy altas, o de informar la humedad relativa a la que puede estar expuesto.
- La iluminancia que brinda comodidad al usuario en cuanto al sentido de la vista se refiere, así como disminución del consumo energético en condiciones de iluminación que no es necesario que el aparato de iluminación este encendido.
- La calidad de aire, que brinda seguridad al usuario informándole de momentos en los que puede estar sometido a gases tóxicos inapropiados para ser inhalados por el ser humano, además que le brinda cierto grado de confort.

Las características de esas variables se definen en la sección 2.2.2.

Para llevar a cabo el monitoreo y control de las variables es necesario desarrollar un diseño electrónico, mecánico y de software, el cual es presentado en las siguientes secciones.

3.1 Diseño electrónico

Basándose en las cuatro variables a controlar u observar es necesario seleccionar la tarjeta de control, así como sensores y actuadores que serán utilizados en el sistema, además de desarrollar los circuitos de potencia necesarios que vayan acorde a la necesidad de los actuadores.

3.1.1 Tarjeta de control

Se trata de la tarjeta que realiza el muestreo de las señales, ejecuta el algoritmo del bloque de control y entrega la señal de control correspondiente para cada una de las variables. Además se encarga de enviar y recibir información a través de comunicación WiFi, que permita al usuario visualizar el estado de las variables y la necesaria para realizar el proceso de control, desde cualquier lugar en el que se tenga acceso a internet.

Para llevar a cabo ese proceso se selecciona la tarjeta Arduino Uno cuyas características fueron mostradas en la sección 2.2.5. La tarjeta opera con 5V de alimentación USB o entre 7 y 12V para una fuente de voltaje externa, una corriente máxima de 40mA, la frecuencia de trabajo está dada por un cristal de 16 MHz. Las dimensiones de la tarjeta son 5,3 cm de ancho por 6.8 cm de largo y 1.5 cm de alto.

Dado que los registros del microcontrolador dedicados para la conversión análoga digital tienen un tamaño de 10 bits, la resolución del conversor análogo digital es

$$\frac{V_{+ref} - V_{-ref}}{2^{\#bits}} = \frac{5-0}{2^{10}} = 4.9mV$$

Valor que es útil a la hora de realizar la obtención de los datos muestreados.

Todos los pines del microcontrolador son llevados a unos conectores regleta tipo hembra, que permiten conectar y desconectar de manera simple, los sensores y actuadores que sean necesarios para la interacción entre el programa desarrollado y el entorno exterior.

La fuente de alimentación seleccionada para el encendido de la tarjeta es a través del cable USB a 5V, que a su vez emula un puerto serial para realizar la comunicación entre la tarjeta y el computador, permitiendo el monitoreo del desarrollo del programa a través del puerto serial.

3.1.2 Sensores

Para realizar las tareas de monitoreo y control es necesario tener un elemento de medida que permita cuantificar el estado de cada una de las variables, para tal fin se eligen los sensores mencionados en la sección 2.2.3

Sensor de gas MQ135

Es empleado como un sensor de calidad de aire, cuya resistencia de carga es de 1KΩ, entregando un voltaje de 63 mV en condiciones normales "Aire limpio", aumentando el valor de la tensión al estar sometido a concentración de gases contaminantes. Tomando como referencia los 63 mV y haciendo una equivalencia al 90% de calidad de aire (CA) y 3V a 20% de la calidad de aire obtiene la ecuación 7, que permite expresar el porcentaje de la calidad de aire dependiendo del voltaje entregado por el sensor.

$$\%\,CA = \frac{(Vi-0.063)V*(90-20)\%}{(3-0.063)V} + 20\% = 27.2387Vi + 18.284 \qquad (7)$$

Sensor LM35

Como se describe en la hoja de datos de este sensor, entrega 10mV/°C, por lo que para obtener la temperatura correspondiente se aplica la ecuación 8.

$$T(°\mathrm{C}) = 100Vi \qquad (8)$$

Sensor de temperatura y humedad RHT03

El sensor permite realizar mediciones de temperatura y humedad, garantizando por parte del fabricante la estabilidad y fiabilidad del dispositivo, el sensor es totalmente digital, por lo que no hay que realizar operaciones matemáticas para obtener el valor de las variables.

Por especificaciones del fabricante es necesario conectar una resistencia de 1KΩ entre *Vcc* y el pin de datos. Las mediciones para este sensor deben realizarse cada segundo, pues de lo contrario el sensor envía una señal de error, este dato es

importante tenerlo en cuenta a la hora de programar, ya que se puede cometer el error de pedir datos al sensor antes de tiempo, obteniendo una respuesta incorrecta.

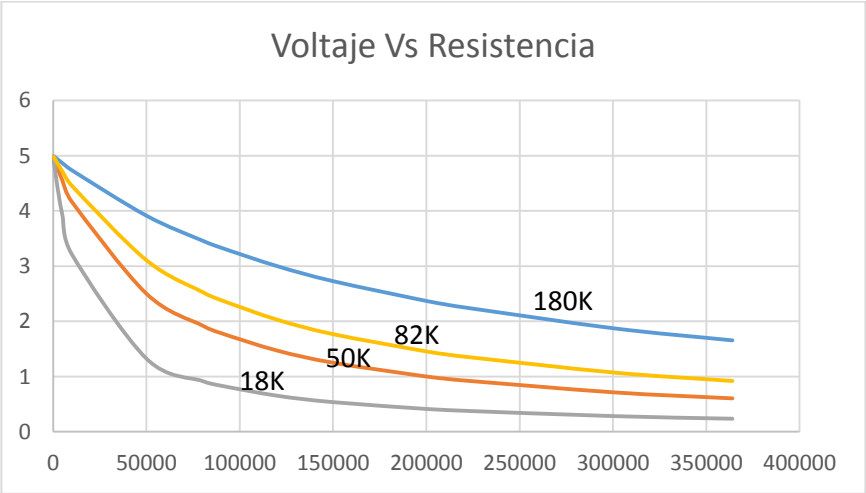
Fotorresistencia LDR

Realizando la medición de resistividad en los puntos críticos se obtiene que sin luz tiene una resistencia R=483KΩ, mientras que a máxima luz el valor de la resistencia R=235Ω. El circuito de acondicionamiento se compone de un divisor de tensión, donde es colocada la resistencia variable a Vcc mientras la resistencia fija va a tierra, con el fin de que con un aumento de la luminosidad la señal de salida aumente su tensión. La expresión que indica el voltaje de salida está dado por la ecuación 9.

$$Vo = Vcc\frac{Rf}{Ri+Rf} \qquad (9)$$

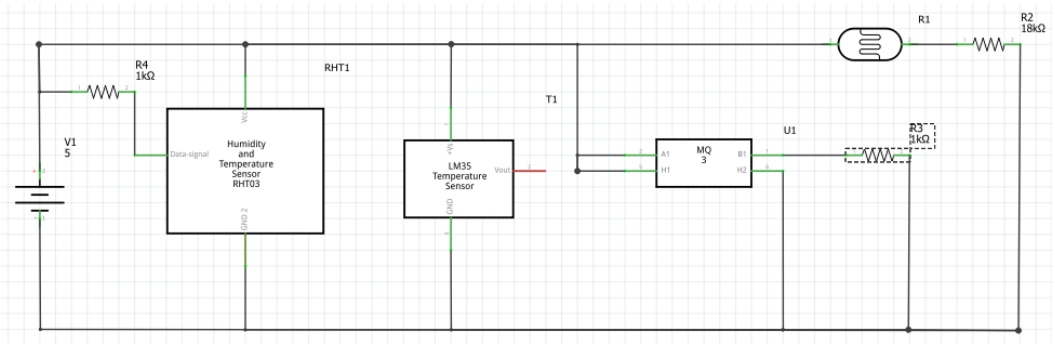
Sabiendo que Vcc=5V, y que para cuando no hay luz el voltaje deseado es aproximadamente 0V (Vo≈0V), mientras que cuando la luz sea máxima el voltaje aproximado sea 5V (Vo≈5V). Despejando *Rf* de la ecuación 9 se obtiene los valores de resistencia que oscilan entre cero e infinito, lo cual en la práctica no es posible, por lo tanto se definen valores de voltaje de salida +- 0.5V, obteniendo que la resistencia fija debe estar entre 1.58KΩ y 68KΩ. En la Figura 7 se grafica los voltajes obtenidos para la ecuación 9, empleando diferentes valores para *Rf,* donde se puede observar que a mayor valor de resistencia la curva se va haciendo más lineal, pero va perdiendo sensibilidad. Teniendo en cuenta estas condiciones se selecciona una resistencia de 180 KΩ, que hace más lineal el sistema.

Figura 7. Comparación voltaje de salida con diferentes resistencias



Una vez realizado los cálculos correspondientes para los circuitos se realiza el diagrama de conexión de los sensores descritos anteriormente como se observa en la Figura 8, donde la tensión de 5V es entregado por un pin disponible en la tarjeta Arduino Uno.

Figura 8. Conexión sensores



3.1.3 Controlador y actuadores

Dependiendo de la variable a controlar se selecciona la estrategia de control así como los actuadores, los cuales se encargaran de actuar sobre la variable dependiendo de la señal entregada por el regulador. A continuación se relacionan las variables y sus respectivos actuadores.

Para la variable iluminancia se emplea un regulador PID y como actuador un diodo led que aumenta la iluminación del sistema dependiendo de la señal entregada por el regulador. Entre los factores por los que es escogido el led están: su fácil implementación, permite regular la intensidad de luz, su bajo consumo (1.5-2.2V, 15-30mA), alrededor de los 0.06W, le permite ser conectado directamente a un canal de salida del Arduino Uno sin que altere su comportamiento. Se conecta una resistencia de 220Ω en serie al diodo led, con el objeto de delimitar la corriente que fluye a través del mismo, para evitar que este se dañe.

Para la variable temperatura también se emplea un controlador PID y como actuador se selecciona una resistencia fija bobinada – cementada (tipo tiza) de 4.8Ω que al ser alimentada con una tensión de 5 V consume 1A de corriente, consumiendo una potencia máxima de 5W que se puede ver representada en una transformación de aumento de temperatura.

También se emplea un ventilador que funciona con una alimentación de 5V dc y una corriente de 170 mA, controlado por un regulador on-off, que permite disminuir la

temperatura del sistema, al ingresar aire con menor temperatura, lo que se traduce en la disminución de tiempo de enfriamiento una vez el sistema ha superado la temperatura de referencia .

Para la variable calidad de aire se hace uso de un ventilador que funciona con una alimentación de 5V dc y una corriente de 170 mA, el cual se activa una vez se detecta que la calidad de aire ha disminuido de cierto punto, con el fin de brindar seguridad al usuario frente a la presencia de gases contaminantes y/o tóxicos, extrayendo el aire del interior enviándolo hacia el exterior.

Debido a que cada canal del Arduino puede entregar como máximo 40mA y el mínimo requerido por los actuadores es de 170mA es necesario hacer uso de una fuente de voltaje externa que suministre el requerimiento de cada actuador, lo que conlleva a que exista la necesidad de un circuito eléctrico de potencia que permita acoplar las señales de control entregadas por el Arduino con los actuadores.

3.1.4 Circuito de acoplamiento

Como se mencionó anteriormente, la potencia entregada por el Arduino Uno no es suficiente para alimentar a los actuadores, por lo tanto es necesario el uso de otra fuente, lo que conlleva a que se deba diseñar un circuito que permita acoplar las señales de control con los actuadores alimentados por la fuente externa. A demás debe contener los componentes necesarios para el acondicionamiento de señal de los sensores, así como conectores que permitan conectar y desconectar fácilmente tanto las señales recibidas del arduino, como los sensores y actuadores.

Entre los factores eléctricos a tener en cuenta para el diseño del circuito están parámetros en los ventiladores que requieren una alimentación de 5V dc – 170 mA, controlados por una señal on-off de 0-5V dc, la resistencia consume un voltaje máximo de 5V y una corriente máxima de 1A cuya señal de control es regulada mediante PWM, mientras que para para el led se empleara el voltaje entregado directamente por el pin del Arduino Uno. Teniendo en cuenta esos factores se diseña el circuito de la Figura 9.

En el circuito de la Figura 9 se desarrolla la parte de potencia, donde se emplean transistores 2N2222 puestos a trabajar en la zona de corte y saturación, para los ventiladores (Figura 9. Izquierda). La selección de estos transistores está dada por la corriente de colector máxima soportada, que como se muestra en las especificaciones dadas en la hoja de datos es de 0.8A, corriente suficiente para poner en funcionamiento los ventiladores.

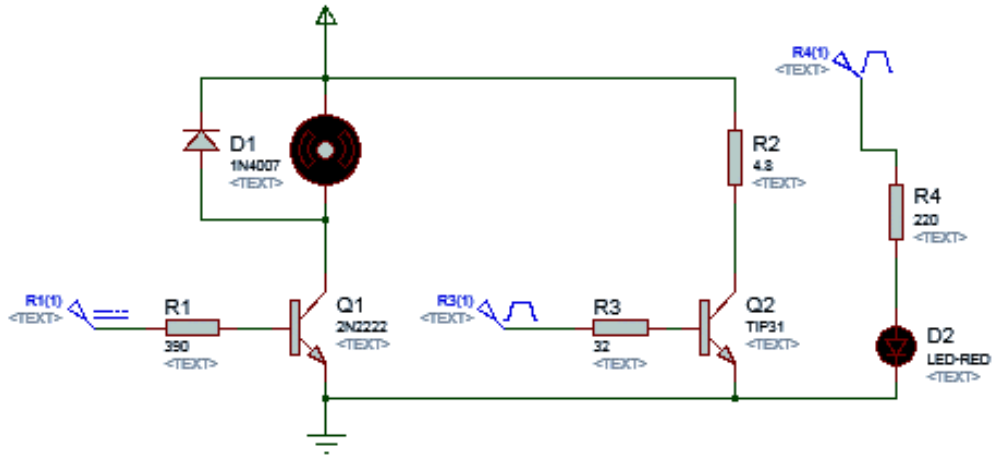
Para colocar los transistores en zona de corte y saturación es suficiente colocar una resistencia en la base, la cual delimita la corriente necesaria para que el transistor se active o desactive, empleando la ecuación 6 y usando los valores dados por la hoja de datos de ganancia de corriente *hfe=35* y voltaje consumido entre base y emisor *Vbe=0.7,* y los valores de voltaje de entrada y consumo dados por las condiciones del circuito *Vi=3V, Ic=0.17A,* se tiene que el valor teórico de la resistencia de base es de 473Ω.

Aunque los ventiladores tienen integrado su propio circuito de control y regulación, se colocan unos diodos en paralelo 1N4007 para protección contra del circuito en dado caso que se conectaran otro tipo de ventiladores.

Para el acople entre la señal de control de temperatura y la resistencia (Figura 9. Centro), se emplea un transistor Tip31, debido a que la corriente empleada es mucho mayor que para los ventiladores. Según la hoja de especificaciones soporta una corriente máxima de colector 3A en dc y 5A mediante modulación de pulso, lo cual se ajusta a los requerimientos planteados. El modo de trabajo, igualmente, es *on-off,* debido a que la señal de control esta modulada por ancho de pulso. Remplazando en la ecuación 6 los valores dados por la hoja de datos *hfe=10, Vbe=1.8V,* la corriente que pasa a través de la resistencia *Ic=1A* y el voltaje de control *Vi=5,* se obtiene que la resistencia de base es 32Ω

Finalmente para conexión del led (Figura 9. Derecha) es suficiente conectar una resistencia de 220Ω en serie para evitar sobretensiones sobre el diodo.

Figura 9. Circuito electrónico de acoplamiento



Una vez obtenidos los circuitos base de potencia, se unen en un esquemático junto con la polarización de los sensores para crear un circuito impreso que permita conectar y desconectar fácilmente los componentes desmontables como sensores y actuadores.

3.1.5 Módulo de comunicación

Se trata de la tarjeta que permite el envió de información a la web por medio de comunicación inalámbrica. Teniendo en cuenta que la tarjeta controladora es el Arduino Uno y los diferentes productos ofrecidos para utilizar con esta tarjeta, se selecciona como módulo de comunicación la tarjeta Arduino WiFi Shield.

La tarjeta Arduino WiFi Shield permite la conexión a internet de la tarjeta Arduino Uno a través de la es especificación 802.11b y 802.11g. Se conecta a una placa Arduino a través de wire-wrap, lo que permite apilarlo en la parte superior de la placa sin modificar en ningún momento la disposición y forma de conexión de los pines. Tiene una ranura que permite ingresar tarjetas micro-SD, que se puede utilizar para almacenar archivos para enviar a través de la red.

Debido a que debe comunicarse de manera física con la tarjeta principal, los pines 4, 7, 10, 11, 12 y 13 del Arduino UNO son dedicados para esta tarjeta, por lo cual no deben ser usados como entradas o salidas en el programa del Arduino (Arduino, 2014).

Para poder operar hace uso de 5V que son proporcionados por la placa Arduino Uno, tiene un mini puerto USB para la actualización del firmware, se comunica con la tarjeta principal a través del puerto SPI y tiene un conector tipo FTDI que permite la depuración serial del WiFi Shield.

Actualización del firmware del Arduino WiFi Shield

Como se indicó anteriormente el módulo WiFi Shield de Arduino permite comunicar de forma inalámbrica la tarjeta Arduino con otros dispositivos, sin embargo, para que funcione correctamente es necesario actualizar el firmware que trae de fábrica, pues con este solo permite detectar y conectarse a una redes WiFi, pero no permite el envió ni la recepción de información. Para llevar a cabo la actualización es necesario seguir los siguientes pasos (jesusvilche.wordpress.com, 2013):

1. Verificar el funcionamiento del WiFi Shield con alguno de los ejemplos que se encuentran en la página oficial de Arduino
2. Descargar los archivos de los firmware de la pagina https://github.com/arduino/Arduino/tree/master/hardware/arduino/firmwares (link proporcionado por la página oficial de Arduino)
3. Encontrar los archivos "WiFi_dndl.elf" y "WiFiHD.elf" ubicados dentro de WiFiShield/Binary y copiarlos en la carpeta de instalación de Flip (C:\Program Files (x86)\Atmel\Flip 3.4.5\bin)
4. Poner el jumper del WiFiShield en On.
5. Conectar el WiFiShield al computador mediante un cable mini usb
6. Ir a Panel de Control/Administrador de dispositivos y actualizar el controlador para el WiFiShield (AT32UCXXXX) desde el botón examinar ubicando el directorio en el que descargo los archivos
7. Abrir la interfaz de comando cmd.exe
8. Digitar el siguiente commando *cd C:\Program Files (x86)\Atmel\Flip 3.4.5\bin> batchisp.exe device AT32UC3A1256 hardware usb operation erase f memory flash blankcheck loadbuffer WiFi_dnld.elf program verify start reset 0*
9. Esperar a que se encienda led azul del WiFiShield
10. Presionar el botón reset del WiFiShield
11. Digitar el siguiente commando *cd C:\Program Files (x86)\Atmel\Flip 3.4.5\bin> batchisp.exe device AT32UC3A1256 hardware usb operation erase f memory flash blankcheck loadbuffer WiFiHD.elf program verify start reset 0*
12. Presionar el botón reset del WiFiShield
13. Quitar el jumper del WiFiShield
14. Desconectar el WiFiShield

3.2 Diseño mecánico

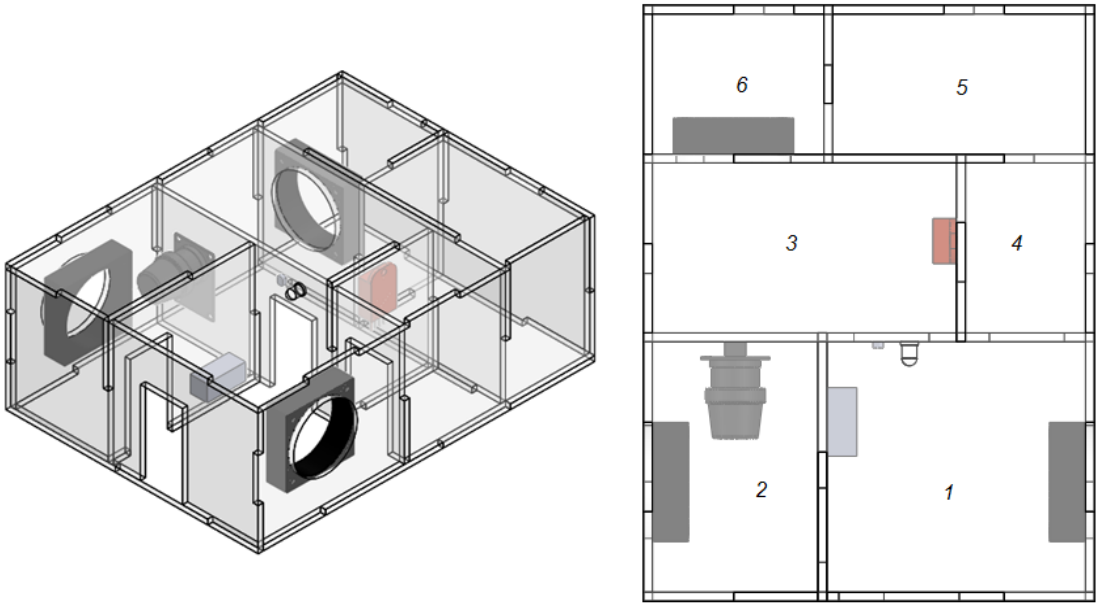
Una vez seleccionados y diseñados todos los elementos electrónicos, basándose en las dimensiones físicas, se procede al diseño mecánico. Se trata de una maqueta que representa la estructura de una casa, en donde se ubicaran cada uno de los sensores, actuadores y tarjetas electrónicas para el funcionamiento del sistema, teniendo en cuenta dimensiones de los dispositivos, y que permita simular el comportamiento de las variables descritas en capítulos. Debe cumplir la condición de ser totalmente desarmable, lo cual facilite su almacenamiento, minimizando espacio ocupado.

El material seleccionado para el desarrollo de la maqueta es acrílico transparente de tres milímetros de espesor, debido a que es un material que ofrece buena resistencia mecánica, estabilidad ante los agentes atmosféricos, como por ejemplo los rayos uv

que son los causantes de la degradación de los materiales expuestos a la intemperie, y además es livianos.

En la Figura 10 se muestran dos vistas del modelo, al lado izquierdo se observa el modelo en vista isométrica mientras que en el lado derecho se muestra la vista superior. Este modelo es desarrollado en la herramienta de diseño mecánico SolidWorks, que permite comprobar el modelo y corregir errores antes de realizar el corte de las piezas. En el modelo se incluyen componentes como sensores y actuadores, teniendo en cuenta las dimensiones presentadas en capítulos anteriores, y se selecciona en qué posición ira cada uno de ellos.

Figura 10. Modelo mecánico en SolidWorks



La maqueta tiene unas dimensiones de 15cm de ancho por 20cm de largo por 7.6 cm de alto. Cuenta con once piezas (sin incluir sensores y actuadores), todas con pestañas y orificos de 3mm de ancho, que le permiten ser totalmente desarmable y que la divide en seis zonas, cuatro de las cuales representan lugares basicos en toda vivienda mientras que las dos sobrantes son empleadas para la ubicación de la tarjeta controladora y la de potencia.

La zona No 1 representa la sala comedor de la casa e incluye los sensores y actuadores de iluminacion y temperatura, debido a que es en esta zona donde las personas comparten mayor parte del tiempo, donde se sientan a ver television, tomar sus alimentos, conversar etc, permitiendoles tener un confort en temperatura e

iluminacion. La zona No 2 representa la cocina y es alli donde se ubica el sensor de calidad de aire, debido a que es en esta zona donde se pueden presentar fugas de gas, permitiendo generar alertas que le brinden seguridad al usuario. La zona No 3 representa la alcoba ubicando ahí el sensor de humedad que le permita al usuario el monitoreo de la humedad relativa de esta zona. La zona No 4 representa el baño, en esta zona no se emplea ningun sensor. Las zonas 5 y 6 se emplean para la ubicación del Arduino Uno y la tarjeta de potencia, respectivamente.

3.3 Obtención modelo sistema y control PID simulado

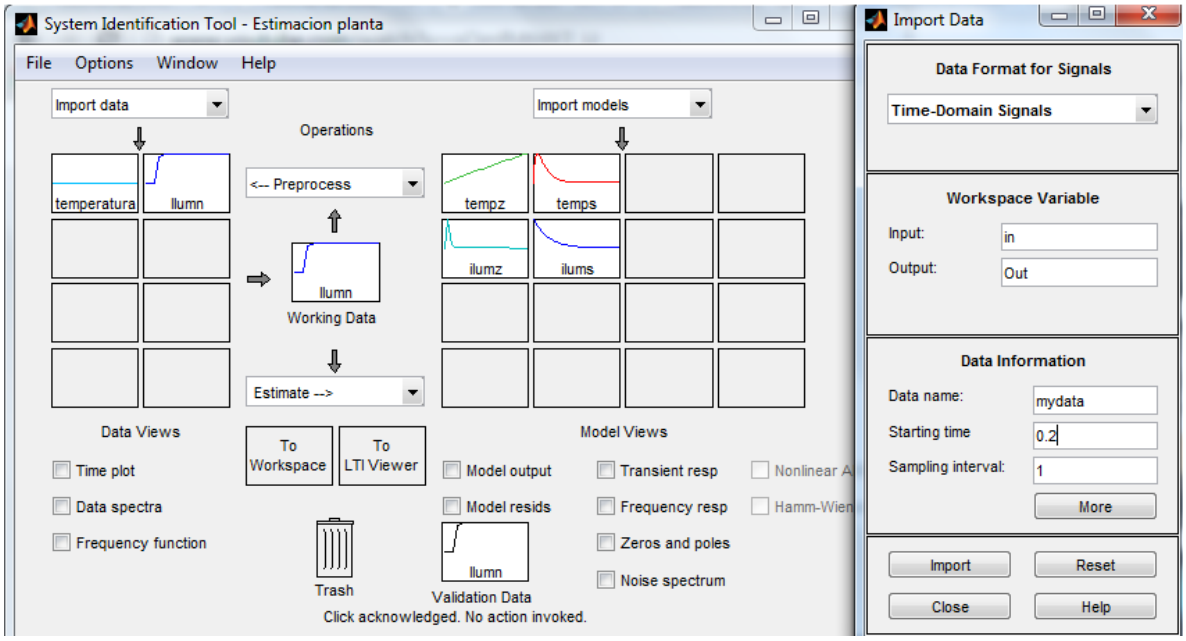
Una vez ensamblado el sistema, ubicando sensores y actuadores en su posición final, se procede a la identificación de los sistemas de manera experimental para la obtención de los modelos. El procedimiento de identificación es realizado en lazo abierto, donde se aplica una señal de entrada sobre el actuador (enviada directamente desde el Arduino Uno), y se obtiene la señal de respuesta real del sistema, para luego obtener el modelo matemático del sistema, utilizando la herramienta *ident* de matlab.

La estimación del modelo es realizada mediante la opción ***Transfer Function Models*** donde es seleccionado el número de polos y ceros con los que se quiere estimar el sistema, teniendo en cuenta la respuesta del sistema obtenida. También es necesario seleccionar dominio de la función de transferencia, tiempo discreto o tiempo continuo.

El procedimiento debe realizarse por separado para las dos variables a controlar, temperatura e iluminancia. Para las dos variables es aplicada una señal escalón con amplitud de 5V (máxima entregada por la tarjeta Arduino) y se obtienen una tabla de datos para cada variable, que contiene la señal aplicada, la señal de salida o respuesta del sistema y el tiempo en el que es tomada la muestra.

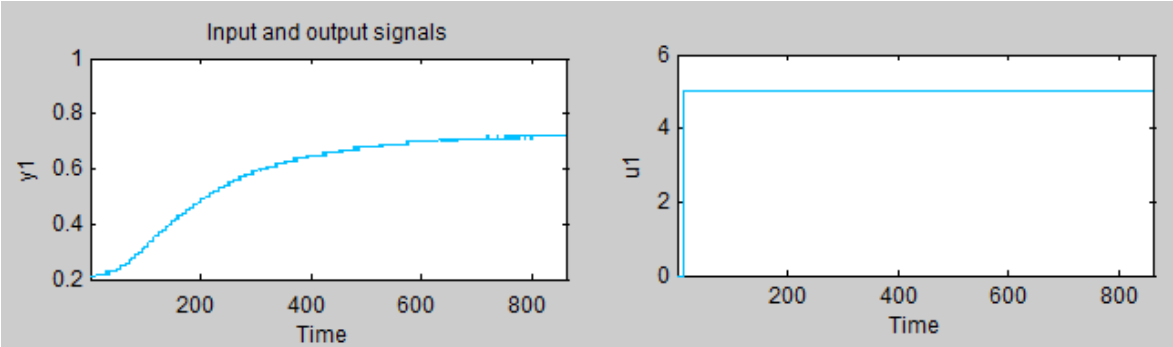
Para la variable temperatura se obtiene una tabla de 4319 datos con un tiempo de muestreo de 200ms, mientras que para la variable iluminancia se obtienen 865 datos con un tiempo de muestreo de 10ms. Los datos obtenidos son llevados al *workspace* de matlab, a las variables *in, out* y posteriormente son importados a la herramienta de identificación *ident (*Figura 11*)* para estimar el modelo correspondiente.

Figura 11. Obtencion del modelo mediante Ident



La gráfica de los datos importados para la variable temperatura se presenta en la Figura 12, donde se observa que el sistema se aproxima a un sistema de segundo orden, aspecto que es importante para seleccionar el tipo de función de transferencia que se quiere obtener, por lo cual se escoge que la función de transferencia estimada sea de segundo orden, con dos polos y un cero, en tiempo discreto, obteniendo la ecuación 10.

Figura 12. Respuesta variable temperatura

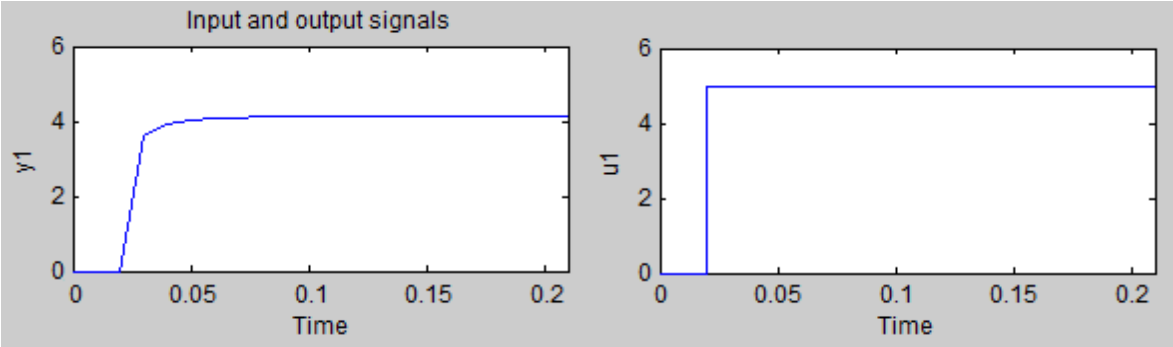


Al evaluar la adaptación del modelo obtenido con los datos medidos, se obtiene que el modelo se adapta en un 97.67%.

$$F(z) = \frac{7.0536e{-}07z}{z^2 - 1.996z + 0.994} \qquad (10)$$

La grafica de los datos importados para la variable iluminancia se muestran en la Figura 13, donde se observa un comportamiento exponencial, condición por la cual es seleccionado aproximar el sistema a uno de primer orden, con un polo y un cero, en tiempo discreto, obteniendo la ecuación 11.

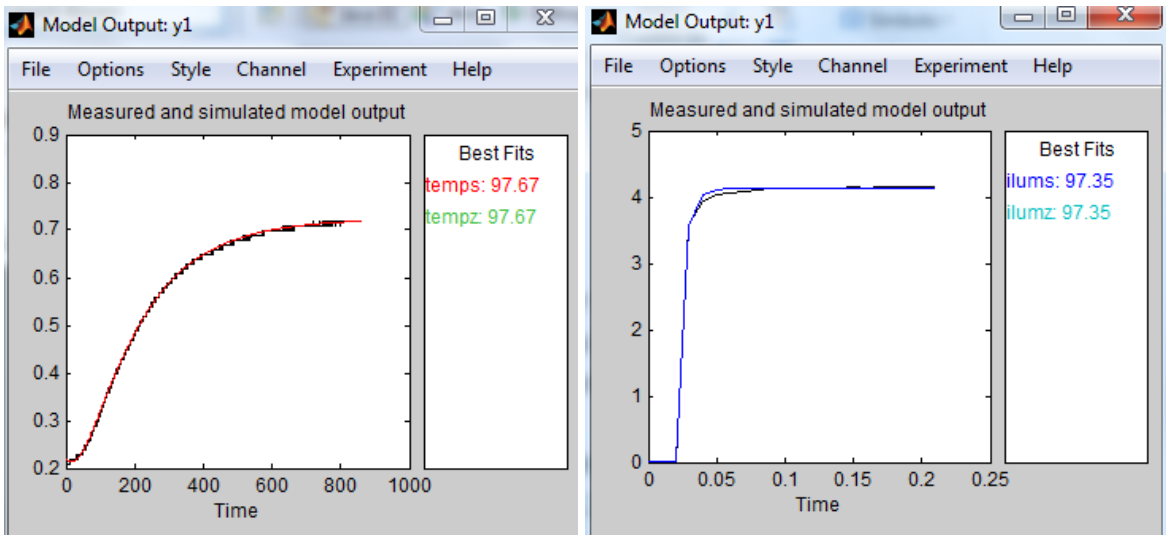
Figura 13. Respuesta Variable iluminancia



Al evaluar la adaptación del modelo obtenido con los datos medidos, se obtiene que el modelo se adapta en un 97.35%.

$$F(z) = \frac{0.7107}{z - 0.1379} \qquad (11)$$

Figura 14. Comparación modelos estimados con los datos reales

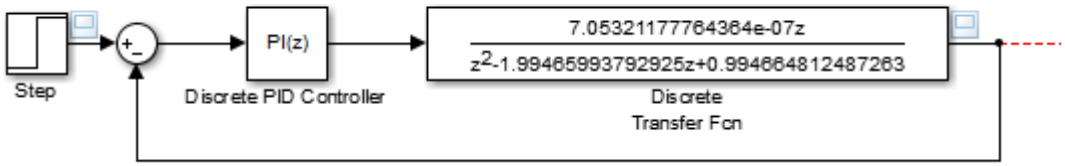


Una vez obtenidos los modelos se realiza la simulación, con el fin de obtener las constantes del PI para luego ser implementadas por programación en el arduino uno.

Para tal fin se emplean los bloques de función de transferencia discreto y el PID disponibles en simulink, los cuales facilitan la obtención de las constantes de control.

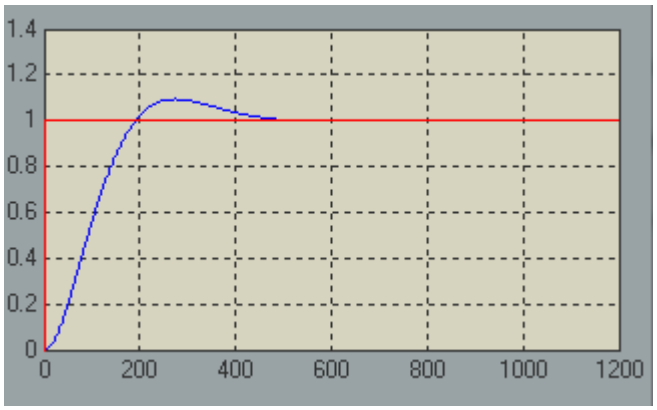
La simulación de lazo de control de temperatura se muestra en la Figura 15, donde se coloca la función de transferencia de la ecuación 10 correspondiente a la variable y el bloque PID, para el cual es establecido el tiempo de muestreo de 200 ms.

Figura 15. Simulación lazo de control temperatura



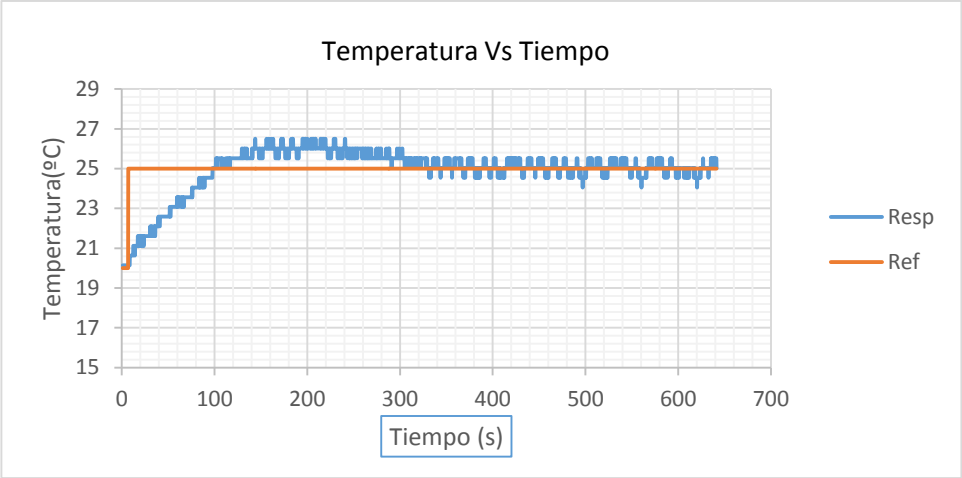
Dado un tiempo de establecimiento de 200 segundos un comportamiento subamortiguado con un máximo sobre impulso del 10% (Figura 16) y cuyo valor correspondiente a las constantes es P=0.006, I=1,1.213. Estos valores serán empleados posteriormente para ser implementados en el controlador PID digital en Arduino.

Figura 16. Respuesta escalón a la temperatura

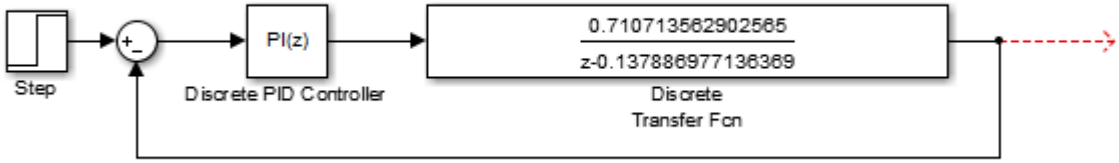


En la Figura 17 se presenta la respuesta real de la variable temperatura, donde se observa que es aproximada a la gráfica obtenida de la simulación, obteniendo una respuesta de un sistema sub amortiguado, con un sobre impulso menor al 10%, con un tiempo de establecimiento de 250 segundos.

Figura 17. Respuesta Real del sistema



La simulación de lazo de control de iluminancia se muestra en la Figura 15, donde se coloca la función de transferencia de la ecuación 11 correspondiente a la variable y el bloque PID, para el cual es establecido el tiempo de muestreo de 10 ms.

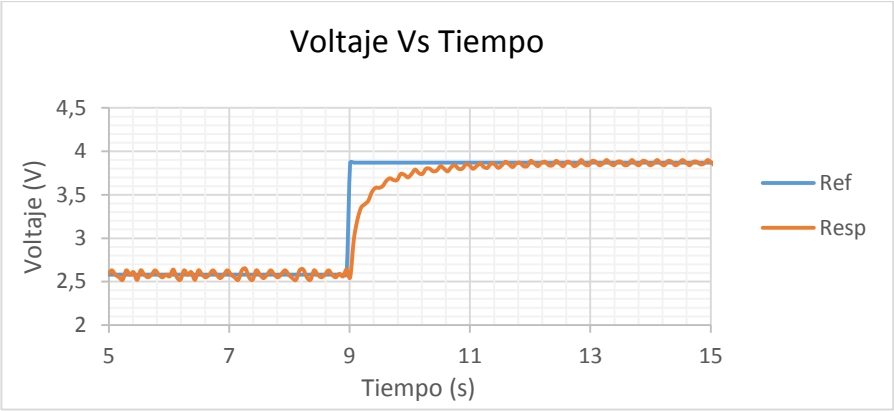


Para un tiempo de establecimiento de 1 segundo se obtiene un comportamiento exponencial (Figura 18) con las constantes P= 13.1159, I= 0.08858.

Figura 18. Respuesta escalón a la iluminación

En la Figura 19 se presenta la respuesta real de la variable iluminancia, donde se observa obteniendo una respuesta exponencial con un tiempo de establecimiento de un segundo, la cual es muy aproximada a la repuesta obtenida en la simulación Figura 18.

Figura 19Respuesta real Iluminancia



3.4 Configuración dispositivo en Xively

Antes de realizar las tareas de programación para la tarjeta controladora y el dispositivo de monitoreo, es necesario realizar la configuración del dispositivo en la web, que servirá como interfaz entre la tarjeta de control y el software, el cual se encargara de almacenar y trasmitir los datos entregados por cada dispositivo.

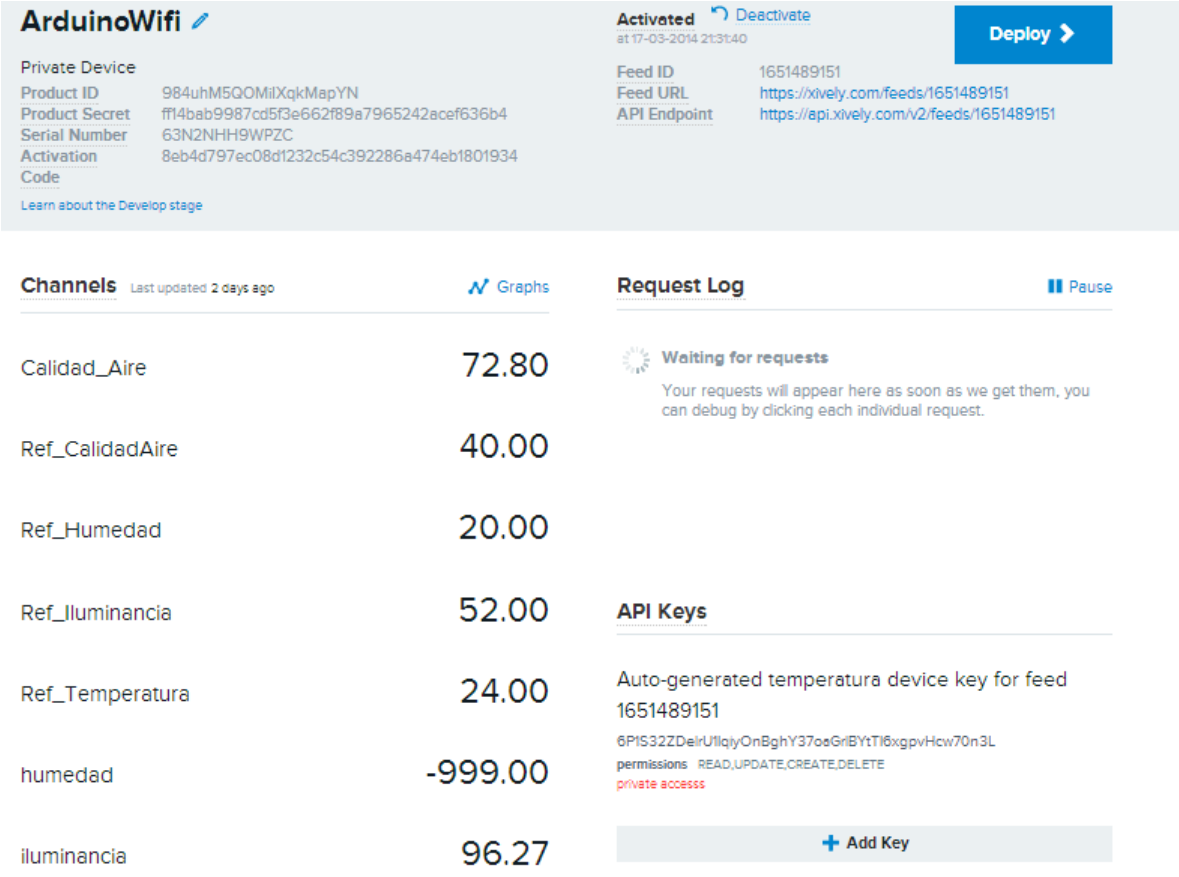
Para poder configurar el dispositivo, como primera medida debe crearse una cuenta de usuario en la página oficial de Xively, la cual permitirá tener acceso privado a los dispositivos creado. Posteriormente en la pestaña *develop* se agrega un nuevo dispositivo, al cual se le da el nombre y el cual tendrá todas las variables a manipular.

Una vez creado el dispositivo (Figura 20), la plataforma, automáticamente le asigna un código numérico de identificación llamado *feed id,* y una clave o llave alfanumérica propia de ese dispositivo llamada *Api Key.* Estos códigos son de vital importancia, ya que son los códigos de identificación en la red y serán necesarios para conectar los equipos que se deseen a este dispositivo.

Finalmente se crean las variables deseadas, agrupadas o identificadas como *datastream*, que corresponderán a los datos a trasmitir entre dispositivos y que se irán modificando vez que se haga un llamado al dispositivo desde cualquier otro equipo. Un aspecto a tener en cuenta es que el mismo nombre utilizado en la

plataforma Xively debe ser empleado en cualquier otra plataforma, conservando símbolos, mayúsculas y minúsculas, porque de lo contrario la comunicación no se realizara.

Figura 20. Dispositivo web en la plataforma xively



Una vez creado el dispositivo y nombrado las variables, se podrá establecer la comunicación con cuantos dispositivos se desee.

3.5 Programación tarjeta controladora

Como se indicó en la sección 2.2.5 y 3.1.1 como tarjeta de control se emplea una tarjeta Arduino Uno, la cual usa su propio entorno de programación llamado Arduino que es descargable gratuitamente de su página oficial. Las tareas a realizar por la tarjeta están dadas por el muestreo de las señales, el acondicionamiento de las señales (si es necesario), él envió del valor escalar de la variable a la web, la recepción de los valores de referencia desde la web, la ejecución de los comandos de control, el

envió de los valores o estados correspondientes a cada puerto y el inicio de la operación nuevamente, en el orden mostrado en la Figura 21.

La parte inicial del programa, ***configuraciones iniciales,*** comprende las primeras líneas del programa junto con la función ***void setup().*** En las primeras líneas se realiza la importación de librerías que serán empleadas en el desarrollo del programa siendo necesarias las siguientes:

1. WiFi, SPI. Empleadas para la comunicación entre el modulo y la tarjeta arduino, así como la transmisión de información a través de comunicación WiFi.
2. Xively. Permite configurar y ajustar la información de acuerdo a los requerimientos establecidos por la empresa para la transmisión de información, así como determinar si la comunicación está disponible y se ha realizado de manera correcta.
3. DHT. Brinda varias funciones que facilitan la lectura de los datos entregados por el sensor de temperatura RTH03.
4. PID. Versión para arduino que permite implementar fácilmente un controlador PID, que como único requisito indispensable es conocer el valor de las constates correspondientes. Puede ser empleado junto con el módulo *pwm* para entregar la señal de control directamente en un puerto.

En las siguientes líneas se declaran variables necesarias para la conexión a la red que le dará acceso a internet a la tarjeta, para ello se emplea las variable *ssid* que contiene el nombre de la red a conectarse, y la variable *pass* que contiene la contraseña de la red en caso de que esta la tenga. Luego se declaran los parámetros necesarios para establecer la conexión entre la el sistema físico y la plataforma xively, para ello se usan las variables *xivelykey* y *xivelyfeedId* que contienen los identificadores del *"aparato"* dados al configurar un nuevo dispositivo en la plataforma *xively.*

Luego es necesario crear constantes tipo char que contengan los mismos nombres de las variables creadas en la plataforma xively, con el fin de enlazar la comunicación entre los dos dispositivos sin errores. Una vez establecidas las constantes se enlazan todas en un arreglo llamado *datastream.* Tambien son creadas las variables que almacenaran los valores leídos o que serán colocados en los puertos del Arduino, y variables empleadas para trabajo interno del programa. Se definen las estructuras PID estableciendo el nombre de las variables de entrada, salida y los valores de las constantes correspondientes para cada controlador.

Por ultimo en la función ***void setup()*** se inicia una comunicación serial para que sirva de monitoreo en caso de que la tarjeta controladora sea conectado a un computador,

se define el modo en que trabajara los puertos seleccionados (salida, ya que entrada está dado por defecto), se activa el trabajo de los PID y por ultimo establece la comunicación WiFi entre el Arduino y la red especificada anteriormente.

Una vez completadas las configuraciones se inicia un ciclo infinito en el que se lleva a cabo todas las tareas de lectura de señales, transmisión de datos, y control de las variables. Para la lectura, control y evaluación de condiciones de cada una de las variables se crea una función, con el objeto de facilitar la programación, las cuales son llamadas desde la función principal ***void loop()*** la cual ejecutara el ciclo infinito.

Debido a las variables a ser sensadas son cuatro, se crean cuatro funciones que realizan los procesos de ***muestreo y acondicionamiento,*** donde se hace lectura del puerto correspondiente, se lleva el valor de lectura a su nivel de voltaje correspondiente (teniendo en cuenta la resolución del conversor análogo digital del Arduino UNO) y sumando o restando valores que lo lleven a que las condiciones iniciales del sistema sean cero, empleando las ecuaciones 7, 8 y 9 corespondientes a los sensores. El ***procesamiento de datos,*** donde se evalúa la acción controladora y se toma una decisión y por último él ***envió de información a los puertos,*** donde se lleva la señal de control a una salida PWM o un puerto ON-OFF según sea el caso para controlar el sistema.

El proceso llevado a cabo para cada una de las variables viene descrito por los siguientes procedimientos (omitiendo el paso de muestreo y conversión a voltaje):

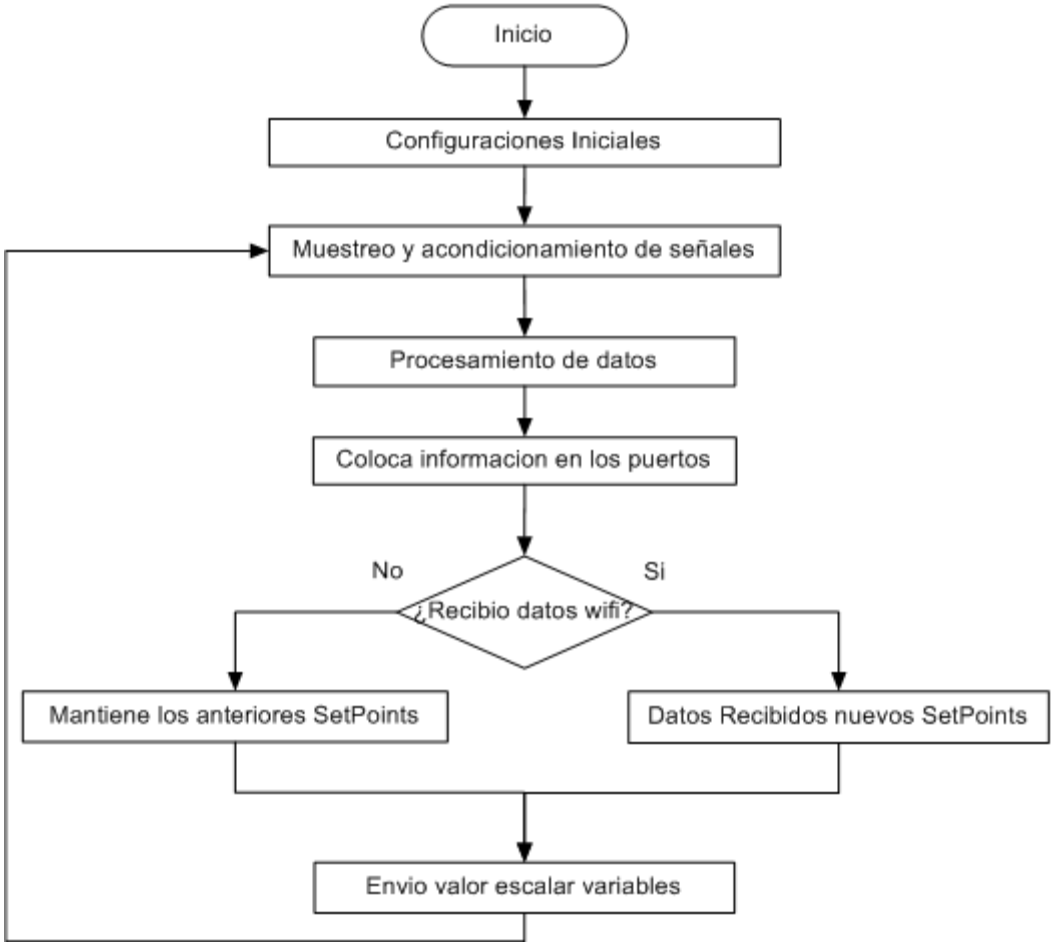
Para la variable temperatura y dadas las condiciones su sensor (sección 3.1.4) en el circuito de acoplamiento, para obtener el valor de temperatura en función del voltaje vasta con multiplicar su valor por 100, obteniendo así la temperatura real. Luego es evaluada la condición de realización de cálculos del PID, si la condición es afirmativa entonces se procede a enviar la señal al puerto análogo de temperatura (teniendo en cuenta la resolución del conversor Digital –Análogo) para que se ejecute la acción de control sobre la resistencia, y finalmente se evalúa la condición de temperatura actual mayor a la temperatura de referencia para encender un ventilador si la condición es cierta o de lo contrario apagarlo.

Debido a que la variable humedad únicamente es sensada, en esta función simplemente se realiza la lectura del sensor mediante la función *read* de la librería *DHT* evaluando si el valor se ha recibido o ha ocurrido algún error, omitiendo procesos de conversión análoga digital ya que el sensor arroja el valor escalar de humedad y temperatura.

Para la variable calidad de aire, se acondiciona la señal para que las lecturas del sensor se lleven a una escala de porcentaje, donde pueda determinarse la calidad de aire, que sea apta para el consumo humano. Luego es evaluada la condición que si el valor leído es mayor a un valor de referencia dado, se activara una señal, para que active el ventilador que permita agregar aire limpio al lugar para mejorar el nivel de la calidad de aire

Para la variable iluminación es necesario llevar el nivel de voltaje leído en condiciones de poca luminosidad a cero (sección 3.1.4) para lo cual se debe restar un nivel de tensión de 0.27, y finalmente llevar el valor leído a una escala de porcentaje que permita determina el porcentaje de iluminación. Posteriormente es evaluada la condición de realización de cálculos del PID, para enviar la señal al puerto análogo la señal de control para que se ejecute la acción sobre el led.

Figura 21 Diagrama de flujo tarjeta controladora



Una vez realizado el muestreo y ejecución de control para cada una de las variables, es necesario realizar la lectura y escritura de los datos WiFi para permitir la supervisión remota. Para ello es necesario preguntar si fueron recibidos datos en el puerto WiFi, si la condición es cierta es necesario obtenerlos y sustituir los datos anteriores por estos nuevos (para los set Point) y posteriormente realizar el ***envio escalar de cada una de las variables*** que han sido medidas anteriormente.

3.6 Programación Android

Como se ha planteado en el proyecto la supervisión de las variables se realizara a través de un dispositivo móvil. Para ello es necesario realizar el desarrollo de software para un sistema operativo determinado, para este caso Android. Son diferentes los lenguajes de programación que pueden ser utilizados para el desarrollo de aplicaciones en Android (sección 2.2.1). Tomando como referencia que la página oficial de Android para desarrolladores basa sus guías y ejemplos en java, el entorno seleccionado para el desarrollo de la aplicación es Eclipse.

3.6.1 Configuraciones iniciales del entorno de desarrollo

Para llevar a cabo el desarrollo de aplicaciones para Android en Eclipse es necesario instalar unos paquetes y/o herramientas, que brinden las características necesarias a la hora de programar (librerías, simuladores, etc), tales como el SDK y el JDK.

Sdk (Sotfware Development Kit)

http://developer.android.com/sdk/index.html
http://www.androidpit.es/sdk-android

Como su nombre lo indica es el paquete de desarrollo de software, con el cual se desarrollan aplicaciones y se ejecuta el emulador de Android para cualquier versión, viene dado para el lenguaje de programación Java. Proporciona las bibliotecas API y las herramientas de desarrollo necesarias para agilizar el desarrollo de la aplicación Android.

Instalación y configuración

Descargar e instalar JDK (herramientas de desarrollo para Java), Java SE 7u45 o más actual desde la página:
http://www.oracle.com/technetwork/java/javase/downloads/index.html

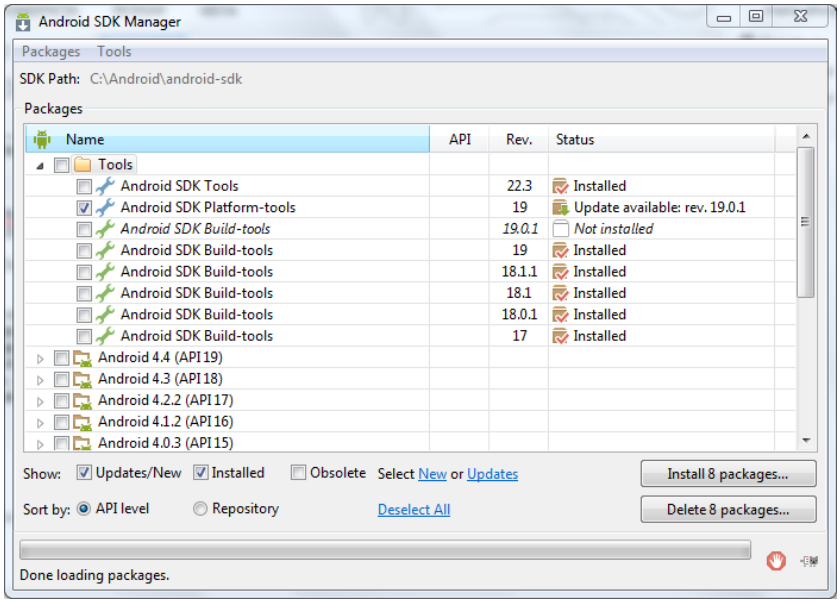
Descargar e instalar SDK de android desde la página: http://developer.android.com/sdk/index.html (tener en cuenta el directorio de la instalación)

Luego de la instalación se ejecutara automáticamente el SDK Manager, donde se seleccionaran las versiones de las API a instalar

Instalar las herramientas y las API deseadas, tener en cuenta que un móvil, por ejemplo, con una API 19 ejecuta aplicaciones realizadas para API 10, mientras que un móvil con API 10 no ejecuta aplicaciones realizadas para API 19.

Descargar Eclipse de la página http://www.eclipse.org/downloads/ (pagina oficial de Eclipse) y extraer los archivos descargados.

Figura 22. SDK Manager de Andorid

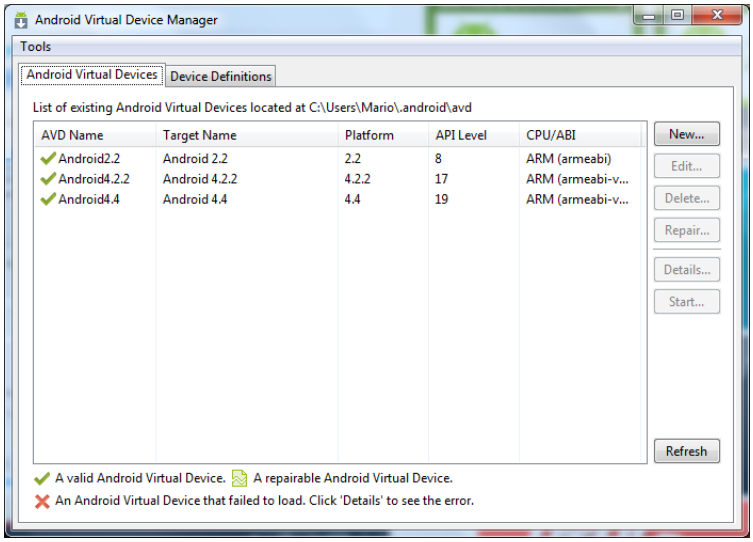


Instalar Plugin mediante el siguiente procedimiento: help - Install New Sotfware-Add-Name: “Nombre” - Link: https://dl-ssl.google.com/android/eclipse/ (pagina para actualización de plugins) - seleccionar Developer Tools – Next – Aceptar términos – Finalizar.

Configurar las preferencias de eclipse: window – preferences – Android – SDK Location: “directorio de instalación del SDK en pasos anteriores” – ok.

Configuración de la máquina virtual (simulador): ejecutar el AVD Manager, New… - dar los parámetros deseados (tipo de procesador, versión del SO, espacio de memoria, tamaño de pantalla…) – ok.

Figura 23. AVD manager de Android



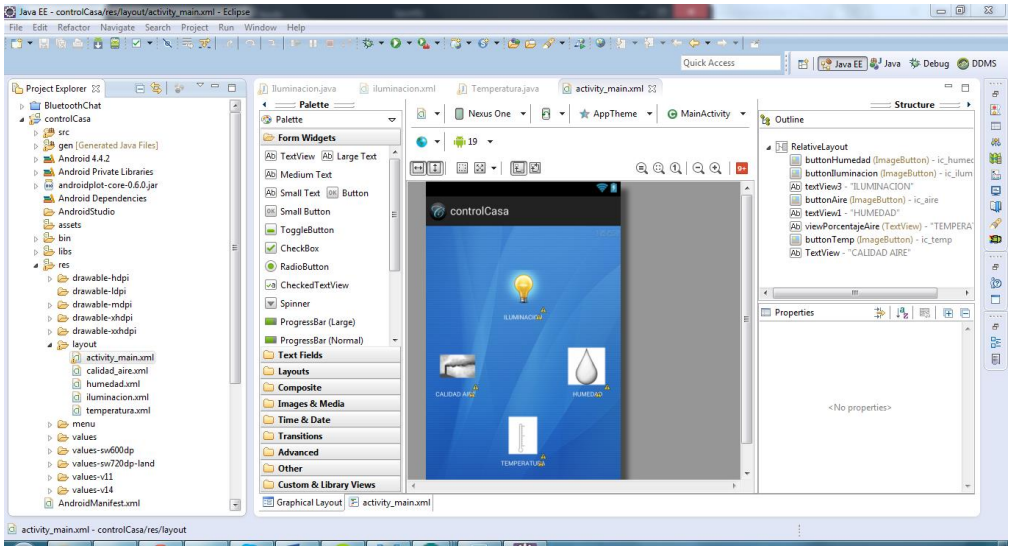
Luego de creado el simulador, seleccionarlo y oprimir el botón *Start*… donde se iniciara la simulación del equipo (Figura 24).

Figura 24. Simulación de un dispositivo Android con el AVD

3.6.2 Secciones y estructura de un proyecto Android

Para el desarrollo de la aplicación es importante tener conocimiento del entorno de trabajo, para lo cual se hace una breve descripcion a continuacion, partiendo de la siguiente imagen.

Figura 25. Ventana principal Eclipse - Android

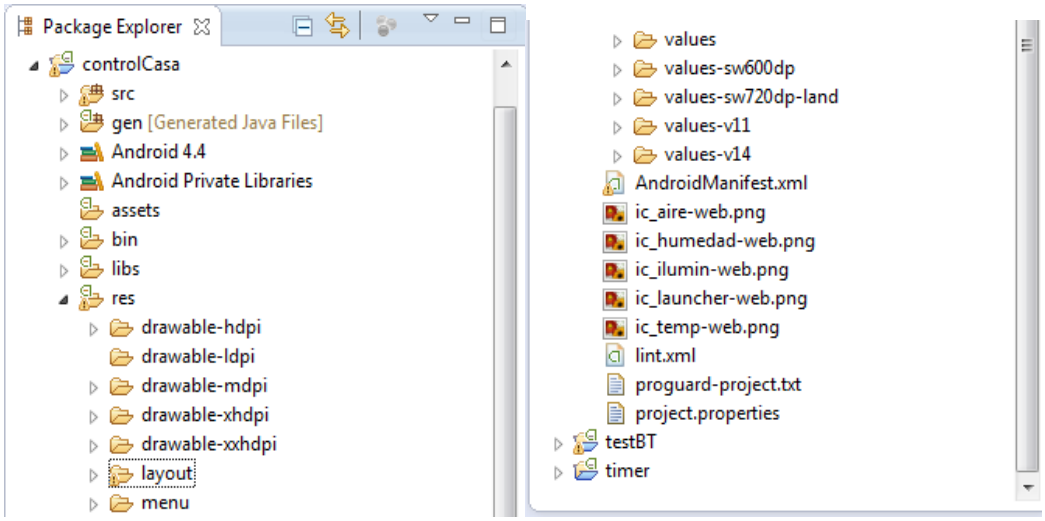


En la parte superior se encuentra la barra de herramientas, desde donde se crean, guardan, compilan y ejecutan las aplicaciones realizadas (entre otras opciones más).
En la parte izquierda y de color amarillo, se puede ver el proyecto creado con un conjunto de subcarpetas en estructura árbol que contienen los códigos fuente, librerías, imágenes, manifiestos y permisos necesarios para el desarrollo de la aplicación.

La parte central debajo de la barra de herramientas permite navegar en las diferentes ventanas que han sido abiertas para el trabajo en el proyecto.
Por último la parte sobrante es el área de trabajo y es donde se desarrollara toda la aplicación, bien sea la parte gráfica, que estará desarrollada en código xml, y la parte de funcionabilidad y eventos desarrollada en java.

Debido a que el proyecto es organizado en subcarpetas (Figura 26) es útil tener conocimiento de que se tratan y como se emplean las más utilizadas de ellas, con el fin de hacer el desarrollo de código fácilmente, la descripción de las más utilizadas se describe a continuación:

Figura 26. Carpetas del proyecto en estructura árbol



La carpeta ***src*** contiene las clases de java con todo el código desarrollado para darle interactividad a la aplicación, es allí donde se crean y modifican las clases de java.

La carpeta ***res*** es donde se modificara toda la parte visual de la aplicación, iconos, ventanas, texto, etc. Para ello usa las siguientes subcarpetas:

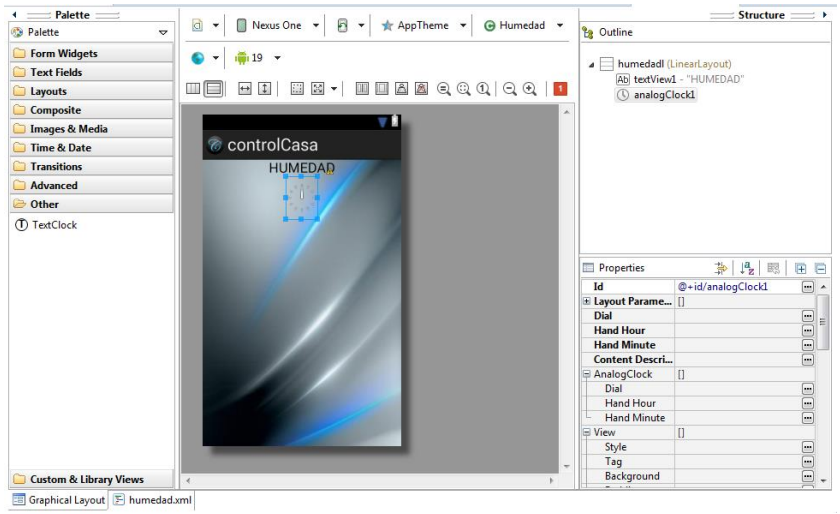
- **Drawable:** son cinco carpetas (drawable-hdpi, -mdpi, -ldpi, -xhdpi, -xxhdpi) donde se colocan todas las imágenes, fondos e iconos que tendrá la aplicación. Dependiendo del tamaño y la calidad de la imagen se seleccionara en que carpeta almacenar
- **Layout:** es allí donde se incluyen cada una de las ventanas que tendrá la aplicación, tienen extensión xml.
- Otras carpetas que no se modifican para el desarrollo de la aplicación.

Además contiene el manifiesto (*Andorid Manifest*), donde se declarara los permisos y las herramientas a usar del dispositivo móvil (cámaras, sensores, bluetooth, WiFi, etc) en etiquetas xml.

Edición de Layouts

La edición de las ventanas de una aplicación se hacen en lenguaje xml y viene para ser editado de dos maneras, gráficamente y por código.

Figura 27. Paleta y parte grafica de la aplicación



Para la edición de la ventana de forma gráfica, el *sdk* tiene una paleta (Figura 27) con herramientas usadas frecuentemente en aplicaciones que se subdividen entre las siguientes formas:

- Form Widgets: botones, cuadros de texto (solo para visualización), check box, radio buttons, barras de progreso
- Text fields: cuadros de texto (para la edicion por parte del usuario), incluyen diferentes formatos, números, direcciones, contraseñas, nombres, etc.
- Layouts: permiten la organización de la pantalla por secciones, permiten desarrollar una ventana más organizada. Contiene las opciones de organización en grillas, horizontales, verticales, entre otras.
- Composite: desplazamientos horizontales y verticales a través de la pantalla, menús auxiliares.
- Images & Media: para incluir imágenes, botones con imágenes y videos
- Time & Date: Calendario, reloj y cronometro.

Cada elemento se puede incluir a la aplicación arrastrándolo hasta donde se desea colocar. Para modificar las propiedades de un objeto, se da clic derecho sobre él y se selecciona la propiedad a modificar, es usual cambiar el "id" del objeto para reconocerlo de manera fácil en el desarrollo de la aplicación al ser usado en una clase.

La edición por código se maneja por medio de etiquetas xml, donde hay que tener conocimiento de que etiquetas se pueden incluir y como es su estructura, todo lo descrito en la parte grafica se puede realizar a través de etiquetas, editar propiedades (id, tamaños, ubicaciones, fuentes, colores, etc).

Para le edición por código, ya que muchas veces facilita la creación de la interfaz, es necesario tener en cuenta como se crea un objeto en *XML (*Figura 28*)*. Un objeto en *XML* se declara de la siguiente manera:

<NombreObjeto

Todas las propiedades del objeto/>

Donde nombre del objeto puede ser: TextView, Image, ImageButton, Button, PorgressBar, etc.

Figura 28. Edicion de la parte grafica en codigo XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/humedadl"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    android:background="@drawable/wallpaper">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="HUMEDAD"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <AnalogClock
        android:id="@+id/analogClock1"
        android:layout_width="52dp"
        android:layout_height="70dp" />

</LinearLayout>
```

Las propiedades de un objeto se declaran mediante el comando:

android:*propiedad=”parámetros”*

Dentro de las propiedades se puede incluir: id, background, textcolor, layout_width, layout_height, entre otros.

3.6.3 Desarrollo de la aplicación

Una vez adquirido un conocimiento básico del entorno de programación y las herramientas que se pueden usar, se procede al desarrollo de la aplicación.

La aplicación tiene por desarrollo una interfaz de visualización, donde el usuario puede observar el estado de cuatro variables en su casa, temperatura, iluminación, calidad de aire y humedad, y definir *setpoints* o puntos de referencia en dos de ellas, dependiendo de sus necesidades o gustos. También debe incluir un protocolo de

comunicación WiFi, el cual le debe permitir recibir y enviar todos los datos a la plataforma web Xively, para integrar la aplicación con el sistema de control.

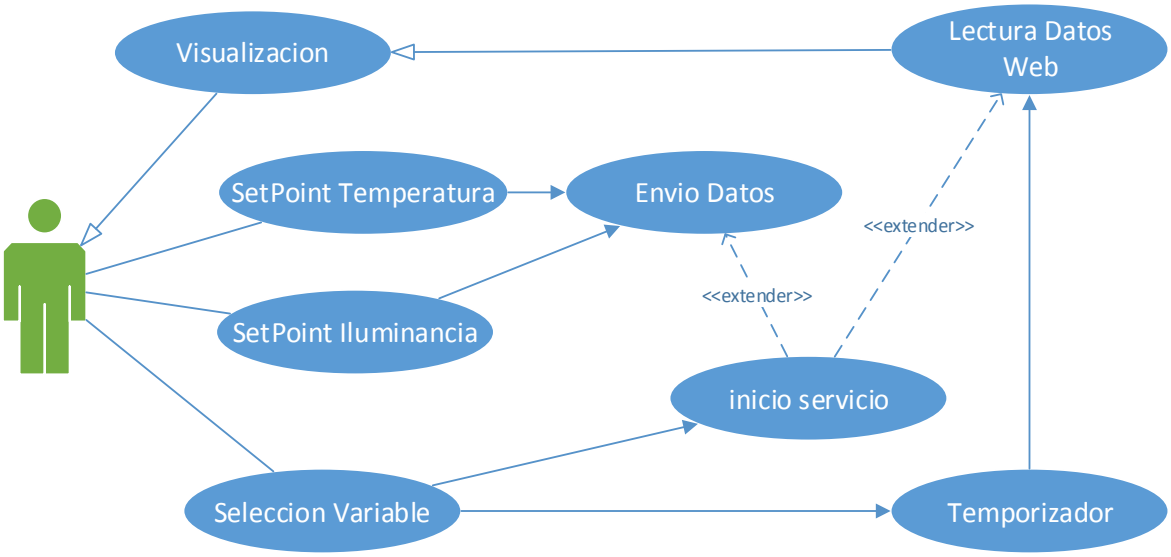
Cada una de las variables se visualiza en una pantalla diferente, por lo tanto se tienen cuatro pantallas de visualización de variables y una pantalla principal, para la selección de la variable donde debe definirse los elementos a utilizar en la interfaz, para darle un aspecto que la permita identificar fácilmente.

La temperatura está dada en grados Celsius y esta presentada a través de una barra de nivel, además se muestra un cuadro de texto que indica el valor escalar de la temperatura en el estado presente. Las variables iluminación y la humedad vienen dadas en porcentaje, se muestran por medio de una barra de estrellas que se colorean dependiendo del porcentaje de iluminación o humedad presente, respectivamente. La variable calidad de aire está dada en porcentaje y se muestra en una barra de estrellas y en un cuadro de texto que muestra el valor porcentual actual de la calidad de aire. Por último se desarrolla la ventana principal que contiene los iconos que enlazan a la ventana de cada variable.

La Figura 29 presenta el diagrama de caso de uso de la aplicación, donde se puede visualizar la forma como el usuario puede operar el sistema. Según los mencionado en párrafos anteriores y como se presenta en la figura, el usuario tiene la opción de seleccionar alguna variable para abrir la ventana correspondiente, así como establecer los puntos de referencia para las variables a controlar.

Una vez seleccionada la variable se inicia el servicio de comunicación, que permite realizar el enlace entre la plataforma Xively y la aplicación. A su vez es iniciado el temporizador el cual permite realizar la captura de información desde la plataforma web de forma periódica, para mostrarlos en la aplicación y ser visualizados por el usuario.

Figura 29. Diagrama de caso uso de la aplicación

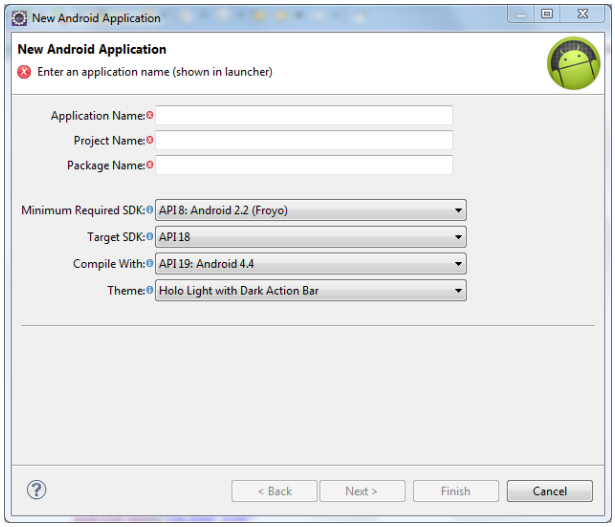


Por otra parte al usuario establecer un punto de referencia (*set point*) de alguna de las dos variables, el programa realiza él envió de los datos a la plataforma web, que como requisito previo ha sido iniciar el servicio para establecer el enlace.
El procedimiento que se describe a continuación es el empleado para llevar a cabo el desarrollo de la aplicación con las características mencionadas anteriormente.

El primer paso es crear un nuevo proyecto en Eclipse a través de los comandos

File->New->Android Application Project

Figura 30. Creación proyecto en Eclipse



Una vez se abre la ventana de la Figura 30 se debe dar el nombre a la aplicación y seleccionar el nivel API para el cual se desea desarrollar la aplicación.

->Next->Next

Se puede modificar el icono con el que quiere identificar la aplicación, o dejar el que incluye el SDK por defecto

->Next->Next->Finish

Una vez finalizada la creación del proyecto aparecen las carpetas descritas anteriormente. Posteriormente se incluyen todas las imágenes a utilizar en las diferentes ventanas de la aplicación, fondos y botones, en la carpeta *drawable-mdpi.*

Luego en la carpeta *layouts* es necesario crear cuatro archivos *android xlm,* a parte del existente activity_main, que corresponden a la ventana de cada variable. Para el *activity_main* se incluyen cuatro *ImageButton* y a cada uno se le coloca la imagen relacionada a su variable, ademas se coloca un *textView* para cada imagen, que facilita al usuario identificar el tipo de variable ademas de colocar un fondo a la pantalla. En los archivos xml correspondiente para cada variable se incluyen barras de estado, *textView*, textField y los demás componentes requeridos.

Un aspecto a tener en cuenta para el desarrollo de aplicaciones android viene dado que al lanzar una nueva aplicación se crea un nuevo hilo de ejecución, su importancia radica en que se encarga de atender todos los eventos de los diferentes componentes, ejecutando los métodos onDraw(), onCreate(), onKeyDown() entre otros, que son los que permiten la interacción entre la aplicación y el usuario. Esto es importante ya que si el programa requiere realizar tareas de trabajo intensivo (por ejemplo descargar datos de internet), que tomen bastante tiempo en ser ejecutadas, pueden bloquear la aplicación hasta que finalice la tarea, eliminando las posibilidades de realizar cualquier otra tarea y si el tiempo supera más de 5 segundos el sistema lanzara un mensaje de advertencia “la aplicación no responde” (Universidad Politecnica de Valencia, 2011).

Una solución para este problema es crear un nuevo hilo que ejecute la tarea de trabajo intensivo, así no se bloquea el hilo principal y se pueden seguir atendiendo los eventos de usuario. También es importante tener en cuenta que la única forma de modificar la interfaz de usuario es desde el hilo principal, por lo tanto si se intenta

alterar desde otro hilo la ejecución de la aplicación lanzara error. Una forma de realizar un cambio a la interfaz desde otro hilo es emplear el método

```
runOnUiThread(new Runnable() {
  @Override public void run() {
            /*Acciones en hilo principal*/
      }}
```

que ejecuta las acciones implementadas en el método run() sobre el hilo principal estando dentro de un hilo auxiliar.

Teniendo en cuenta lo anterior y una vez creada la interfaz gráfica se inicia con el desarrollo de las clases en java, para darle funcionabilidad a los cuadros de texto, botones y barras de progreso que se han incluido. Para tal fin se crea la misma cantidad de clases como de ventanas creadas en xml, y para facilitar el trabajo se les da el mismo nombre.

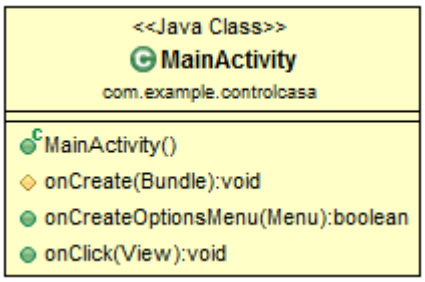
El primer objeto desarrollado en java es el ***mainAvtivity*** (Figura 31) para el cual es necesario implementar un *OnClick()* con el fin de capturar eventos de toques o pulsaciones sobre la pantalla con el objeto de lanzar la ventana que el usuario haya seleccionado y lanzar la actividad correspondiente.

Uno de los comandos requeridos para enlazar los objetos del archivo xml en el objeto java es

```
setContentView(R.layout.activity_main);
```

Donde se identifica con cual ventana creada en la carpeta layouts se va a trabajar, para este caso la ventana *activity_main*.

Figura 31. Diagrama Clase MainActivity



Luego se declaran las variables a usar, incluyendo los objetos implementados en el *layout* con el fin de reconocerlos y darles la funcionabilidad necesaria, esto se hace dependiendo del tipo de objeto y dando el nombre o el *id* igual o similar al que se le dio en xml.

Así por ejemplo

```
ImageButton buttonTemp;
```

es un icono que se llamara buttonTemp. Una vez declarado es necesario asignarle un objeto del layout que se hace mediante el comando *findViewById(R.id.**ID**),* donde ID es el identificador dado en el archivo *xml.* Por ejemplo

```
buttonTemp=(ImageButton)findViewById(R.id.buttonTemp);
```

A buttonTemp se le asingo un icono creado en xml con el nombre de *buttonTemp*. Este procedimiento es realizado con las otras cuatro variables, para así completar un total de cuatro iconos identificados de la ventana principal.

Una vez establecidas las variables que indicaran cada icono o variable es necesario agregarles la tarea a realizar al ser seleccionados, pera ello se hace uso de la función *onClick* la cual captura el objeto sobre el cual se ha realizado la acción para posteriormente ser evaluado mediante una sentencia de control *switch* que inicia la ventana del objeto requerido mediante las dos siguientes líneas de instrucción

```
Intent int1= new Intent("objeto_a_iniciar");
startActivity(int1);
```

Donde *objeto_a_iniciar* corresponde a la clase se requiere iniciar, declarando un nuevo obtjeto tipo *Intent* y posteriormente el inicio. Finalmente como son cuatro iconos los que deben realizar la misma acción, lo único que se debe hacer es sustituir el objeto a iniciar.

Posteriormente son desarrollados los objetos correspondientes a cada variable, donde las instrucciones serán prácticamente las mismas, con la diferencia de los elementos empleados para la visualización. El diagrama de flujo de la Figura 26 indica la lógica de programación que se llevara para cada variable y que se explica en párrafos posteriores.

En las *configuraciones iniciales* se definen cuadros de texto, botones, barras de progreso y demás componentes empleados en la interfaz, también se definen las

variables que permiten identificar el dispositivo creado en la plataforma xivley (los mismos creados en la programación del Arduino), el *ApiKey* y el *FeedId* así como las variables que almacenaran los datos a enviar o recibir a través de la web, teniendo las mismas consideraciones que se tuvieron para el Arduino. Luego es necesario ***iniciar el servicio*** según las especificaciones dadas por la plataforma web xivley, donde se emplea la clase de la Figura 32, que emplea los métodos *onServiceConnected* y *onServiceDisconnected,* que inician o finalizan el enlace entre la aplicación y la plataforma.

Figura 32. Diagrama clase HttpServiceConnection

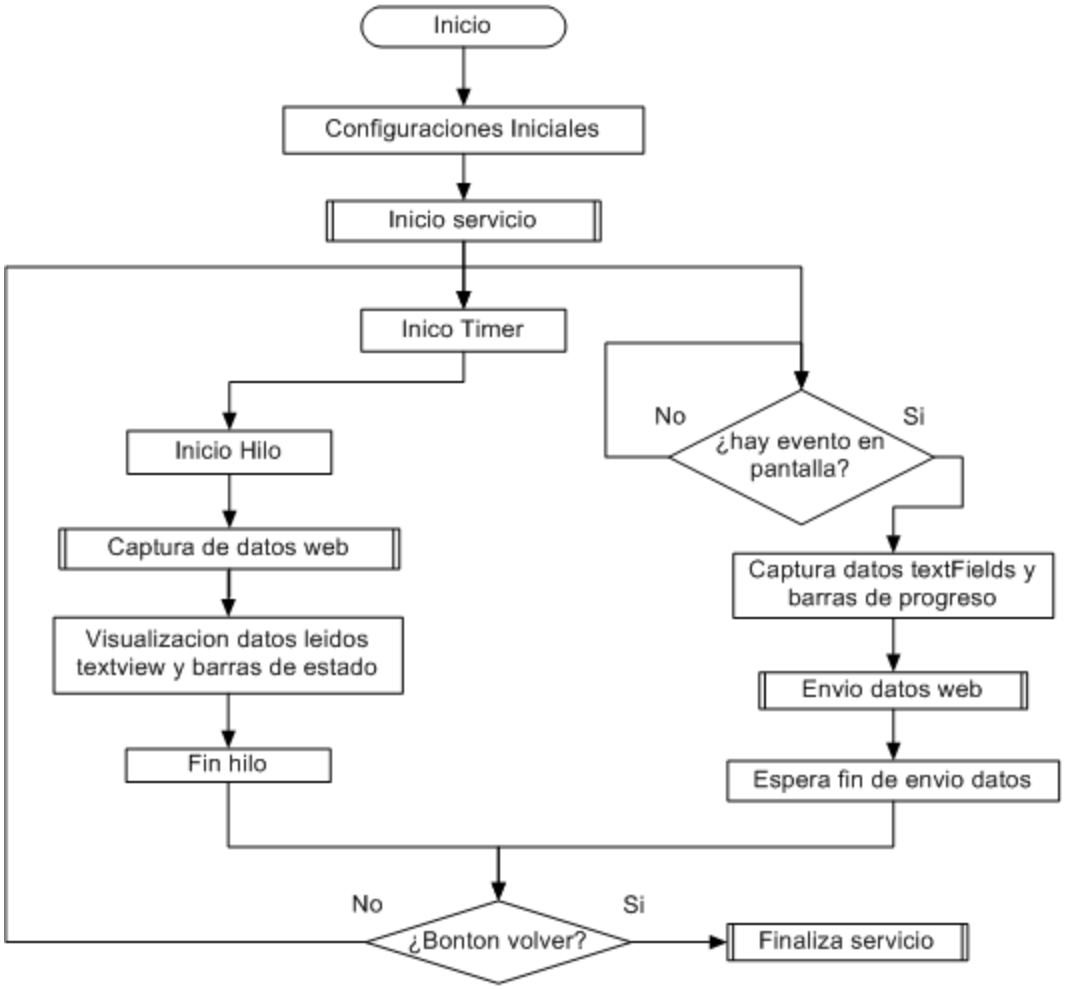
Debido a que la transmisión de datos a través de la web es lenta, es necesario hacer el uso de un hilo auxiliar que realice la captura de datos periódicamente, con el fin de que el proceso sea automático (que el usuario no pulse ningún botón), para mostrarlos posteriormente en la pantalla. Para hacer ese proceso periódico se hace uso de un temporizador (Timer) que inicia la ejecución del hilo cada determinado tiempo y empleando el hilo para realizar la tarea de captura de datos web, asegurando así que el programa se ejecute con regularidad.

Para la creación del temporizador es necesario implementar ellas siguientes líneas de código

```
TimerTask timerTask = new TimerTask() {
        @Override
        public void run()
        {
            //tarea a realizar
        }}
    };
Timer timer = new Timer();
timer.scheduleAtFixedRate(timerTask, int desde_cuando, int cada_cuanto);
```

donde TimerTask es un clase que emplea el método run() para ejecutar el código deseado o la tarea a realizar, posteriormente se crea la entidad timer y se pone en marcha (*InicioTimer*), y por último se define cada cuanto tiempo se requiere que se ejecute la tarea y a partir de instante de tiempo.

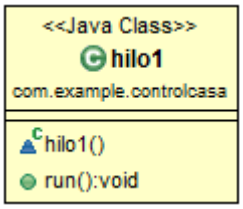
Figura 33. Diagrama flujo de objeto para las variables



Para este caso en específico la tarea a realizar es la ejecución del hilo de captura de datos web, con lo que vasta llamar a la entidad *thread.start()* la cual realiza el ***inicio del hilo***. Cabe resaltar que esta tarea se realiza paralelamente a las funciones principales de la aplicación que comprende la captura de información de la pantalla.

Para la ejecución del hilo, es creada la clase hilo1 (Figura 34), que emplea el método *run()* para ejecutar la captura de información desde la plataforma web.

Figura 34. Diagrama de clase Hilo

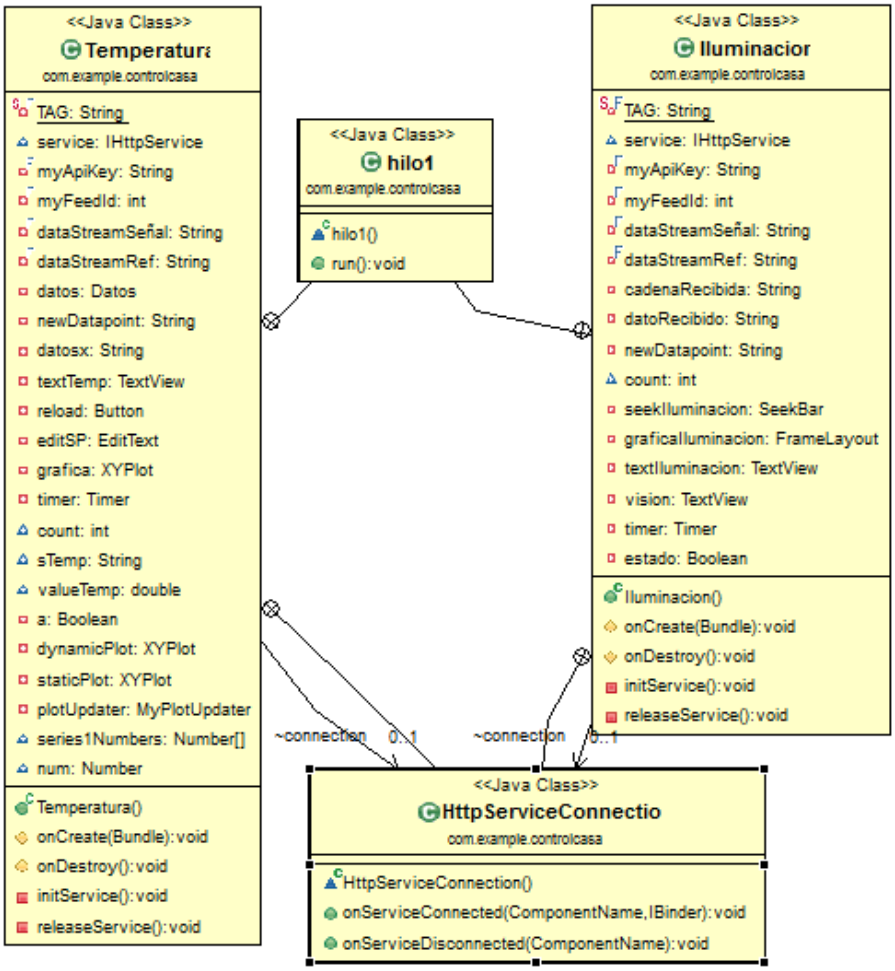


Una vez finalizada la captura de datos web es necesario mostrar la información deseada en la pantalla, para lo cual es necesario emplear el método *runOnUiThread* que se ejecuta en el hilo principal de la aplicación, como se mencionó en párrafos anteriores, permitiendo modificar cuadros de texto y barras indicadoras de estado de acuerdo a los datos recibidos y posteriormente el ***fin del hilo***.

Mientras tanto en un proceso paralelo a este hilo se capturan los cambios efectuados en la pantalla ejecutando métodos OnCreate(), OnClick() entre otros, capturando información desde los textField y barras de progreso para establecer los set points que serán enviados a la web, que al igual que la captura de datos web toma bastante tiempo el envió de los datos web, esta vez alterando el hilo principal, por lo cual el usuario debe esperar a que se envíen los datos para realizar una nueva modificación. Finalmente se evalúa si se ha seleccionado el botón volver, tarea que hace por defecto el hilo principal, para finalizar el servicio de comunicación o volver a ejecutar las tareas de captura y envió de datos.

La diferencia entre una ventana y otra está dada esencialmente en el objeto de visualización y captura para cada una de las variables. Configurando los objetos por las clases que se describen a continuación:
La clase *Temperatura* (Figura 35), emplea el método de captura del punto de referencia para la variable a través de un EditText (editSP) numérico y el metodo onClickListener, que captura el valor numérico digitado para enviarlo a través de la web luego de seleccionar el EditText, mientras que la forma de visualización del estado de la variable es dada por un TextView (textTemp) donde es colocado el valor de la variable a través del metodo *setText.*

Figura 35. Diagrama clase Temperatura e iluminación

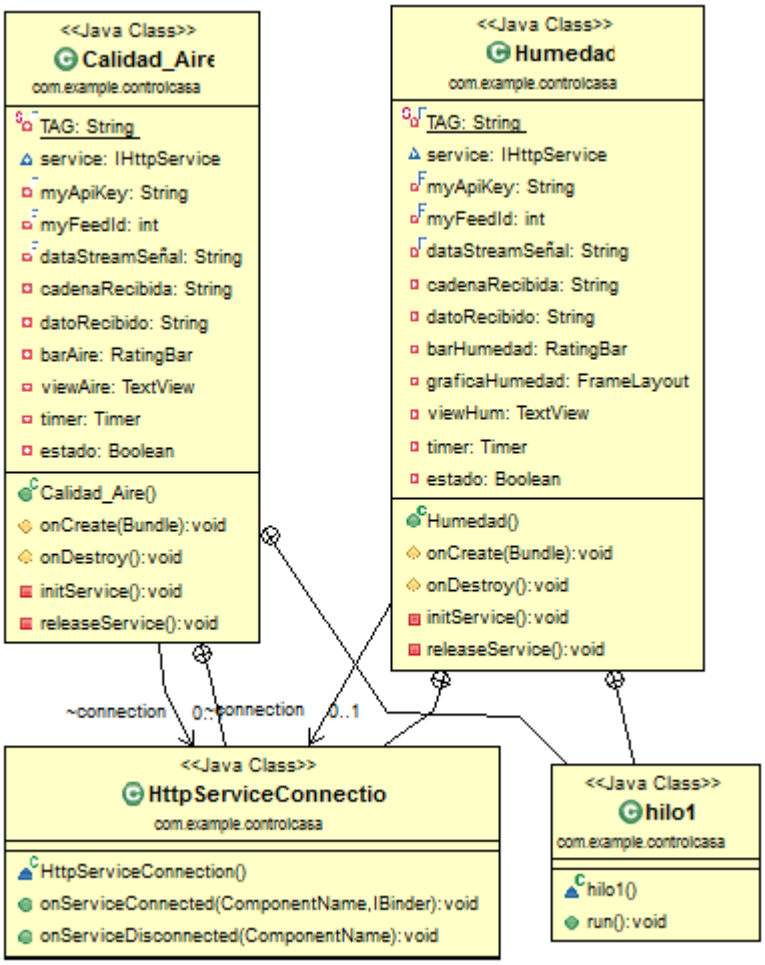


En la clase iluminación (Figura 35) es empleado un TextView (textIluminacion) para la visualización del estado de dicha variable, mientras que el objeto de captura de la referencia está dado por un objeto *seekbar* (seekIluminacion) para el que es capturado el estado en porcentaje mediante el método *setOnSeekBarChangeListener* que a su vez puede ser implementado a través de tres estados de la barra onProgressChanged que captura el valor cuando se presenta un cambio en el progreso de la barra, onStartTrackingTouch que captura el valor cuando se inicia el desplazamiento del progreso de la barra y el método onStopTrackingTouch que captura el valor al finalizar el desplazamiento de la barra. El progreso de la barra también es presentado en un TextView (vision), lo que le permite al usuario verificar el punto de referencia que será enviado.

Para la clase Calidad_Aire (Figura 36) es empleado un TextView (viewAire) para mostrar el estado de la variable de forma numérica, mientras que se emplea un objeto

RattingBar (barAire) que permite visualizar el estado de la variable de forma gráfica a través de cinco estrellas representando la equivalencia al valor numérico. Para modificar el progreso del objeto es suficiente emplear el método setReating que recibe como parámetro el valor en punto flotante de la cantidad de estrellas que se desean rellenar.

Figura 36. Diagrama Clase calidad de Aire y Humedad



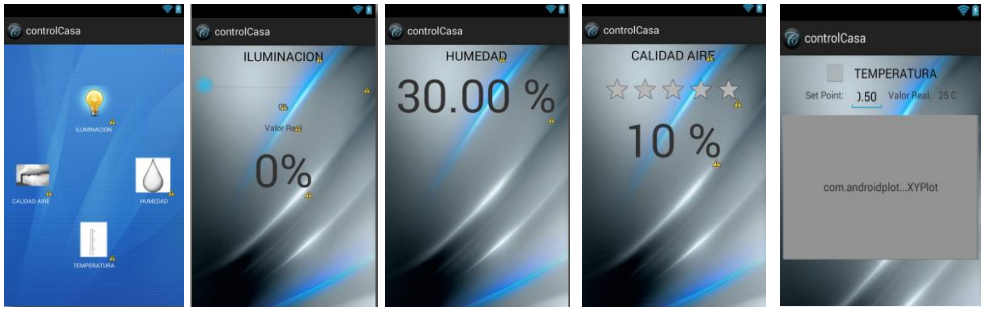
La clase Humedad (Figura 36) utiliza un TextView (viewHumedad) para mostrar el valor actual se emplea Debido a que las variables de humedad y calidad de aire no serán controladas, se emplean solo elementos de visualización, para tal fin se hace el uso de dos objetos, un TextView para mostrar el valor numérico de la variable y un ratingBar que permite la visualización de forma gráfica en forma de estrellas el estado de la variable equivalente al valor numérico. Para colocar el progreso correspondiente del objeto ratingBar es suficiente emplear el método setRating que recibe como

parámetro el valor en punto flotante de la cantidad de estrellas que se desean rellenar.

Las variables *myApiKEy, myFeedId, dataStreamSeñal, dataStreamRef,* que se presentan en todas las clases, contienen los parámetros dados por el dispositivo creado en Xively, lo que permite realizar el enlace entre la plataforma web y la aplicación desarrollada. Además se puede observar que todas usan la clase *HttpServiceConnection* e *Hilo,* como se describió en párrafos anteriores.

En la Figura 37 se muestran las ventanas desarrolladas, cuyas características y funcionalidades se describieron en los párrafos anteriores.

Figura 37. Ventanas de la aplicación



4. Conclusiones

Se logró desarrollar un sistema que contiene cuatro tipos de sensores, temperatura, iluminancia, calidad de aire y humedad, de los cuales dos son controlables (temperatura e iluminancia), mientras que para la variable calidad de aire es activado un sistema de ventilación en caso de que se detecten gases contaminantes.

El sistema es conectado a internet, empleando la plataforma gratuita Xively, que permite realizar envió, recepción y almacenamiento de información entregada por los sensores y condiciones establecidas por el usuario, permitiendo enlazar un sistema de monitoreo y control remotamente a través de internet.

Para el monitoreo remoto de las variables se desarrolló una aplicación para dispositivos con sistema operativo Android, que emplea la conexión a internet para realizar un enlace con la plataforma Xively, permitiendo visualizar en cuatro ventanas el estado de cada una de variables y permitiendo al usuario establecer puntos de referencia para las variables que son controladas.

Es importante tener en cuenta que las condiciones de funcionamiento de la planta deber ser lo más aproximadamente posible a la condiciones de modelamiento y diseño de control, pues de lo contrario el sistema no se comportará de la forma deseada. Para la variable iluminación es necesario que el sistema se encuentre totalmente oscuro, para que el bombillo pueda realizar correctamente la terea de control, pues de lo contrario el sistema responderá de forma inesperada y quedara funcionando como dispositivo de medida. Para la variable temperatura es necesario que el sistema esté cerrado herméticamente, ya que alguna fuga en el sistema hace que el sistema no responda de la forma esperada.

El la velocidad de respuesta del sistema está restringido por las características de comunicación dadas por la red de internet ya que el proceso de envío y recepción de datos a través de internet toma tiempos relativamente altos, lo que hace que si el usuario se encuentra en el mismo lugar de las variables a controlar, pareciera que el sistema tiene un retardo cuando son modificados los puntos de referencia, en especial en la variable iluminancia.

5. Bibliografía

Android.com. 2014. The android source code. [En línea] 2014. [Citado el: 23 de 01 de 2014.] http://source.android.com/source/index.html..

Arduino. 2014. arduino.cc. [En línea] 2014. http://www.arduino.cc/es/; http://arduino.cc/en/Main/arduinoBoardUno; http://arduino.cc/en/Main/ArduinoWiFiShield.

Areny, Ramon Pallas y Valdes, Fernando E. 2007. *Microcontroladores. fundamentos y aplicaciones con pic.* Barcelona : MARCOMBO, S.A., 2007. ISBN: 9788426714145.

aulaclic. 2005. aulaclic.es. [En línea] 2005. [Citado el: 15 de 03 de 2014.] http://www.aulaclic.es/articulos/wifi.html.

Brown, Theodore L., y otros. 2004. *Quimica: La ciencia central.* Mexico : Pearson Education, 2004. pág. 15. ISBN 970-26-0468-0.

Cengel, Yunus A. 2012. *Termodinamica.* s.l. : McGraw-Hill Interamericana de España S.L, 2012. págs. 739-741. ISBN-10: 607150743X.

conex-lab.com. 2012. conex-lab.com. [En línea] 2012. [Citado el: 18 de 07 de 2014.] http://www.conex-lab.com/iot-platform.html.

definicionabc. 2008. definicionabc.com. [En línea] 2008. [Citado el: 15 de 03 de 2014.] http://www.definicionabc.com/tecnologia/ventilador.php.

Esteban Dávila Náder. 2014. elespectador.com. [En línea] 05 de 29 de 2014. [Citado el: 18 de 07 de 2014.] http://www.elespectador.com/noticias/actualidad/el-internet-de-cosas-articulo-495403.

Falcon, Jose A. Carballar. 2010. *Wi-fi: lo que se necesita conocer.* España : RC Libros, 2010. ISBN: 9788493776909.

Farrás, Juan Guasch. 2012. *Enciclopedia de la OIT, capitulo 45.* España : D - INSHT (Instituto Nacional de Seguridad e Higiene en el Trabajo) , 2012.

Ge, Wenqing; Lai, Hua; Qi, Mingjie; Cui, Linlin. 2012. *Design of Temperature and Humidity Monitoring Terminal System Based on Android.* [ed.] Engineering Design and Manufacturing Informatization (ICSEM), 2012 3rd International Conference on System Science. Kunming : IEEE CONFERENCE PUBLICATIONS, 2012. Vol. 1.

Heng, Iem, y otros. 2012. *A Unique Environmental Mobile Device for Detecting Hazardous Chemicals.* Brooklyn : IEEE CONFERENCE PUBLICATIONS, 2012. Global Humanitarian Technology Conference (GHTC), 2012 IEEE.

InstitutoSindicaldeTrabajo, Ambiente y Salud(ISTAS). 2007. *La prevencion de riesgos en los lugares de trabajo.* s.l. : ParaleloEdición S.A., 2007. págs. 35-51. ISBN: 84-607-3133-2.

iphoneandord.com. 2013. iphoneandord.com. [En línea] 06 de 04 de 2013. [Citado el: 23 de 02 de 2014.] http://iphoneandord.com/los-5-mejores-sistemas-operativos-para-celulares/.

jesusvilche.wordpress.com. 2013. jesusvilche.wordpress.com. [En línea] 10 de 2013. [Citado el: 10 de 03 de 2014.] http://jesusvilchez.wordpress.com/2013/10/20/arduino-leonardo-wifi-shield-windows-8/.
José Miguel Roca Chillida. 2013. Informe Tic Facil. *InformeTicFacil.com.* [En línea] 2013. [Citado el: 18 de 07 de 2014.] http://www.informeticplus.com/que-es-el-internet-de-las-cosas.
Marquez, Arnolfo. 2008. Sensores de humedad. [En línea] 2008. http://es.scribd.com/doc/74029800/Sensores-de-humedad.
Martínez, Francisco Javier Rey y Gómez, Eloy Velasco. 2005. *Calidad de ambientes interiores.* s.l. : Ediciones Parafino S.A, 2005. iSBN: 9788497325400.
medirtemperatura. 2013. medirtemperatura.com. [En línea] 2013. [Citado el: 05 de 03 de 2014.] http://medirtemperatura.com/sensor-temperatura.php.
Milton, M.A.A. y Khan, A.A.S. 2012. *Web Based Remote Exploration and Control System.* s.l. : IEEE CONFERENCE PUBLICATIONS, 2012.
Mohammad, B., y otros. 2013. *Portable wireless biomedical temperature monitoring system: Architecture and implementation.* Abu Dhabi, United Arab emirate : s.n., 2013.
Ojeda, Luis Thayer. 2013. *Monitoreo de energía por Internet.* Santiafo de Chile : Omilex, 2013.
Pantoja, Yuscy. 2012. slideshare. [En línea] 10 de 30 de 2012. [Citado el: 15 de 03 de 2014.] http://es.slideshare.net/Dabyus/sensores-de-gases.
quees.info. 2013. quees.info. [En línea] 2013. [Citado el: 17 de 06 de 2014.] http://www.quees.info/que-es-la-tecnologia.html.
R, Achkar, y otros. 2013. *Gas Ascerainment via Smartphone.* Beirut : American University of Science and technology, AUST, 2013.
R.A, Gonzales. 2013. *Sistema informático para la supervisión y control de un horno eléctrico desde dispositivos móviles con Android.* Barcelona : Universidad Oriente, 2013.
revistaitnow. 2014. revistaitnow.com. [En línea] 18 de 06 de 2014. [Citado el: 15 de 07 de 2014.] http://revistaitnow.com/2014/06/convergencia/smartphones-se-apoderaran-del-mercado-en-el-2016/.
Rita Conde. 2013. celulares.about.com. [En línea] 2013. [Citado el: 15 de 07 de 2014.] http://celulares.about.com/od/Smartphones/ss/Tecnologia-NFC-Descubre-Sus-Beneficios.htm.
Rodrigo, Javier Garcia. 2013. *Técnicas y procesos en instalaciones eléctricas.* s.l. : Ediciones Parainfo S.A, 2013. pág. 158. ISBN 10: 8497323343.
Rodríguez, María Álvarez, Artime, Irene Ferreira y A.B.A. 2012. *Dispositivos moviles.* Gijon, España : Universidad de Oviedo, 2012.
Santamaria, Eduardo. 1993. *Electronica digital y microprocesadores.* s.l. : Univ Pontifica Comillas, 1993. ISBN 8487840337.

Schwarz, Ronan, Dutson, Phil y Steele, James. 2011. *The Android Developer's Cookbook: Building Applications with the Android SDK.* s.l. : Pearson Education, 2011. ISBN-10: 0321897536.
Sensors empower the "Internet of Things". **Margery Conner. 2010.** s.l. : EDN Network, 2010. ISSN 0012-7515.
Thompson, D.R., y otros. 2013. *Adaptive sensing of time series with application to remote exploration.* Karlsruhe : IEEE CONFERENCE PUBLICATIONS, 2013.
Universidad Politecnica de Valencia. 2011. androidcurso.com. [En línea] 2011. [Citado el: 14 de 05 de 2014.] http://www.androidcurso.com/index.php/tutoriales-android/36-unidad-5-entradas-en-android-teclado-pantalla-tactil-y-sensores/271-hilos-de-ejecucion-en-android.
Weiss, Scott. 2002. *Handheld Usability.* United Kingdom : John Wiley and Sons Ltd, 2002. ISBN 0470844469.
Xataka. 2013. Huawei ya es el tercer fabricante de Smartphones, según Gartner. [En línea] 13 de 08 de 2013. [Citado el: 24 de 04 de 2013.] http://www.xatakamovil.com/mercado/huawei-ya-es-el-tercer-fabricante-de-smartphones-segun-gartner.
xively. 2014. xively. [En línea] 2014. https://xively.com/.