SIMULADOR DOMÓTICO PARA ENTRENAMIENTO Y APROXIMACIÓN A LA DOMÓTICA

ROBERT LEONARDO SALAZAR MANRIQUE

UNIVERSIDAD PILOTO DE COLOMBIA
FACULTAD DE INGENIERÍA MECATRÓNICA
BOGOTÁ D.C.
2015

SIMULADOR DOMÓTICO PARA ENTRENAMIENTO Y APROXIMACIÓN A LA DOMÓTICA

ROBERT LEONARDO SALAZAR MANRIQUE

MONOGRAFIA DE TRABAJO DE GRADO PARA OPTAR AL TITULO DE INGENIERIO MECATRONICO

Profesor Orientador
MS.C RUBÉN DARÍO HERNÁNDEZ BELEÑO PH.D ©

UNIVERSIDAD PILOTO DE COLOMBIA
FACULTAD DE INGENIERÍA MECATRÓNICA
BOGOTÁ D.C.
2015

Nota de aceptación:

El trabajo de grado, titulado "***SIMULADOR DOMÓTICO PARA ENTRENAMIENTO Y APROXIMACIÓN A LA DOMÓTICA***" elaborado y presentado por el estudiante **Robert Leonardo Salazar Manrique**, como requisito parcial para optar al título de Ingeniero Mecatrónico, fue aprobado por el Jurado Calificador.

Firma del presidente del jurado

Firma del jurado

Firma del jurado

Bogotá D.C. 30 de Enero de 2015

DEDICATORIA

Este trabajo con la totalidad del proyecto está dedicado a mis padres, Domingo y Angela, que me apoyaron y creyeron en mi incluso en esos momentos de flaqueza.

AGRADECIMIENTO

Agradezco a mis padres por el apoyo durante toda la carrera y durante el proceso de este proyecto.

Agradezco a mi hermana Adriana Salazar por la ayuda que me ha brindado cada vez que la he requerido.

Agradezco a mi tutor de tesis Rubén Hernández por su buena indicación durante el proceso que llevamos.

Agradezco a cada uno de mis compañeros y futuros colegas que colaboraron directa o indirectamente con este proceso.

Finalmente quiero agradecer el apoyo que durante el pregrado me brindo la facultad de ingeniería mecatrónica de la universidad Piloto de Colombia.

RESUMEN

Esta monografía muestra el proceso de desarrollo de un simulador domótico para ambientes estudiantiles, abarcando desde su concepción hasta su ejecución. Este simulador contendrá 3 partes fundamentales: el programa que ejecuta el simulador, una tarjeta que leerá sensores del mundo físico y un control remoto programado para Android.

Un simulador es una realidad creada con características de algún fenómeno, en este caso el simulador se crea con miras de emular visualmente una casa y poder usar variables del mundo real, y simular algunas.

El objetivo de este proyecto es crear un simulador de bajo costo capaz de correr variables en tiempo real y que alguien pueda interaccionar de manera simple, como en un videojuego. También se busca añadir elementos del uso actual de la domótica como un celular inteligente.

El simulador fue realizado en Unity3D®, la interfaz con los sensores se hizo a través de una tarjeta electrónica de ARDUINO®, el control remoto se hizo a través de Android Studio, y se comunica previamente a un modulo Bluetooth® para lo cual es necesario el protocolo de comunicación RS-232.

En la monografía se encontrará la documentación previamente consultada, contextualizando al lector para lo que encontrará en los capítulos siguientes. En el desarrollo se podrá ver el porque se elige cada componente, su planteamiento y validación del sistema.

SUMMARY

This paper shows the development process of a home automation simulator made for student environments. The paper goes from the inception to the execution of the project. This simulator will have 3 fundamental components: the software that executes the simulator, the board that reads the sensor form the physical world, and a remote controller programmed for android.

A simulator is a reality created with characteristics from any phenomenon, in this case the simulator is created to emulate visually a house and being able to use the real world variables, also simulating some of them.

The objective in this project is create a low-cost simulator capable of run real-time variables and with the possibility that anyone can interact in a simple way, as in a video game. Also is wanted to add current elements of use in home automation as a smart phone.

The simulator was made in Unity3D, the interface with the sensors was made with an ARDUINO® electronic board, the remote controller was made with Android Studion, and it communicates before with a Bluetooth® module, for which is need a RS-232 Interface.

In the monograph will be find previously consulted documentation, contextualizing the reader for what will be found in the next chapters. In the development will be found the reason why was selected each component, its planning and system validation.

CONTENIDO

pag.

INTRODUCCIÓN 7
1. ASPECTOS GENERALES 9
1.1. ANTECEDENTES 9
1.1.1 Showrooms iniciales online para comercialización de domótica 9
1.1.2 Eventos online y el TCP/IP. 10
1.1.3 Simulación del comportamiento de una casa. 10
1.1.4 Sistema de visualización de aplicación de planeamiento de escenarios y contexto. 10
1.1.5 HOME I/O®. 11
1.2 PLANTEAMIENTO DEL PROBLEMA 12
1.3 JUSTIFICACIÓN 14
1.4 OBJETIVOS 14
1.4.1 Objetivo general 14
1.4.2 Objetivos específicos 14
1.6. ALCANCES Y LIMITACIONES 15
1.7. LÍNEA DE INVESTIGACIÓN 16
1.8 DISEÑO METODOLÓGICO 16
2. MARCO REFERENCIAL 19
12. MARCO REFERENCIAL 19
2.1 MARCO TEÓRICO 19
2.2 Marco técnico 22
2.2.1 Bluetooth® 22
2.2.2 Comunicación serial 22
2.2.3 Sensorica 23
2.3 MARCO LEGAL 24
2.3.1 Marco legal Nacional 24
2.3.2 Marco legal internacional 24
3. INGENIERÍA DEL PROYECTO 27
3.1 ARQUITECTURA PROPUESTA PARA EL SIMULADOR 27
3.2 DISEÑO DEL AMBIENTE VIRTUAL 28
3.2.1 Modelo 3D 28
3.2.1.1 3DSMax® 28
3.2.1.2 Modelado en 3DSMax 29
3.2.1.3 Exportación a .fbx 30
3.2.1.4 Motor de Videojuegos 32
3.2.3 Unity3D 34
3.2.3.1 Creación de escenas en Unity3D 34

3.2.3.2 Interfaz de Unity3D 34
3.2.3.3 Ambientación en Unity3D 36
3.2.3.3.1 Creación del terreno 37
3.2.3.3.2 Cambio del cielo 38
3.2.3.3.3 Usar la luz solar o ambiente 38
3.2.3.3.4 Integrar el personaje 38
3.2.3.3.5 Importación del modelo .fbx 39
3.2.3.4 Variables dentro de la casa 40
3.2.3.4.1 Luces 40
3.2.3.4.2 Animaciones 42
3.3 SENSORICA 43
3.3.1Selección de sensores 43
3.3.2 Calibración de los sensores 44
3.3.2.1 PIR-SE-10 44
3.3.2.2 HIH-4030 45
3.3.2.3 TEMT6000 46
3.3.2.4 MQ-2 48
3.3.2.4 LM-35 49
3.3.3 Integración con la tarjeta de desarrollo 49
3.3.3.1 Implementación con ARDUINO UNO 50
3.3.4 Banco domótico 50
3.3.4.1 Conexiones 52
3.4 CONTROL REMOTO 53
3.4.1 Capa física 53
3.4.2 Conexión de módulos 53
3.4.2.1 Módulo RS-232 54
3.4.2.2 Módulo Bluetooth 54
3.4.2 Desarrollo de la aplicación en Android 56
3.4.2.1 Creación de la interfaz 58
3.4.2.2 Programación de las Activities 59
3.5 INTEGRACIÓN CON EL SIMULADOR 63
3.5.1 Integración de Unity3D con el control remoto 63
3.5.2 Integración de Unity3D con la tarjeta de desarrollo 65
4. RESULTADOS, CONCLUSIONES E IMPACTOS 67
4.1 Resultados 67
4.2.1 Práctica de sensores 67
4.2.1.1 Sensor de presencia 67
4.2.1.2 Sensor de gas 68
4.2.1.1 Sensor de temperatura 68
4.2.1.1 Sensor de humedad 69
4.2.1.1 Sensor de intensidad lumínica 69

4.1.2 Control Remoto 70
4.1.2.1 Puertas 70
4.1.2.1 Luces 72
4.3 Conclusiones 76
4.4 Trabajos futuros 76
BIBLIOGRAFÍA 78
ANEXOS 82

LISTA DE FIGURAS

pag.

Figura 1:Visualización del simulador hecho por Ju-Ming Su et.al. 11
Figura 2: Imagen oficial de HOME I/O® la cual muestra la posibilidad de conexión fuera del simulador. 12
Figura 3: Panel domótico de la universidad de León. 13
Figura 4: Diseño metodológico del presente trabajo 17
Figura 5: Arquitectura propuesta para el simulador. 27
Figura 6: Plano usado en el simulador 29
Figura 7: Modelo ya extruido en 3DSMax. 30
Figura 8: Propiedades de exportación de Include para .fbx 31
Figura 9: Propiedades de exportación de Advanced Options para .fbx 32
Figura 10: Controles superiores de Unity3D 35
Figura 11: Interfaz de usuario en Unity3D 36
Figura 12: Propiedades del terreno 37
Figura 13: Propiedades gráficas para cambiar el cielo 38
38
Figura 14: Propiedades de los modelos importados a Unity3D 39
Figura 15: Escena global ambientada 40
Figura 16: Point Lighr en Unity3D 41
Figura 17: Spot Light en Unity3D 41
Figura18: distribución de las luces en Unity3D. 42
Figura 19: creación de una animación en Unity3D. 43
Figura 20: esquema de implementación de los sensores 44
Figura 21: Higrómetro RH101 45
Figura 22: Esquemático del circuito usado para calibración del sensor TEMT6000.46
Figura 23: Gráfico de dispersión para la identificación del sensor TEMT6000 47
Figura 24: Relación de resistencias para sensor MQ-2 48
Figura 25: Panel domótico listo para la práctica de sensores. 51
Figura 26: Plano esquemático de conexión de los sensores al ARDUINO UNO. 52
Figura 27: Conexión desde TeraTerm al módulo Bluetooth 55
Figura 28: Configuración del módulo RN-41 55
Figura 29: Diseño PCB para el conector Bluetooth 56
Figura 30: Interfaz de usuario de Android Studio. 59
Figura 31: Diagrama de flujo para la programación de Activities 60
Figura 32: Diagrama de flujo del método onCreate() 62

Figura 33: Diseño de la interfaz de la actividad secundaria. 63
Figura 34: diagrama de flujo del método Update() para la comunicación Bluetooth.64
Figura 35: Script compilado y añadido al Character Controller 65
Figura 36: diagrama de flujo del método Upadte para la comunicación con la tarjeta de desarrollo 66
Figura 37: Estado inicial para los resultados 67
Figura 38: Demostración del sensor de presencia 68
Figura 39: Demostración del sensor de gas. 68
Figura 40: Demostración del sensor de temperatura 69
Figura 41: Demostración del sensor de humedad 69
Figura 42: Primer demostración del sensor de Intensidad Lumínica 70
Figura 43: Segunda demostración del sensor de Intensidad Lumínica 70
Figura 44: Botón seleccionado para abrir la primer puerta 71
Figura 45: Prueba de control remoto con la primer puerta 71
Figura 46: Prueba del control remoto con ambas puertas 72
Figura 47: Botón seleccionado para prender la primer luz 73
Figura 48: Prueba de control remoto con la primer luz 73
Figura 49: Prueba de control remoto con la segunda lu 74
z 74
74
Figura 50: Prueba de control remoto con la tercer luz 74
Figura 51: Prueba de control remoto con la cuarta luz 75
Figura 52: Prueba de control remoto con todos los botones accionados 75

INTRODUCCIÓN

En la actualidad se puede ver la influencia de las tecnologías domóticas en el cotidiano de la gente alrededor del mundo, es por eso que el acercamiento a los usuarios, o futuros desarrolladores o incluso instaladores, es de vital importancia.

Se puede observar que los bancos domóticos o incluso industriales, solo constan de estaciones estáticas en las cuales se pueden ir implementando sensores y actuadores uno a uno, sin aplicarlo a ningún tipo de ambiente o proceso real. En otras palabras, si se desea usar una estación domótica para el aprendizaje, nos vemos cortos a la hora de poder visualizar exactamente como trabajan estos en campo.

Es por eso que el proyecto mostrado en esta monografía esta en mira de crear un ambiente, donde se puedan visualizar, y enseñar las funciones de sistemas domóticos en un ambiente virtual, el cual a su vez pueda ser conectado a sensores del mundo real. Lo que se busca es crear una herramienta que permita hacer entender de una forma más fácil los alcances en domótica, y a su vez dar pautas iniciales para seguir explorando el campo de simulaciones y domótica.

Como objetivo del proyecto se quiso crear un ambiente en el que se pudiera recrear un control usado hoy en día en cualquier sistema domótico. Para esto se elige crear una aplicación en Android que maneje las variables en nuestro sistema.

1. ASPECTOS GENERALES1.1. ANTECEDENTES

En la actualidad, el paradigma de las simulaciones está siendo implementado en innumerables campos, ya sean o no tecnológicos. Dicho esto, la domótica no queda exenta de aplicaciones simuladas, pero tampoco es notorio el interés en el desarrollo de simulaciones entorno a este tema.

El avance que ha presentado está ligado generalmente a lo comercial, esto hace que la documentación disponible por la academia sobre simulaciones necesariamente aplicadas a la domótica sea estrictamente limitada. No quiere decir, que la academia no presente ningún tipo de trabajo al respecto, solo que el campo está poco documentado.

Por otra parte al usar como ambiente principal de desarrollo de la herramienta un motor de videojuegos, se puede involucrar la naturaleza de estas aplicaciones para una previa vista de lo que se ha realizado y este relacionado directamente con las simulaciones.

A continuación se presentaran los casos más notorios, en los cuales se notarán una evolución de lo que se ha hecho en el campo, y como la academia se ve relegada a estos temas.**1.1.1 Showrooms iniciales online para comercialización de domótica**

Cabe destacar que a lo largo de los años los principales actores interesados en realizar este tipo de entornos virtuales fueron compañías productoras de hardware en domótica, y se realizaban más como un accesorio para atraer clientes en conferencias, o en el sitio web de la compañía que comercializaba el producto. Como se puede apreciar en The little guys [1] y Domótica Viva S.L. [2], eran producciones muy básicas, donde al usuario se le limitaba mucho el espacio en el que podía interaccionar. Se puede ver claramente que solo son imágenes y animaciones en 2D que dan pocas señales de realismo, obviamente son productos que ya datan de varios años, más sin embargo son aún herramientas que se usan para vender. En estas simulaciones se pueden ver las características de los sistemas, y generalmente se da algún texto de apoyo al usuario, pero al ser web no tienen la posibilidad de demostrar alguna variable real.

1.1.2 Eventos online y el TCP/IP.

El trabajo de Javier Gonzáles Serran et. al. [3] a pesar de no mostrar fecha de publicación se puede observar que es de años de la década pasada, teniendo esto en cuenta es aproximado a los años donde las aplicaciones web fueron iniciando, esto hace que en su tiempo fuese muy innovador. Este proyecto tomaba muestras reales de un sistema domótico y las representaba en un entorno web. Algo que la industria domótica usa mucho en la actualidad. Cabe destacar que fue uno de los primeros intentos de usar el protocolo TCP/IP con sistemas de automatización domésticos, pero se puede categorizar dentro de simulador por las características de este producto, en otras palabras se usó para ver las variables del mundo real en un entorno que lo emulara. Otro punto a destacar de este proyecto es la especificación que realiza en cuanto a todos los protocolos de instalación y de comunicación entre sistemas domóticos, se ve la profundización que se hace para poder estandarizar el producto final.**1.1.3 Simulación del comportamiento de una casa.**

Por otro lado en revistas indexadas se encuentra muy poca documentación con algo tan puntual. Como indica G.Conte et. al [4], si hay simulaciones de domótica en ingeniería, en este caso se buscaba generar una serie de datos que fuesen lo suficientemente parecidos a una casa real. Allí se modeló el consumo eléctrico de cada electrodoméstico, y dentro de una serie de características se creó el comportamiento genérico de cada uno, de allí se obtuvieron resultados dirigidos hacía el consumo eléctrico en estos sistemas cuando están controlados o no por alguna variable, dando así una vista más enfocada al análisis de datos del gasto energético visto desde un simulador. Es un simulador ya que esta tomando el comportamiento del sistema en un entorno no real, así no se creen interfaces de movimiento, si se está creando un ambiente de simulación de datos.**1.1.4 Sistema de visualización de aplicación de planeamiento de escenarios y contexto.**

Durante la revisión literaria uno de los proyectos más cercanos a lo que se quiere lograr fuer realizado en Taiwan por Jun-Ming Su et. al. [5], donde se enfocaron en realizar un acercamiento a las simulaciones para automatización en casa para “Visualization System of Context-aware Application Scenario Planning (VS-CaSP)”, “Sistema de visualización de aplicación de planeamiento de escenarios y contexto” donde la finalidad era realizar un software que ayudase tanto a las personas con bajo conocimiento técnico a familiarizarse con estos sistemas, tanto a los diseñadores de este tipo de aplicaciones. Lo que se hizo en este proyecto fue usar sensores domóticos con ZigBee e integrarlos a una red capaz de soportarlos en un entorno 3D simple como se puede observar en la figura 1.

Cabe destacar de este proyecto que la arquitectura fue enfocada para todos los sensores de una casa, dando todo tipos de escenarios a la hora de simular y observar comportamientos. Además dan enfoque a la visualización 3D, y hacer diagramas por situaciones, queriendo tener respuestas especificas en el simulador dependiendo los sensores leídos.Figura 1:Visualización del simulador hecho por Ju-Ming Su et.al.

Fuente: An easy-to-use 3D visualization system for planning context-aware applications in smart buildings, Computer Standards & Interfaces [5].

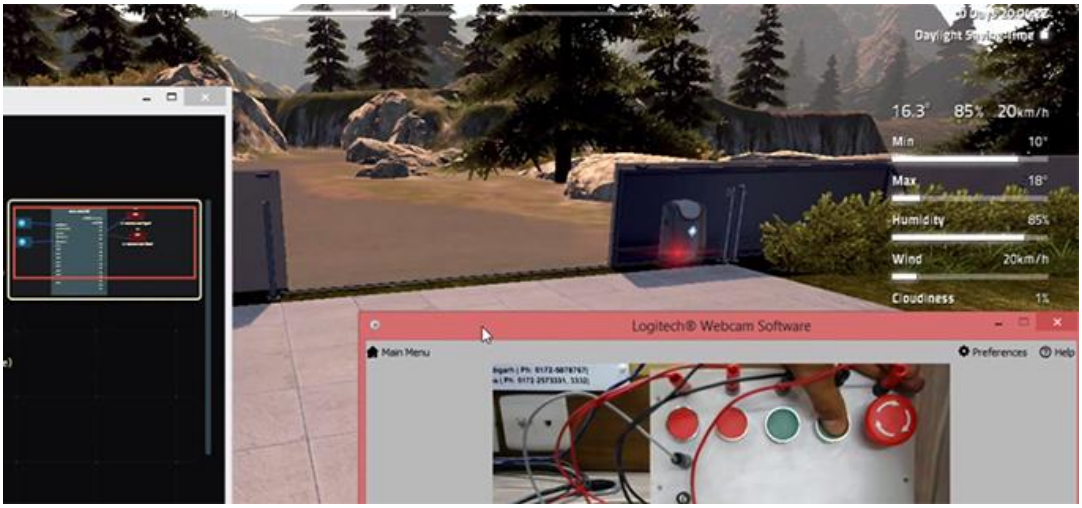
1.1.5 HOME I/O®.

Actualmente, solo se puede acudir a una empresa consolidada con simuladores en domótica, se trata de trata de la empresa RealGames ® a través de su producto Home I/O® [6]. Esta empresa ya va muy avanzada en este tipo de sistemas, el software puede conectarse a un PLC sin ningún problema para poder programar cualquier acción planeada para una casa. Tiene librerías diseñadas para visualizar rápidamente variables que toman largos tiempos de analizar. Por ejemplo la luz se puede simular la artificial junto la luz solar en cuestión de minutos, y ver como estas interaccionan en el ambiente. Otro ejemplo destacable

es el de la temperatura, se puede ver en la interfaz un mapa de calor en una parte inferior para poder visualizar cada rincón de la casa.

En general este sistema es el más avanzado en el tema, aparte se puede decir que es el único que reúne todas las características que se buscan puntualmente en el ambiente virtual que se desea desarrollar.

Como se puede observar en la figura 2 las capacidades de este software son ilimitadas, ya que permite conectividad en tiempo real con dispositivos como switches y botones que generalmente son usados para manejar PLCs, y dando de está forma todo un entorno en el cual experimentar la domótica, afectando tanto esas variables naturales, como esas que propiamente queremos programar. Básicamente es tener una ventana hacía una realidad que estamos creando.Figura 2: Imagen oficial de HOME I/O® la cual muestra la posibilidad de conexión fuera del simulador.



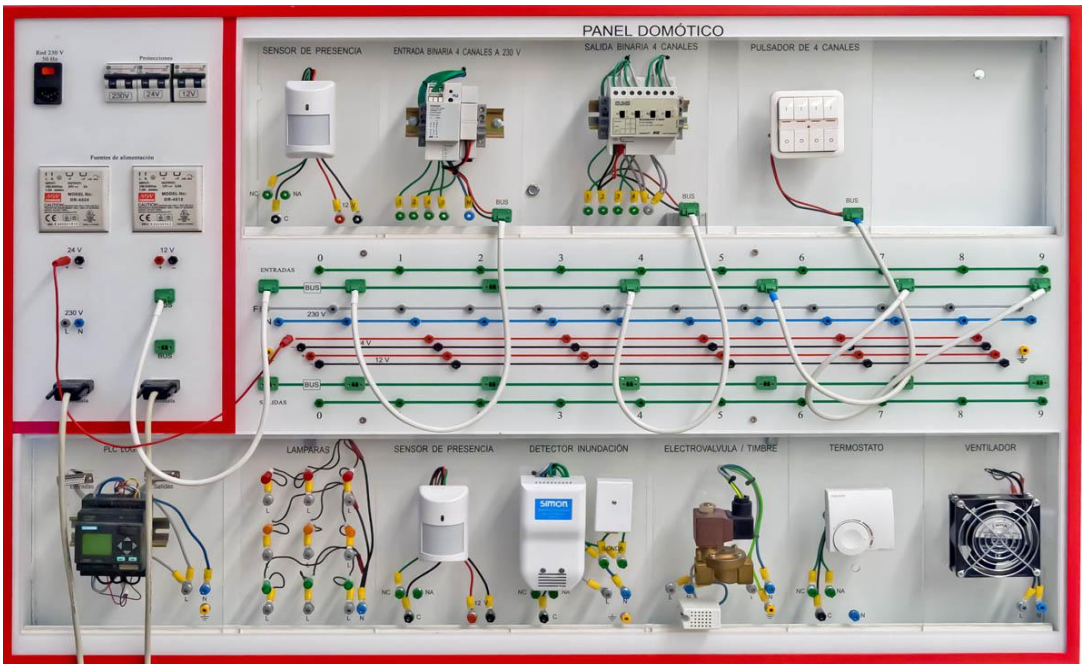
Fuente: HOME I/O® [6].

1.2 PLANTEAMIENTO DEL PROBLEMA

La domótica es un campo de gran incidencia actualmente en la arquitectura y la mecatrónica, es por eso que tener personal calificado es un tema muy importante para países como Colombia en el cual se está masificando el uso de estas tecnologías.

Es por eso que en las universidades, institutos o empresas que tienen que ver con el tema generalmente adquieren bancos como el mostrado en la figura 3 para la

enseñanza de estas tecnologías. Como se puede ver tienen sensores, actuadores, PLC, conexión al programador y si es necesario drivers para conectar tanto los sensores como los actuadores al PLC. Figura 3: Panel domótico de la universidad de León.



Fuente :http://lra.unileon.es/es/content/sistemasfísicos/panel_domótico.

El problema con estos bancos domóticos radica en que los laboratorios nacionales en domótica no aprovechan las tecnologías en visualización que ofrece el mundo de hoy, y a pesar de que son en algunos casos muy accesibles, Colombia no dispone con laboratorios de este tipo.

Es por esas razones que el proyecto que se plantea busca crear un ambiente virtual donde se puedan implementar conceptos de domótica, enfocado a la visualización, es decir, se puede ver al instante en una simulación 3D cuales son las ordenes que le damos al sistema. Inicialmente se hará una aproximación al uso de objetos domóticos, integrando unicamente sensores.

De acuerdo a lo mencionado se puede plantear la interrogante ¿Qué tan factible es desarrollar simuladores de domótica con objetos de bajo costo como herramienta de entrenamiento?**1.3 JUSTIFICACIÓN**

Colombia aún está entrando al mundo de los videojuegos, es por eso que el uso de estas tecnologías no es tan amplio a cómo deberían. Con ánimos de mostrar lo que estás aplicaciones pueden entregar se buscó crear un producto que supliera con necesidades de los diseñadores e instaladores de sistemas domóticos.

Un proyecto que puede ser realizado sin grandes presupuestos por la disposición de la comunidad desarrolladora de software, hacen de esta propuesta atractiva a investigarse, y a generar gente que pueda dominar las herramientas. El campo que puede abrir un proyecto de este tipo a crear más aplicaciones que tengan que ver o no con el objetivo, son inimaginables. Como un paso inicial a simulación estará bajo los parámetros que se delimiten, pero podrán generar conocimiento donde las simulaciones se puedan crear bajo lineamientos de otras líneas de aplicación.

El proyecto busca contextualizar y crear un ambiente de aprendizaje para la enseñanza de sistemas domóticos. En donde se busca poner a disposición una serie de herramientas reales y virtuales que garanticen la interacción de los sistemas domóticos aplicados.**1.4 OBJETIVOS1.4.1 Objetivo general** Crear un entorno de aprendizaje de domótica, el cuál permita la interacción de variables reales con un mundo virtual simulado.**1.4.2 Objetivos específicos**:

a) Diseñar la arquitectura del simulador.

b) Analizar y escoger los sensores que ofrece el mercado aplicables al simulador.

c) Realizar el modelo 3D del entorno virtual con las variables del simulador en el software Unity3D

d) Diseñar e implementar el protocolo entre el entorno virtual y el simulador.

e) Integrar los sensores del mundo real al mundo virtual.

f?? Realizar las pruebas finales de integración.**1.6. ALCANCES Y LIMITACIONES**

Para empezar el proyecto no será escalable, es decir no se hará de forma en la que se puedan instalar otros dispositivos sin tener que modificar cada parte del proyecto. Está simulación servirá solo con una única casa, al usar otra se tendría

que realizar otro diseño para el simulador, esto se debe a que por cada escena hay que adecuar todo el espacio de nuevo en cualquier motor de videojuegos, desde el terreno hasta cada variable domótica que se integre.

Este proyecto da para realizar muchos más a futuro, pero de momento se puntualizará para no recaer en problemas de cumplimiento o de fallos por negligencia en la programación. Por eso se hará una simulación de una sola casa, con un entorno limitado a la movilidad del personaje dentro de esta. Los comandos que se den se tienen que reproducir en tiempo real y con los parámetros que se indiquen por fabricación. Los valores tomados del mundo físico si se pueden visualizar en la simulación, y a su vez se pueden crear alertas a partir de sus valores. La principal tarea de la tarjeta será comunicar los sensores con el computador que este corriendo la simulación, de esta forma los sensores deberán tener en lo posible una señal adecuada de 0-5Voltios para que la tarjeta pueda leerlo, como análogo o digital.

Como se mencionó anteriormente se llegará a una simulación que muestre variables del mundo físico, sin usar control de variables análogas. El uso de variables análogas aún no se puede realizar debido a que un videojuego trabaja a una frecuencia que no se controla directamente en el código, esta frecuencia (FPS frames per second) depende de muchas variables que maneja el computador en ejecución. De allí se puede sugerir que se llegase a realizar un control de una variable análoga sería de forma en la que la simulación solo lea los valores del sensor, sin intervenir en el controlador, es decir que se haga el controlador externamente. Pero al realizar estos controladores se estaría saliendo de los objetivos propuestos, que a fin de cuentas da tema para otro proyecto.

También se hará una interfaz de usuario en Android, para configurar las variables de la simulación, tal cual como en un programa domótico real.

1.7. LÍNEA DE INVESTIGACIÓN

La línea de investigación en la que se enmarca el proyecto claramente es automatización y domótica, ya que directamente se quiere trabajar con sistemas automáticos en espacios habitables. Aunque una simulación puede entrar a trabajar en cualquier campo, no se pretende entrar a ninguna otra línea de investigación sugerida por la facultad de ingeniería mecatrónica de la Universidad Piloto de Colombia.

1.8 DISEÑO METODOLÓGICO

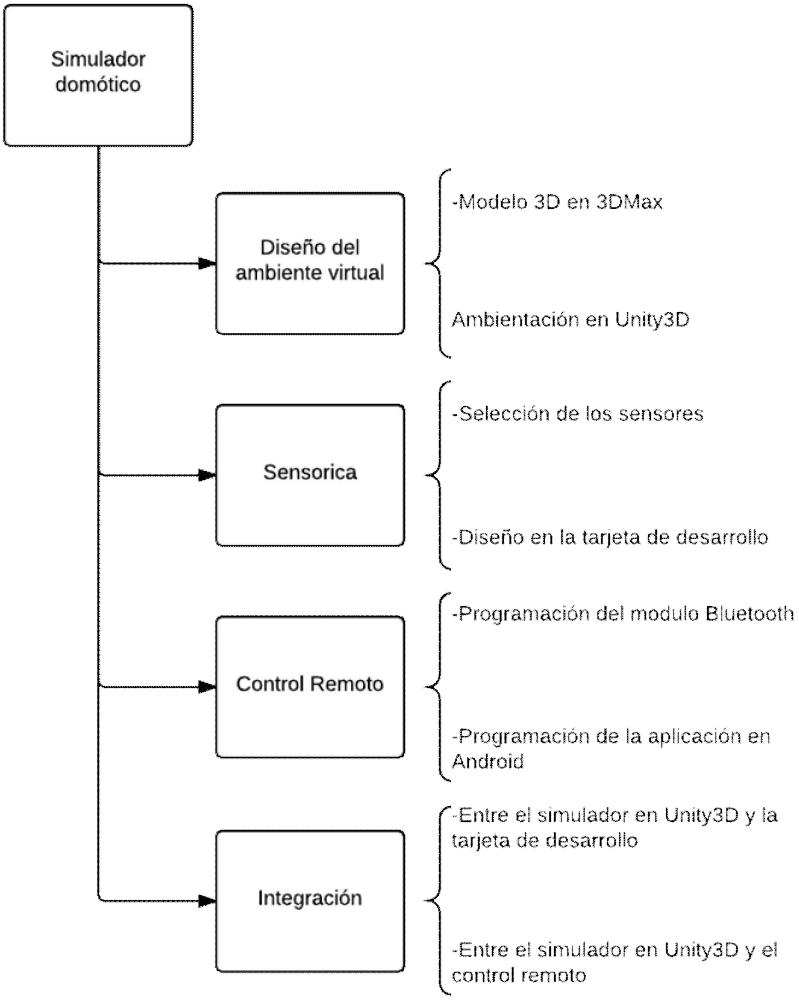
El proyecto se analizará desde la parte domótica en principio, dando como característica que sensores y sistemas domóticos caben dentro del proyecto,

además de dar una mejor idea de que cosas se pueden entorpecer entre ellas. El siguiente proceso sería implementar en la simulación como tal y experimentar que tan parecidas o análogas con la realidad son estas variables en la simulación. Finalmente se buscará una mínima aproximación al paradigma de experiencia de usuario viendo que tan amigable es con algunas personas la aplicación como tal.

A partir de lo que Ming Sun et. al. [7] postulan en cuanto metodología de la simulación, el proyecto se puede encajar dentro de *Application Methodology Directed by Target* ya que la simulación se basa en el objetivo, y busca en una forma general mediante el uso de diferentes perspectivas que llevan a dar un acercamiento más preciso de lo que sería la realidad desde el análisis de estas variables.

El siguiente diagrama muestra el proceso por el cual se llevará la metodología.

Figura 4: Diseño metodológico del presente trabajo



Fuente: propia

2. MARCO REFERENCIAL2.1 MARCO TEÓRICO

El presente proyecto consiste en simular un entorno domótico, por lo tanto se debe tener muy claro todo eso que se entiende por simulación. Para empezar el diccionario en línea de la RAE 1define simular como: ”Representar algo, fingiendo o imitando lo que no es.” Pero para una versión más acertada de lo que es simulación se puede usar la definición del sitio comercial Modelbenders LLC [8], donde la definición básica de simulación es: “Simulation is the process of designing a model of a real or imagined system and conducting experiments with that model”, “Simulación es el proceso de diseñar un modelo de un sistema real o imaginario al que se le hacen experimentos”. Dicho esto entendemos que simular es el hecho de crear un entorno basado en un fenómeno real o imaginario; para fines de acercamiento o de diseño se crean simulaciones imaginarias, donde el fenómeno no ha sido probado en ningún sitio, y mediante análisis se modela el entorno, y modelos reales es lo que más se usa, dando una aproximación al problema. Según Valverdúll [9] la simulación no puede tomar en cuenta todos los parámetros que involucran a un fenómeno real, no debe ser tenida como una herramienta secundaria ni mucho menos como un instrumento débil. Como tal la simulación se concibió desde la matemática, y de allí es de donde se hace la distinción de simulación discreta y continua, tal cual como una variable puede serlo, y en caso donde una simulación contenga varios tipos de variables no se le llama híbrida o mixta, si no se pasa a usar el tipo de variable predominante durante el entorno.

Para un entendimiento más literario de lo que se hará en este proyecto, la simulación a realizar cabe dentro de un experimento no material, que a su vez es realizado por modelos computacionales, es decir *in virtuo.* Como tal, una simulación no se debe tomar como un elemento secundario de la realidad, más bien se debe tomar como una realidad en si: así como vemos nuestros pensamientos dentro de la realidad organizado en conceptos e ideas, tal cual es una simulación, un entorno organizado por conceptos e ideas dando características a una realidad en sí. Obviamente hay ciertos parámetros que una simulación aún no puede contener, pero solo por el hecho de emular su naturaleza principal y llevarla a un entorno donde se piense que tal cual como un modelo real, lo lleva a un estado de realidad. En otras palabras, para hacer experimentos mediante simulaciones que sean tomados en cuenta para la elaboración de un producto, o inclusive de un modelo físico, hacen de ese entorno una realidad.

1 RAE: Real Academia Española

Ahora para entender que puede caber dentro de una simulación se darán ejemplos de simulaciones. El primero es simulación en construcción donde se toma maquinaria articulada, donde aplican principios de cinemática directa e inversa, además de un set de instrucciones básicas como “cargar cuchara”. De allí parten para la planeación de como será el proceso en especifico que se hará, es decir la simulación en este caso sirve para prever problemas en ejecución por espacio especialmente. El documento es parte del trabajo realizado por Vineet R. Kamat et. Al [10], ya que también han realizado un sistema de programación para este tipo de aplicaciones en simulación.

Además de propósitos de diseño, las simulaciones sirven para efectos de entrenamiento, ya sea de maquinaria, o de estrategias que puedan aplicar las personas en ciertas situaciones. Como el grupo que desarrollo una plataforma de simulación de combate en 3D, realizado por Fei Yao et. al. [11] donde buscan de una forma básica crear un entorno de entrenamiento militar, haciendo uso de la herramienta Unity3D, en este trabajo se puede ver claramente la diferencia de un videojuego tipo *shooter*[2] para hacerlo cercano a la industria y la academia.

Como también se ha visto, hay varias empresas que están desarrollando simuladores de automóviles para enseñarle a aquellas personas que no se atreven a hacerlo de primeras en un auto real. Es el caso del proyecto de Jian Jie et. al. [12] Donde, usando también la plataforma Unity3D logran crear un entorno simple para esta finalidad, allí se puede ver el gran acercamiento que se le puede dar al entrenamiento por medio de herramientas de simulación, que en la actualidad no son costosas de producir, además de poderse usar con múltiples tecnologías.

De esta forma estas simulaciones pueden ser realizadas en Unity3D, para mejor entendimiento de la plataforma se puede remitir a [13], allí se explican las bases para poder ver el funcionamiento de esta plataforma de videojuegos, allí se puede ver que tipo de programación y diseño de algoritmos puede manejar, y como se pueden generar IA[3] para el uso que se quiera dar. En general da una perspectiva para el programador de como trabajar en este ambiente, pero no da nociones de los otros componentes del sistema.

Además de la revisión en simulaciones se debe tener en cuenta la sensorica, ya que hará un papel importante en el proyecto. En domótica en general se llevan a

2 Shooter: juegos caracterizados por ser del tipo bélicos, donde el objetivo generalmente es derrotar al enemigo por medio de armas de fuego.

3 IA: Inteligencia Artificial

cabo desarrollos en análisis de información, y no tanto en capa de aplicación según Liyanage C. De Silva et. al. [14] donde las aplicaciones se están haciendo en reconocimiento de imagen y de sonido principalmente, la búsqueda en patrones que ayuden a identificar problemas, o también realice control de acceso. Por otra parte también se están llevando a cabo adaptaciones en el sistema general de lo que es cuidado de personas con discapacidades, o adultos mayores, para mejorar la calidad de vida mediante tecnología. Pero los dos campos en los cuales se está dando gran importancia son los de administración de la energía, y el uso multimedia dentro de la misma casa. Como se puede ver actualmente los estudios en domótica no se enfocan en realizar nuevos dispositivos, sino en como estamos usando esos dispositivos y que mejor uso se les puede dar, que nuevas aplicaciones se pueden crear desde las ciencias computacionales especialmente y como ayuda esto a mejorar la calidad de vida y a hacer mas sostenible una casa.

El campo de acción de la sensoria en automatización de espacios habitables es un tema de acción muy amplio, pero la documentación consultada condensa este tema de una manera ideal. Según Dan Ding et. al. [15] La sensorica en una casa se puede agrupar en dos grandes tipos: Censado directo en el entorno, o sistemas mediados por infraestructura. Dentro de el censado directo se pueden observar las categorías de: Sensores binarios, cámaras de video, Identificación por radio frecuencia (RFID), entre otros. Dentro de los sensores binarios se enmarcan todos ellos que tengan una señal discreta, el ejemplo más claro de esto en domótica son los sensores de presencia, hay o no hay alguien. Las cámaras de video son sensores a los que actualmente se les presta gran atención, y polémica, por sus prestaciones puede dar líos de intimidad y técnicamente el almacenamiento y entrega de la información puede ser problema. Los sistemas RFID actualmente tienen gran acogida, y están llegando ser usados como billeteras virtuales, por lo tanto su aplicación y desarrollo se enfoca principalmente en la encriptación de la información. Siguiendo la clasificación que llevabamos, restan los “otros sensores”, estos sensores son los que como ingenieros consideramos de naturaleza análoga o continua; dentro de este tipo caben los barómetros, termocuplas, higrómetros, entre otros.

Como tal la información necesaria para realizar este proyecto se resume en simulaciones, motores y domótica, ya que sin el conocimiento de ninguno de estos no se puede diseñar la arquitectura que manejará, además de incurrir quizá en problemas de caracterización o simple análisis.

2.2 Marco técnico

A continuación se mostrarán algunos conceptos que deben quedar claros para el entendimiento del proyecto.**2.2.1 Bluetooth**®

El sistema de comunicación Bluetooth® fue creado por el SIG[4] para satisfacer la necesidad de comunicación entre dispositivos. Inicialmente fue creado por un grupo de compañías importantes en la industria tecnológica, pero con el tiempo ha abierto sus puertas para que cualquier compañía pueda aportar en la mejora del Bluetooth®, o también usar el logo y las marcas Bluetooth® para el interés de este. Como tal el sistema de comunicación es inalambrico y creado para satisfacer ese vacio que existia entre muchas tecnologías a la hora de conectarse facilmente. Actualmente Bluetooth® es mayoritariamente usado en los llamados "dispositivos inteligentes" para poder realizar tareas especificas.

Actualmente la investigación de Bluetooth® más allá de llevarse a nivel de alta transferencia de datos, o multiple comunicación masiva, se está llevando al punto de dispositivos de bajo consumo, esto se ha dado con los años por el desarrollo en si que ha tenido el sistema. Inicialmente se pensaba como una alternativa al protocolo IEEE 802.11[5], pero con el uso y desarrollo ha tomado un camino totalmente distinto. **2.2.2 Comunicación serial**

Este tipo de comunicación es un protocolo ampliamente usado, para la comunicación de dispositivos. Aunque actualmente se esta dando a un desuso de este, la facilidad con la que se puede manejar con microcontroladores crea una ventaja frente a otros protocolos.

Hace no muchos años se podía encontrar un puerto RS-232 en cualquier computador, pero como se ha mencionado se está haciendo desuso en la industria de consumo (computadores personales, o dispositivos móviles). Pero aún hay muchas aplicaciones que se usan bajo comunicación serial, actualmente se pueden encontrar PLC[6] que se comunican a través del protocolo RS-485.

Los estándares más conocidos de comunicación serial son el RS-232 y el RS-485 que indican: el diseño de los conectores, la velocidad de transferencia, el tipo de cable, la distancia que debe tener el cable con respecto a la velocidad. En general lo que se indica son las configuraciones que se deben tomar para que la

---

4 Special Interest Group
5 Es el protocolo que define el estándar WIFI(Wireless Fidelity).
6 Del inglés "Programmable Logic Controller", controlador lógico programable.

comunicación no tenga problemas.

Como se mencionó la facilidad con la que se puede implementar para comunicar microcontroladores hace que la implementación de este sea atractiva a la hora de desarrollar. Además cuando se usa está configuración se puede hacer la comunicación con tal solo 3 cables(Rx, Tx, GND), claro está que sin ningún control de comunicación por hardware, ya que también se pueden implementar los 9 pines que tiene un conecto DB-5 para comunicación con RS-232. También cabe mencionar que el modulo Bluetooth® que fue utilizado, nos entrega la información bajo estos 3 hilos.**2.2.3 Sensorica**

Una parte importante de los sistemas de automatización son los sensores, los cuales están encargados de tomar los valores de variables físicas del mundo real y entregar esta información de forma en la que se pueda observar y/o procesar en un sistema.

En este proyecto se usarán los siguientes sensores:

- Sensor de presencia: el encargado de verificar el cambio de luz infrarroja en el ambiente, de esta forma determinando si ha entrado alguien a ese espacio.

-Termómetro: encargado de medir la temperatura en el ambiente, esta se puede leer en grados centigrados.

-Higrómetro: sensor usado para medir el grado de humedad relativa en el aire, por lo tanto el valor leido es un porcentaje.

-Sensor fotóelectrico: encargado de medir los cambios de luz en el ambiente, este mide en lux (lx).

-Sensor de gas: este esta encargado de medir las partículas por millón en el aire de otras sustancias, ya sea gas GLP, o humo.

Cabe destacar que existen líneas de estos sensores de magnitud industrial o domótica, e.g. no es igual la humedad de una lavandería a la de una casa, o la diferencia de temperatura que existe en un horno de acero a un edificio con calefacción. Esto será un punto de partida para elegir los sensores, ya que al tener un uso distinto, el costo también es distinto.

2.3 MARCO LEGAL

La practica de implementación de sistemas domóticos siempre será ligada por la regulación de terceros. Estos terceros son los productores de hardware que crean principalmente los equipos de comunicación, estas leyes siempre estarán reguladas si la comunicación es implementada inalambricamente. Si la conexión debe hacer alambrada se debe remitir a la norma eléctrica, que en Colombia es la norma NTC 2050.**2.3.1 Marco legal Nacional**

Actualmente en Colombia no existe ninguna ley que rija simuladores o domótica. Ni tampoco, ningún tipo de estándar NTC que pida normas sobre domótica explícitamente, es decir, que en Colombia se amparan la implementación de este tipo de sistemas sin mayor regulación por parte del estado.

El único aparte que se puede encontrar sobre las tecnologías usadas es la resolución 797 de 2001 del ministerio TIC, en el cual se asignaron las frecuencias en las que pueden trabajar tecnologías Bluetooth®, y el máximo permitido en potencia de estos. Donde se estipula que el límite de potencia o de intensidad de campo es de : 50mV/m a 3 metros máximo 100mW. **2.3.2 Marco legal internacional**

En España el estándar de domótica es la especificación de AENOR EA0026: 2006 Instalaciones de sistemas domóticos en viviendas. Prescripciones generales de instalación y evaluación[]. Mediante el link de la CEDOM(Asociación Española de Domótica e Inmótica) se puede clasificar una instalación domótica en tres niveles mediante un breve cuestionario, estos tres niveles se definen por:

Nivel 1. Son instalaciones con un nivel mínimo de dispositivos y/o aplicaciones domóticas. La suma de los pesos ponderados de los dispositivos incluidos en la instalación domótica debe ser como mínimo de 13, siempre que a su vez cubra al menos 3 aplicaciones domóticas. Es decir, estos 13 puntos deben conseguirse con dispositivos repartidos entre, al menos, 3 aplicaciones distintas que se distinguen por tener diferente color en la tabla. No conseguiría el nivel mínimo de domotización una instalación que alcanza una puntuación de 13 pero que sólo tiene instalados dispositivos de climatización y de control de persianas; necesitaría tener dispositivos instalados en una tercera aplicación como puede ser el videoportero.

Nivel 2. Son instalaciones con un nivel medio de dispositivos y/o aplicaciones domóticas. En este caso la suma de puntos debe ser de 30 como mínimo, siempre

que se cubran al menos 3 aplicaciones.

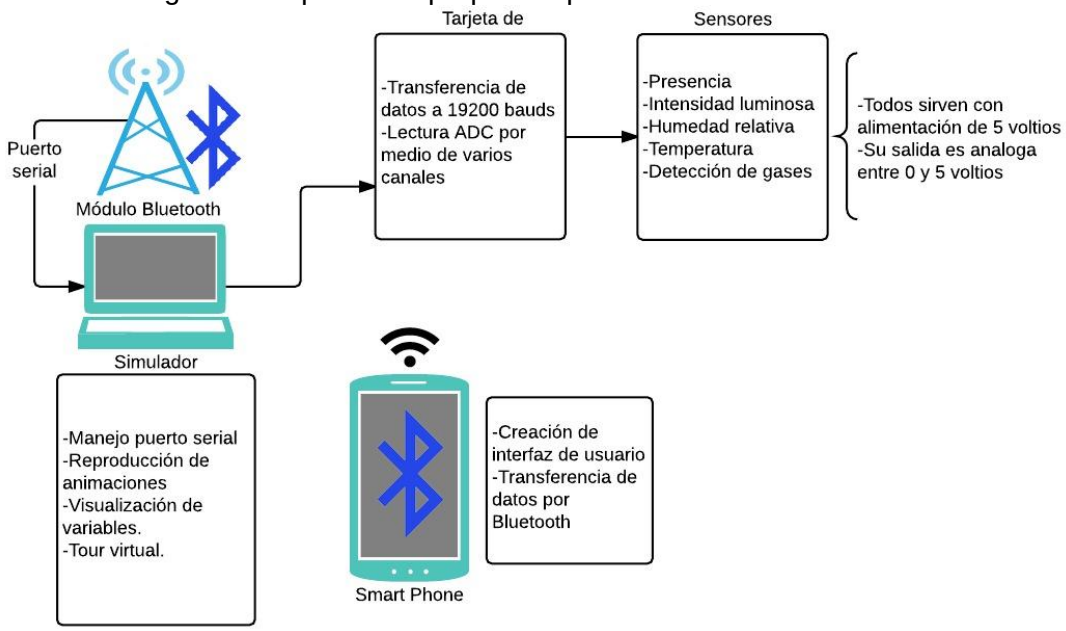
Nivel 3. Son instalaciones con un nivel alto de dispositivos y/o aplicaciones domóticas. En este caso la suma de puntos debe ser de 45 como mínimo, siempre que se repartan en al menos 6 aplicaciones.

Esta norma fue la base para crear la norma de referencia a nivel europeo: Requisitos generales para sistemas electrónicos para viviendas y edificios (HBES) y sistemas de automatización y control de edificios (BACS). Parte 6-1: Instalaciones HBES. Instalación y planificación. Esta norma también expone todas las bases para la clasificación de un sistema domótico, al igual que la norma a nivel español.

3. INGENIERÍA DEL PROYECTO

Este capítulo abarcará todo el desarrollo del sistema de entrenamiento, pasando por todos los puntos expuestos en la metodología, así mismo mostrándose en el mismo orden de aparición. **3.1 ARQUITECTURA PROPUESTA PARA EL SIMULADOR**

El simulador se compone de tres partes principales: el simulador, el control remoto y la interfaz electrónica. En el diagrama de la figura [] se muestra la función de cada uno de estos componentes, así como el tipo de conexión entre ellos. Existen dos tipos de conexión en este diagrama: cableada e inalámbrica. La conexión cableada se muestra como una flecha, y la conexión inalámbrica se representa con ondas.Figura 5: Arquitectura propuesta para el simulador.



Fuente; propia.

A continuación en este capítulo se mostrará el criterio de selección de cada dispositivo, seguidamente del proceso de desarrollo en cada uno de los programas y dispositivos involucrados, finalmente se mostrará la integración de estos mismos.**3.2 DISEÑO DEL AMBIENTE VIRTUAL**

El ambiente virtual se divide en dos partes: el modelado en 3DSMax® y la ambientación en Unity3D®, estos pasos son necesarios para la visualización de

las variables en el entorno, para ver la programación remítase al numeral 3.5.**3.2.1 Modelo 3D**

Para el modelado de la casa se crearon de la casa se valoraron los programas que se encuentran en la tabla 1**.**

Tabla1: Comparativa entre programas de modelado 3D

Programa	Inventor®	Blender®	Maya®	3DSMax®
Mercado objetivo	Diseño industrial, Diseño mecánico	Diseño industrial, animación	Diseño industrial, animación	Diseño industrial, animación
Exportación a .fbx	No	En algunas versiones	Si	Si

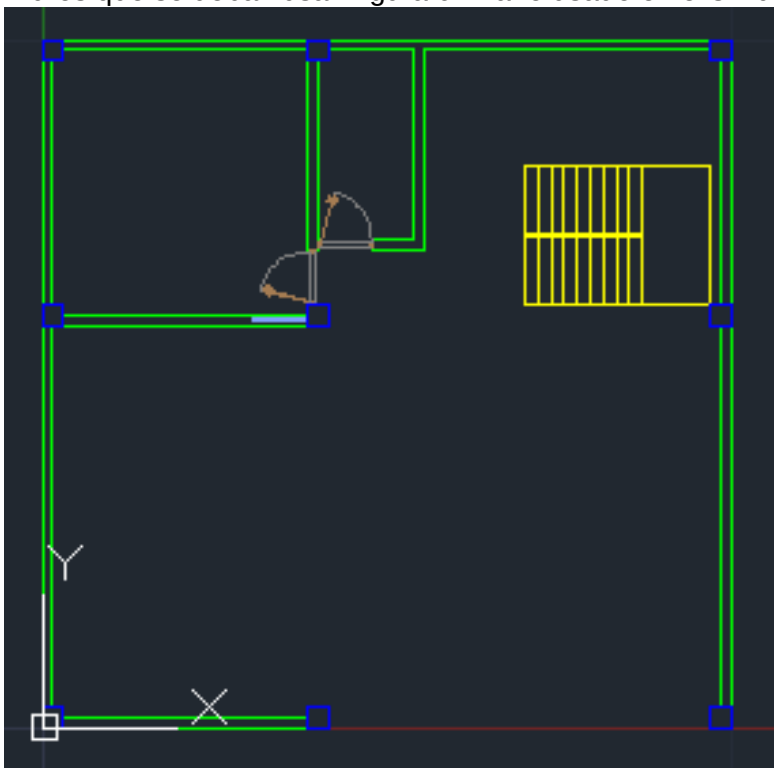
Fuente:Propia

A pesar de que Inventor es un programa con muchas facilidades de diseño, no es apto para este tipo de sistemas, porque la exportación en .fbx no está disponible, y generalmente los motores de videojuegos necesitan de este tipo de archivos para la importación de modelos 3D. Por parte de Blender a pesar de ser un programa de modelado 3D y de ser software libre, su exportación a .fbx no es siempre fiable. En cuanto a Maya tiene prestaciones en modelado muy buenas, y soporta el formato de exportación .fbx, al igual que 3DSMax, y ambos son de la empresa Autodesk, por lo tanto las prestaciones de estos dos programas son muy similares.**3.2.1.1 3DSMax®**

El programa elegido para el modelado fue 3D Studio Max ya que permite la importación de planos en Autocad, y así simplificar el proceso de levantamiento de muros. Además como se mencionó anteriormente permite una exportación a .fbx con opción de elegir varios opciones de formato, de exportación.**3.2.1.2 Modelado en 3DSMax**

Para empezar se debe importar el plano de Autocad a 3DSMax. En el caso de esta simulación la figura 6 muestra el plano de una sola planta que se usó para la ambientación, este plano fue proporcionado por la arquitecta Adriana Salazar. En lo posible para estos planos se deben usar polilineas cerradas, esto facilitará

levantar los muros que se deban usar.Figura 6: Plano usado en el simulador

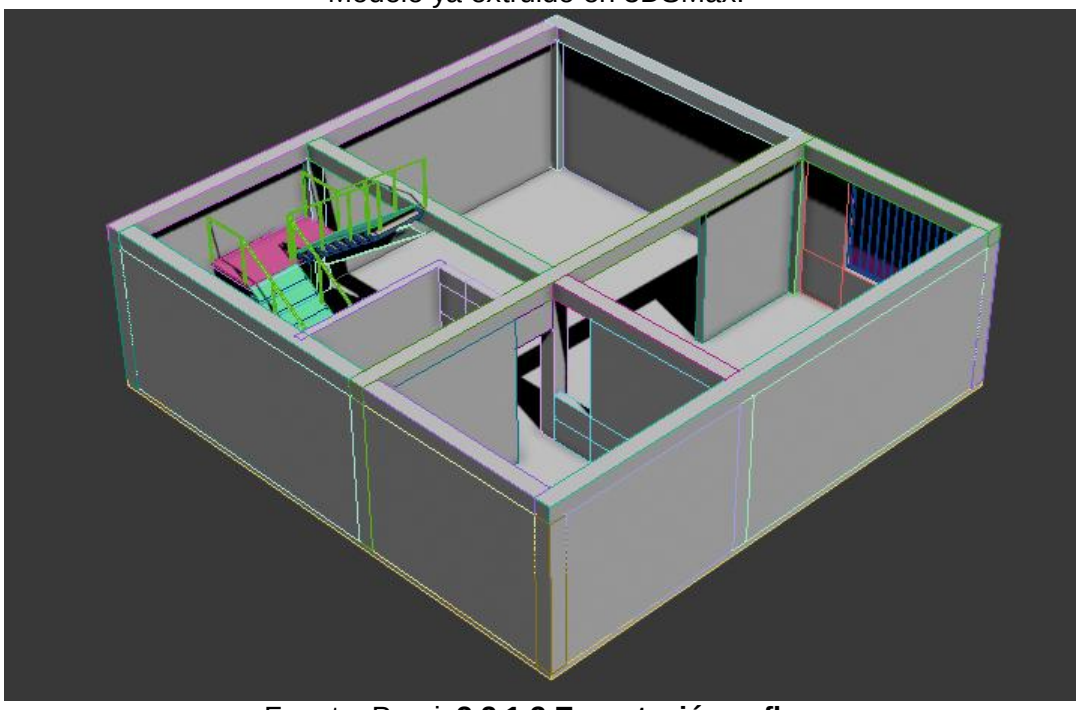


Fuente: Propia

Antes de empezar a trabajar en el modelo se deben configurar el sistema de unidades a usar. Para eso se debe ir a *Customize<Units Setup*... Allí se desplega un cuadro, en el cual se elige el sistema métrico y la escala. En este caso al ser una casa, por la dimensión de esta, se usa el sistema métrico en metros.

Ya una vez configurada la escala se procede a hacer el levantamiento de muros, en el caso de cada muro se debe extruir a la medida del techo, 2,5 metros en este caso. La figura 7 muestra el resultado de levantar muros y crear vigas en el proceso de modelado. Como se puede observar también se pueden ir integrando elementos estaticos propios del ambiente, en este caso son: una ventana y una escalera. También se observa que en la escalera existen tres planos en la escalera, dos verdes y uno rosado. Estos planos se conocen como colisiones, las cuales se crean para reducir la cantidad de elementos que tiene que procesar el motor de física. Es decir, se crean colisiones para cada muro desde su malla ya que son simples cubos, pero en el caso de la escalera se multiplica el número de

cubos, por lo tanto se crean planos que den un movimiento más suave del personaje, esta configuración se debe realizar en el motor de videojuegos, pero es preferible crear los planos y mallas en el programa de modelado 3D.Figura 7: Modelo ya extruido en 3DSMax.

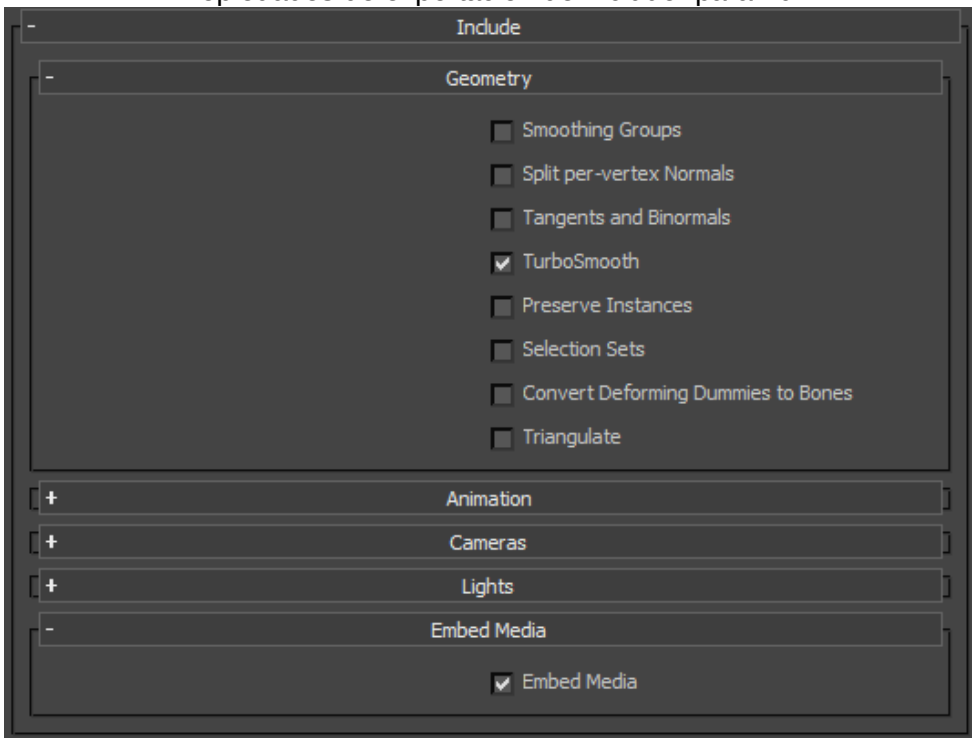


Fuente: Propia**3.2.1.3 Exportación a .fbx**

Una vez este terminado el modelo, se prosigue a crear el archivo de exportación. Para poder realizar esto se va al logo del programa<*Export*. Una vez allí sale aparece un cuadro de archivo, en este cuadro se debe elegir el nombre del archivo final, y la destinación de este, en tipo se elige “Autodesk(.fbx)” y se presiona guardar. Una vez realizado este proceso aparece en pantalla un cuadro de exportación con varias opciones para el formato. Elija una carpeta destino fácil de acceder, o en la carpeta destino de las librerías del motor de videojuegos.

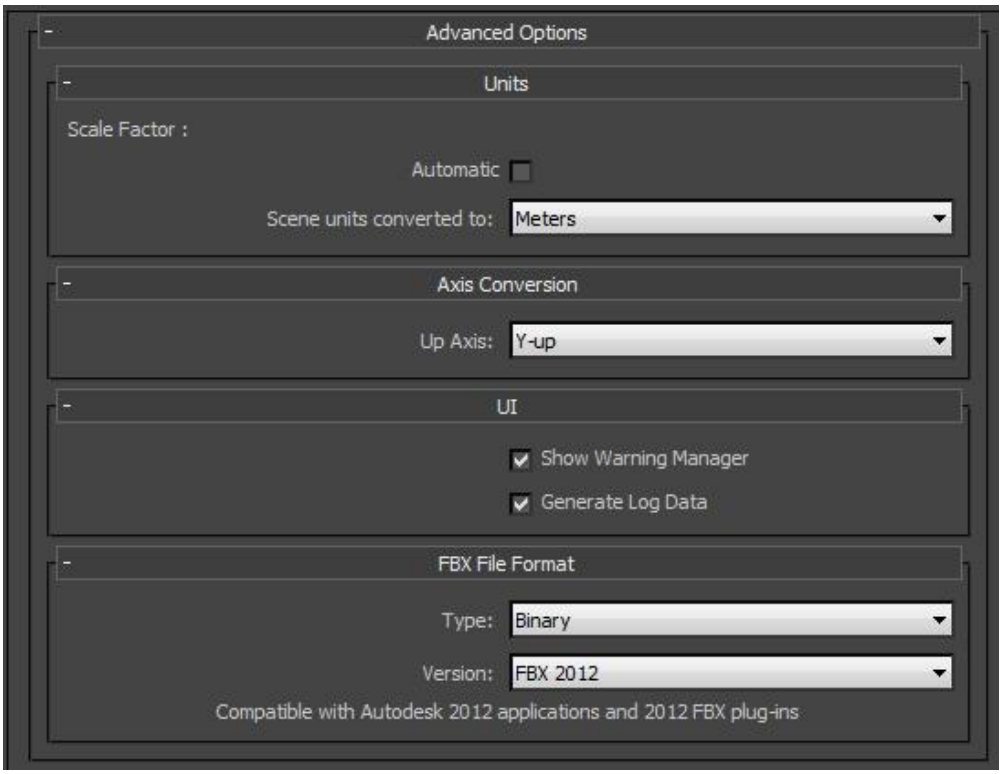
Para crear un archivo acorde a un motor de videojuegos se deben usar los parámetros que aparecen en las Figuras 8 y 9. En el caso de la caja *Include* se habilita en la sección *Geometry* solamente *TurboSmooth*, se desactivan las animaciones, cámaras, y luces, esto es debido a que se deben configurar más adelante en el motor de videojuegos, pero si se activa *Embed Media* ya que esto exporta materiales básicos que se hallan creado en 3DSmax.Figura 8:

Propiedades de exportación de *Include* para .fbx



Fuente: Propia

En el caso de la caja *Advanced Options* se configura de modo que el archivo quede lo más liviano posible. En la primera caja se elije el sistema métrico usado para la configuración inicial del archivo. En *Axis Convertion* se elije *Y-up* ya que para el motor de videojuegos que se eligió en este proyecto el eje Y es hacía arriba, por lo tanto antes de cambiar esta caja es preferible cerciorarse de este punto previamente. El cuadro *UI* no es relevante para lo que se usará este archivo, por lo tanto no importa si las cajas están seleccionadas o no. Por último en la caja *FBX File Format* se elije en el desplegable *Type* la opción *Binary*, y para la versión el que se prefiera, en este caso al ser 3DSMax 2012 la última versión disponible de exportación a .fbx es 2012.Figura 9: Propiedades de exportación de *Advanced Options* para .fbx



Fuente:Propia**3.2.1.4 Motor de Videojuegos**

Para la creación del ambiente simulado es necesario usar un motor de videojuegos. En la tabla 2 hay una comparativa de los motores de videojuego más conocidos, con los parámetros más importantes a la hora de elegir uno entre los propuestos. Para empezar UDK y CryEngine tienen como plataforma objetivo Windows, y para desarrollo en consolas de videojuegos con licencias especiales. Por el contrario Unity3D destaca por la gran cantidad de plataformas objetivo a la hora de desarrollar.

El siguiente punto evaluado es el lenguaje de programación, mientras que CryEngine y Unity 3D pueden trabajar con lenguajes de programación con librerías para acceder a los puertos, por el contrario UDK maneja un lenguaje de programación propio el cual no tiene acceso a estos puertos.

El formato de importación de modelos 3D permite el uso de archivos .fbx para los motores, es fue una de las razones por la cual elegir .fbx como formato de modelos en 3D.

El siguiente punto es uno de los más importantes, el costo de uso de una licencia de cada programa, UDK tiene licencia gratuita para uso no comercial, mientras CryEngine solicita un pago mensual de $9.99USD mensuales para el uso del programa, y Unity3D tampoco exige ningún tipo de pago para el uso no comercial.

Tabla 2: Comparativa de motores de videojuegos

Motor de videojuegos	UDK	CryEngine	Unity3D
Plataformas objetivo	Windows, iOS, Android, Oculus Rift. Xbox360*, XboxOne*, PS3*,PS4*.	Windows, Xbox360, XboxOne, PS3,PS4, WiiU, iOS, Android.	Windows, Linux, Mac,iOS, Android, BlackBerry, WindowsPhone, PS3, PS4,PSVita, PlayStationMobile, Xbox360,XboxONE, WiiU, Web.
Lenguaje de programación	UnrealScript	C++	Javascript, C#, BooScript.
Formatos 3D que soporta	.fbx	.fbx, .obj	.max, .mb,.ma,.fbx,.obj, .3ds, .dfx.
Costo	De uso libre pero para uso comercial de 99$USD**	9.99$USD al mes, o con acuerdo a Crytek	Version gratuita, o versión pro 75$USD al mes, o 1500%USD***

*Necesita una licencia especial que otorga Epic Games.
**luego de 50.000$USD en ganancias cobran 25% de la utilidad.
***Para iOS o Android se debe pagar otra versión pro.**3.2.3 Unity3D**

El motor de videojuegos elegido es Unity3D. Para empezar este software posee librerías que pueden hacer uso del puerto serial, además no tiene costo de uso para desarrollar un proyecto universitario. Además Unity3D ofrece posibilidades de expansión a otras plataforma con mayor facilidad.**3.2.3.1 Creación de escenas en Unity3D**

Lo primero para el uso de Unity3D es crear un proyecto, seguido de una escena.

Si en el computador no existe ningún proyecto creado, el programa enviará un cuadro para crear el proyecto. Elija un sitio de destino, y en las librerías elija:

- Character controller .unityPackage
- Skyboxes .unityPackage
- TerrainAssets .unityPackage

Finalmente elija en “Setup defaults for” 3D y presione Create. De esta forma el motor creará una escena con una cámara. **3.2.3.2 Interfaz de Unity3D**

Cada parte en la interfaz de Unity3D es de utilidad a la hora de desarrollar una escena,

En la figura 10 se observan enumeradas las partes del menú superior del programa.

En el número 1 se encuentran los modificadores, la opción en forma de mano sirve para mover la cámara en la escena, mientras los otros botones son modificadores de: posición, rotación y escala. Todos los modificadores poseen coordenadas en X,Y,Z.

En el número 2 se encuentran las configuraciones del pivote, este es el punto desde donde se cambiarán los modificadores, e.g. si se busca rotar un objeto, el pivote sera el punto donde inicie el eje de rotación.

En el número 3 se encuentran los botones de ejecución, al ser estos símbolos universales, se sobre entiende que en orden son: correr, pausar y ejecutar paso a paso.

En el número 4, están las capas, estas pueden configurarse por malla, de esta forma se pueden crear varias capas para ocultar cierto objetos que pueden interferir a la hora de desarrollar, e.g. si se desea ver el interior de una casa se puede crear una capa del techo y apagarla para que en la escena no sea molesta esa malla.

Por último se encuentra el desplegable *layout*, donde se puede cambiar la interfaz para que el desarrollador tenga un ambiente de uso más acorde al objetivo, ya sea animaciones, o ubicación de objetos en la escena.Figura 10: Controles superiores

de Unity3D

Fuente: propia.

El resto de la interfaz de usuario se puede ver en la figura 11**,** se pueden ver los distintos componentes para desarrollar en Unity3D.

Para empezar en la caja *Hierarchy* se encuentran todos los objetos que se han añadido a la escena. Desde el terreno, el modelo de la casa, el personaje, entre muchos otros elementos que hace un videojuego.

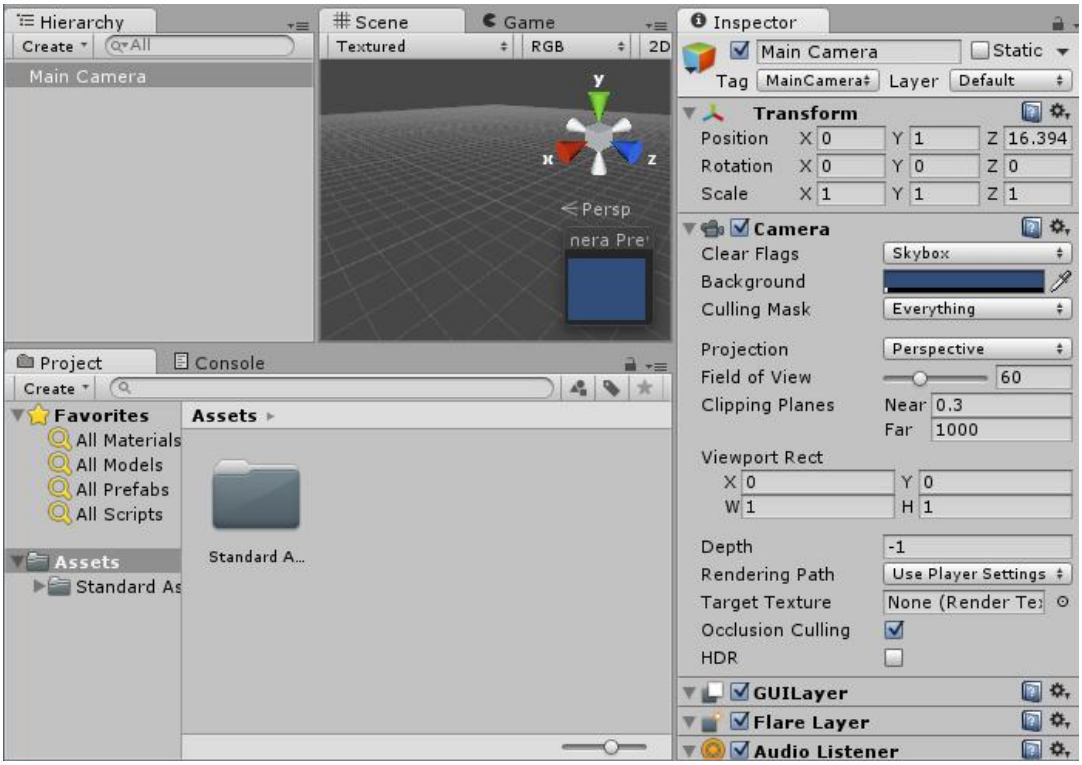
En la caja *Scene* se observa el resultado visual de los objetos que se han añadido a la escena. Dentro de la caja se usan los 3 botones del ratón para navegar.

En la caja *Game* se puede previsualizar la versión del simulador o videojuego sin tener que crear un archivo ejecutable. Esta se abre automáticamente al presionar el botón *Play* en el menú superior

En la caja *Inspector* se encuentran todas las características de un elemento seleccionado en *Hierarchy* o *Scene* en algunos casos también se puede visualizar configuraciones a las cuales se accede desde el menú principal.

En la caja *Project* se encuentran todos los archivos que se han creado para el proyecto, aquí se crean los modelos .fbx, los sonidos que se usen, las animaciones que se creen, los códigos que deba usar el motor y todos esos archivos que deba usar el proyecto. Se recomienda ser muy organizado con estos archivos, ya que a pesar de tener un buscador, es de mayor facilidad saber donde se encuentra el archivo que se deba usar.

Para finalizar se encuentra la caja *Console*, allí se mostrará todo el *Log* de errores, así mismo como los mensajes que se programen para mostrarse en ejecución, esta caja es primordialmente una ayuda al programador.Figura 11: Interfaz de usuario en Unity3D

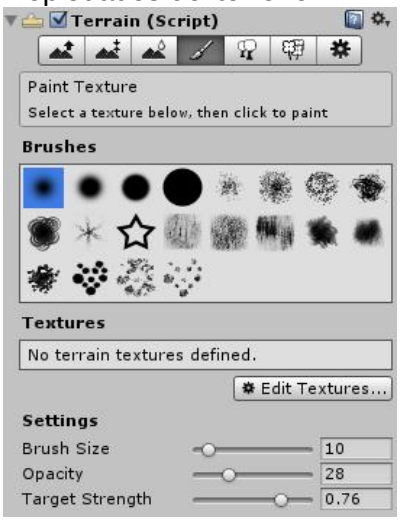


Fuente:propia.**3.2.3.3 Ambientación en Unity3D**

Ya al conocer la interfaz, el flujo de creación de varios objetos para ambientar el simulador es mejor. Antes de empezar a manejar cualquier cosa en el simulador es mejor eliminar el objeto por defecto creado en *Hierarchy*, este objeto tiene por nombre *Main camera*; para eliminarlo simplemente se da click derecho<*Delete*. Ya sin ningún objeto se procede a ambientar de una forma global, para este objetivo se crea el terreno, se cambia el cielo, se pone una luz de ambiente, se importa el modelo, y finalmente se integra al personaje. A continuación se describirá cada uno de estos procesos.**3.2.3.3.1 Creación del terreno**

Para la creación del terreno, se debe ir al menú *GameObject-Create Other-Terrain.* Ya una vez creado el terreno lo elegimos en *Scene* o en *Hierarchy.* Una vez elegido, en *Inspector* ya aparecen las propiedades de este objeto como se muestran en la figura 12. Allí se puede subir o bajar el terreno, inclusive incluir flores o pasto en el terreno, pero para lo que se hará basta con agregar una textura al piso. Para hacer eso, se elige el icono con la brocha como se muestra en la figura, se procede por dar click en *Edit Textures...-Add texture*, en el cuadro

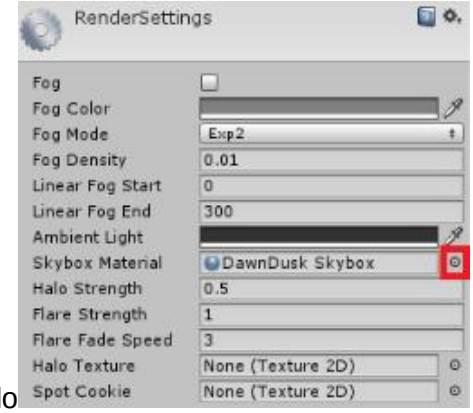
que aparece se da click en *select* de la primer caja de izquierda a derecha. En el nuevo cuadro que aparece se elije una textura. Para modificar el tamaño de esa imagen en el terreno se cambian X,Y por debajo de *Size.*Para visualizar simplemente se da click en *Add.* A este punto el terreno ya debe estar con una textura puesto.Figura 12: Propiedades del terreno



Fuente: Propia

3.2.3.3.2 Cambio del cielo

Para poder cambiar el cielo se accede al menú *Edit-Render Settings*. Al haber presionado este ítem el inspector se presentan las configuraciones que muestra la figura 13. Se debe presionar en el icono que está señalado en rojo para poder cambiar el cielo, allí aparecen de nuevo unos materiales los cuales son materiales en un domo, de esta forma se tiene la percepción de que el cielo rodea todo el espacio en el escenario.Figura 13: Propiedades gráficas para cambiar el



cielo

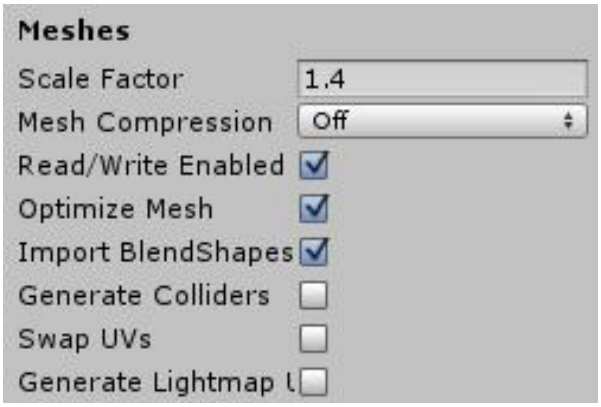
Fuente: Propia.**3.2.3.3.3 Usar la luz solar o ambiente**

Para insertar la luz ambiente se va a *GameObject-Create Other-Directional Light.* En el inspector se pueden cambiar propiedades de la luz como: angulo, color, intensidad, entre otras opciones más avanzadas. El color se puede variar de blanco para que de un color más acercado a la luz solar(naranaja), o a la luz de luna(blanco-azulado).**3.2.3.3.4 Integrar el personaje**

Para poder usar el simulador con las flechas del computador se debe integrar un objeto con varios códigos programados. Unity3D facilita este recurso que se encuentra como un archivo el cual se accede por *project,* este se encuentra en la carpeta *Assets/Standard Assets/Character Controllers.* Ya que para estos productos es mejor tener una cámara en primera persona se arrastra a *Scene* o a *Hierarchy* el objeto gris por nombre *First Person Controller*, muchas veces este objeto esta alejado del terreno, para acercarlo se puede hacer mediante el icono modificador, o por medio de *Inspector,*donde se cambia el valor de *Position.***3.2.3.3.5 Importación del modelo .fbx**

Para importar el modelo se debe agregar a la carpeta *Assets* mediante el explorador de carpetas, al hacer esto demora unos segundos y ya está listo para usar en la escena. Es preferible antes de usarlo elegir un valor de *Scale Factor* como se muestra en la figura 14, esto con el objeto de que no quede sobre dimensionado con respecto al *Character Controller*, además se puede verificar la caja de *Generate Colliders,* esto genera las colisiones de forma automática en el modelo 3D, pero para disminuir la carga de procesamiento se puede ir por cada objeto en la escena y añadir mediante *Inspector-Add Component-Physics* y se elige el tipo de colisión más cercana al modelo de ese objeto, e.g. a un muro se le puede agregar o una colisión de caja.Figura 14: Propiedades de los modelos

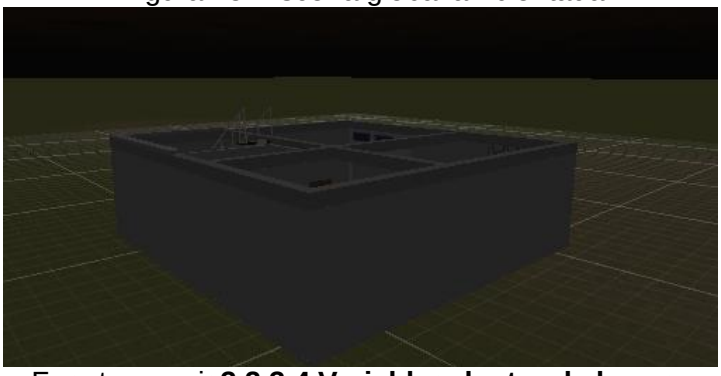
importados a Unity3D



Fuente: Propia

Ya con los pasos mencionados la escena debería estar cercana a la de la figura 15, con el cielo, el terreno, el modelo y el C*haracter Controller* dentro del modelo. No importa si el terreno o el cielo son distintos, eso depende del específico de cada escena.

Figura 15: Escena global ambientada

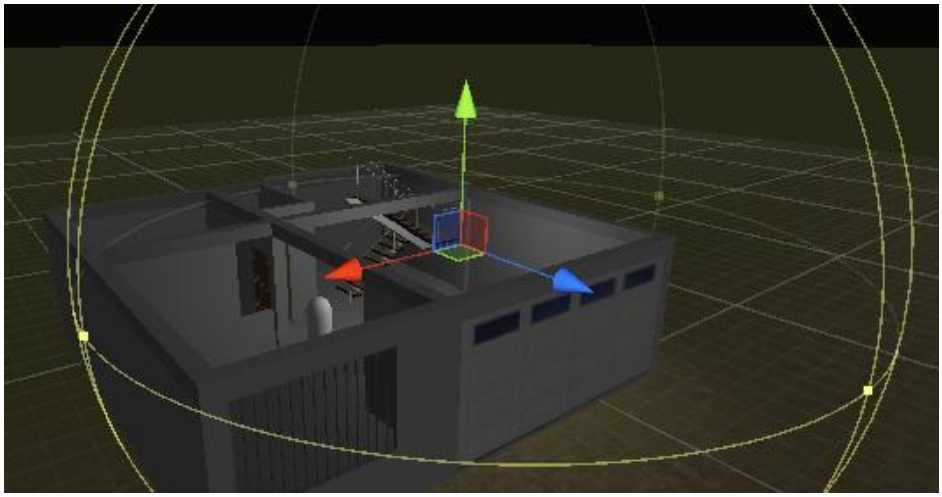


Fuente: propia**3.2.3.4 Variables dentro de la casa**

Estás variables cumplirán la función de responder al control remoto, en las cuales se tienen las luces interiores y unas puertas, las luces unicamente cumplen la función de prender y apagar, mientras las puertas abriran y cerraran.**3.2.3.4.1 Luces**

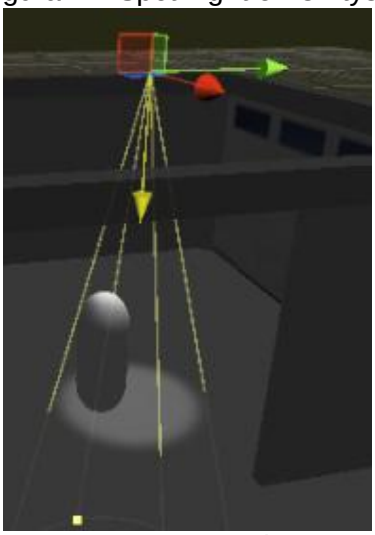
Para las luces se debe tener en cuenta que en Unity3D existen dos tipos de luces que asemejan el comportamiento de las luces de interiores, y ambas se agregan desde el menú *GameObject-Create Other*. La primera es la luz *Point light*, esta luz es la que aparece en la figura 16, donde se puede observar que es una luz con un área circular, a la cual se le puede modificar la ubicación, la intensidad, el color y el área de incidencia mediante el *Inspector.* El otro tipo de luz es el *SpotLight,* esta luz es la que muestra la figura 17, en la que se observa como la luz se crea en forma de cono, a esta tambíen se le pueden modificar todos los atributos desde el *Inspector.*

Figura 16: *Point Lighr* en Unity3D



Fuente: Propia

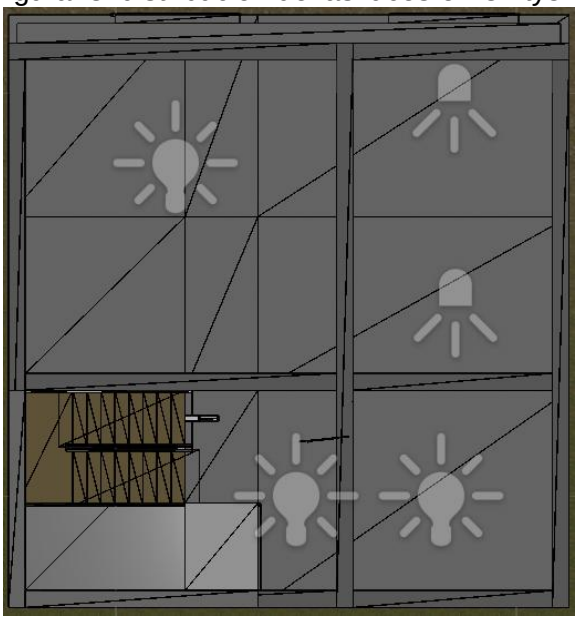
Figura 17: Spot Light en Unity3D



Fuente: Propia

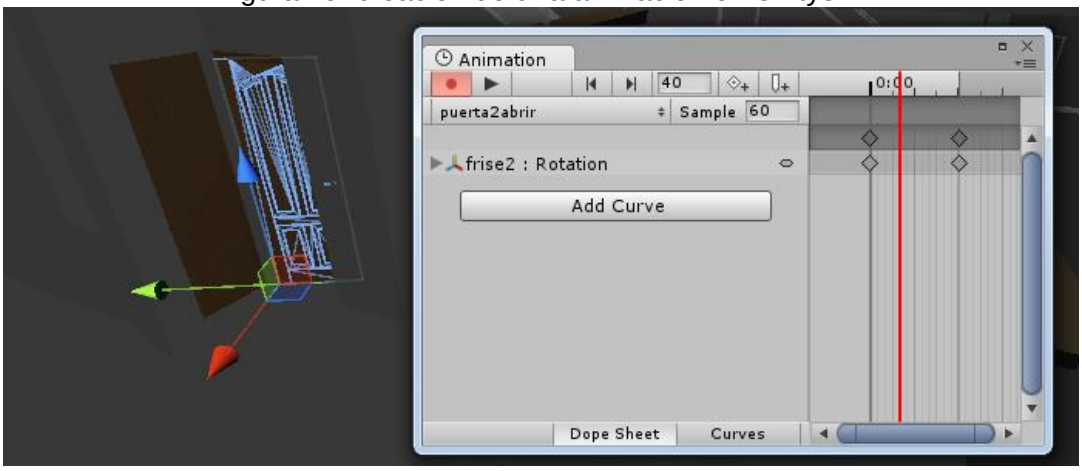
En el caso de este simulador se usaron 3 *Point Light,* y dos *Spot Light* en el orden que muestra la figura18.

Figura18: distribución de las luces en Unity3D.



Fuente: Propia.**3.2.3.4.2 Animaciones** Al cerrar y abrir una puerta se deben crear dos animaciones en ese modelo, una por cada acción. Para crear una animación se debe elegir el modelo específico al cual se animará, el inspector se va al menú *Add Component- Miscellaneous-Animation.* Una vez esto se procede a abrir el menú *Window-Animation,* la pantalla será parecida a la figura 19. Y en el menú desplegable que en esta imagen tiene por nombre “puerta2abrir” se debe dar click en *[Create New Clip],* allí se le da el nombre a la animación y se guarda el archivo. Una vez creado ese archivo se procede a dar click en *Add Curve*, en este caso se debe crear una rotación por lo tanto se sigue en *Transform-rotation.* En la linea de tiempo en 0 se deja el valor de la rotación que se desea, y más adelante se cambia el valor de la rotación deseada. Una vez terminado el proceso de edición de la animación se da click en el botón rojo, esto guardará la animación sin perder ningún cambio. Está animación para ser corrida en ejecución debe ser llamada por la programación que será discutida al final de este capítulo.

Figura 19: creación de una animación en Unity3D.



Fuente: Propia.**3.3 SENSORICA**

Para el diseño del simulador se tuvo en cuenta la creación de un módulo de domótica que se limita a tener sensores, pero no actuadores. Para realizar la parte de lectura de sensores se buscaba integrar algo que pudiese modular inclusive en un futuro, por eso se eligió una tarjeta de desarrollo.

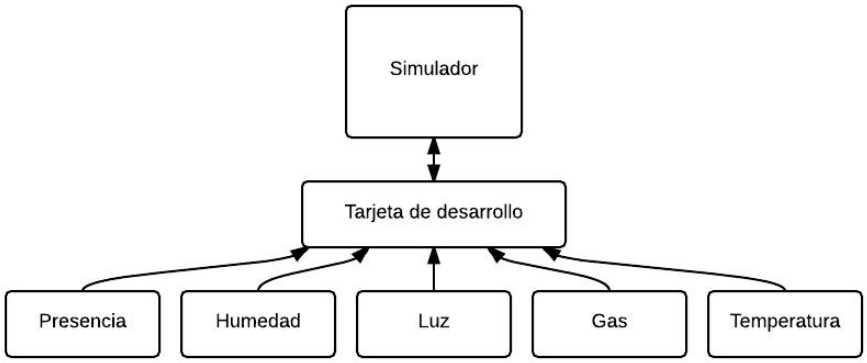
En la figura 20 se puede observar el esquema de los sensores usados en banco domótico. Se observa las cinco variables elegidas para que el simulador pudiese mostrar en pantalla de alguna forma. Estos sensores son leídos por la tarjeta de desarrollo y envía el valor de estos cuando el simulador lo requiera.**3.3.1Selección de sensores**

Para elegir los sensores se tuvo en cuenta que no fuesen sensores de protocolo (sensores knx, z-wave, x10, entre otros...), este criterio de selección es por el precio, puesto que un sensor sale del presupuesto, aún más teniendo en cuenta que es inversión propia.

Para cumplir principalmente se buscó usar sensores de bajo presupuesto por eso los sensores elegidos son:

- PIR-SE-10
- HIH-4030 de humedad.
- TEMT-6000 de intensidad luminosa
- MQ-2 de gases

- LM-35 de temperaturaFigura 20: esquema de implementación de los sensores



Fuente: Propia

Estos sensores garantizan una lectura apropiada para un ambiente domótico además de tener conexiones fáciles, garantizando tener unicamente 3 pines: Vcc, GND y Salida. Además los cinco trabajan con un voltaje de alimentación de 5V.**3.3.2 Calibración de los sensores**

A continuación se mostrará el proceso de calibración o identificación en cada uno de los sensores, así mismo se mostrará la ecuación de caracterización (si aplica al sensor).**3.3.2.1 PIR-SE-10**

Este sensor no necesita de calibración ya que su uso equivale a leer un estado alto(+5v) y uno bajo(0v). El sensor tiene un tiempo de espera de 2 segundos en los cuales no debe detectar movimiento, la lectura de ningún movimiento es de estado alto. Trascurrido ese tiempo al notar algún cambio en la luz infrarroja de la habitación activa la alarma que es cambiar al estado bajo. De esto se aprovecha para programar en Unity3D la acción pertinente.**3.3.2.2 HIH-4030**

Para la calibración de este sensor se uso como primer recurso la línea de caracterización que aparece en el Datasheet. La ecuación 1 representa la relación de voltaje/Humedad que posee este sensor.

$$HR = 26{,}66\,Vo - 13{,}33 \qquad (1)$$

Para comparar se uso el higrómetro RH101 perteneciente al laboratorio de Ingeniería Civil de la Universidad Piloto de Colombia, mostrado en la figura 21.

Figura 21: Higrómetro RH101

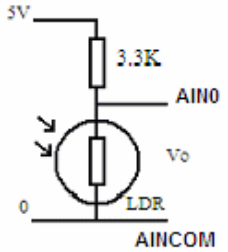
Fuente:Propia

Al comparar los valores de estos dos equipos se observa que el sensor de humedad tiene un offset de 3%, en otras palabras cuando el higrómetro muestra 50% de H.R. el sensor HIH-4030 muestra un valor de 50% de H.R. Esto se hizo a través de varias muestras, y como en los días de valoración no varió mucho la humedad relativa en los laboratorios, los valores oscilan entre 49% y 53% de H.R. donde se notó esta tendencia, por lo tanto el datasheet a comparación del higrómetro tiene unicamente ese pequeño offset, ya que la pendiente o relación de cambio si muestra ser óptima.

3.3.2.3 TEMT6000

El proceso de calibración de este sensor fue tomar los valores de luz que entrega un LDR con función al voltaje. La figura 22 muestra el montaje necesario para realizar esa medición[21]. Se colocaron los terminales de los sensores contiguos, haciendo que la incidencia de luz fuese lo más cercana posible.

Figura 22: Esquemático del circuito usado para calibración del sensor TEMT6000.



Fuente: Measure Light Intensity using Light Dependent Resistor (LDR)[21]

En el artículo online Measure Light Intensity using Light Dependent Resistor (LDR), el valor de Intensidad lumínica/voltaje esta dado por la ecuación 2, la cual disminuye el valor de voltaje de salida mientras la intensidad en lux aumenta. Está toma el valor del divisor de voltaje y una constante entre el LDR y la intensidad lumínica.

$$I[lux] = \frac{\frac{2500}{Vo[v]}}{3.3} \qquad (2)$$

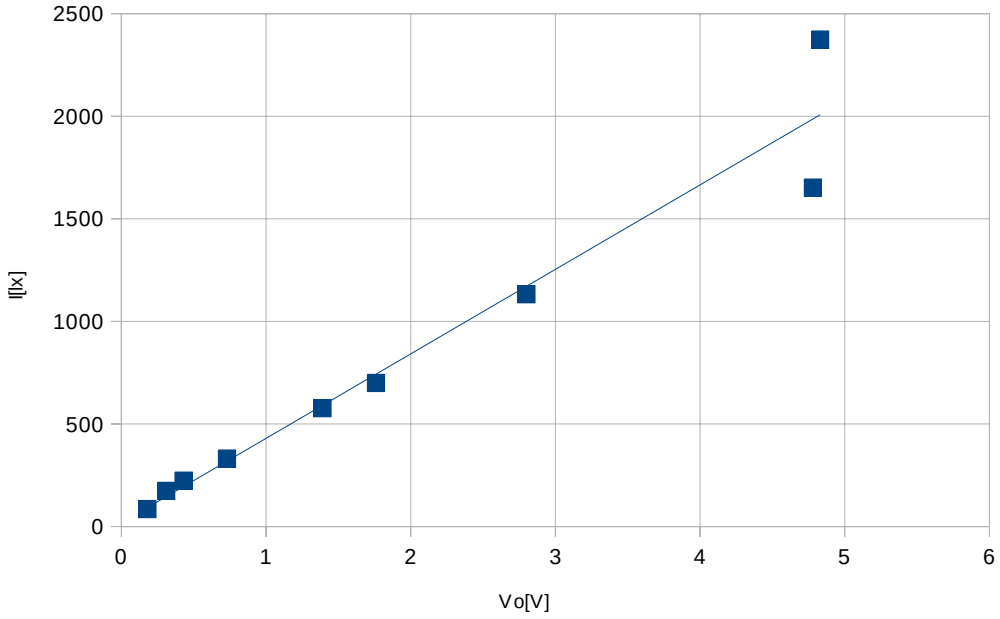
El proceso de identificación de este sensor se debe hacer paralelamente con la identificación del sensor, de esta forma el valor de la intensidad lumínica se verá reflejada en ambas salidas. Para crear la gráfica que muestra la figura 23 se usó la tabla 3, donde en la primera fila se tienen los voltajes obtenidos con el LDR, en la segunda el voltaje del sensore que se está identificando y finalmente el valor de la intensidad lumínica en lux. La gráfica resultante tiene la segunda columna en el eje *x* y la tercera en el eje *y.*

Tabla 3: tabla de identificación del sensor TEMT6000

Voltaje LDR (v)	voltaje en TEMT6000 (v)	Intensidad lumínica(lx)
3,2	0,18	85,227272727
2,33	0,31	173,6246586
2,02	0,43	223,52235224
1,57	0,73	331,01717815
1,04	1,39	576,92307692
0,89	1,76	699,69356486
0,59	2,8	1132,5115562
0,42	4,78	1652,2366522
0,3	4,83	2373,7373737

Fuente: Propia

Figura 23: Gráfico de dispersión para la identificación del sensor TEMT6000



Fuente: Propia

A pesar de mostrar los datos más elevados con mayor dispersión, en el resto de datos se observa que la relación es lineal. Por medio de la herramienta de hojas

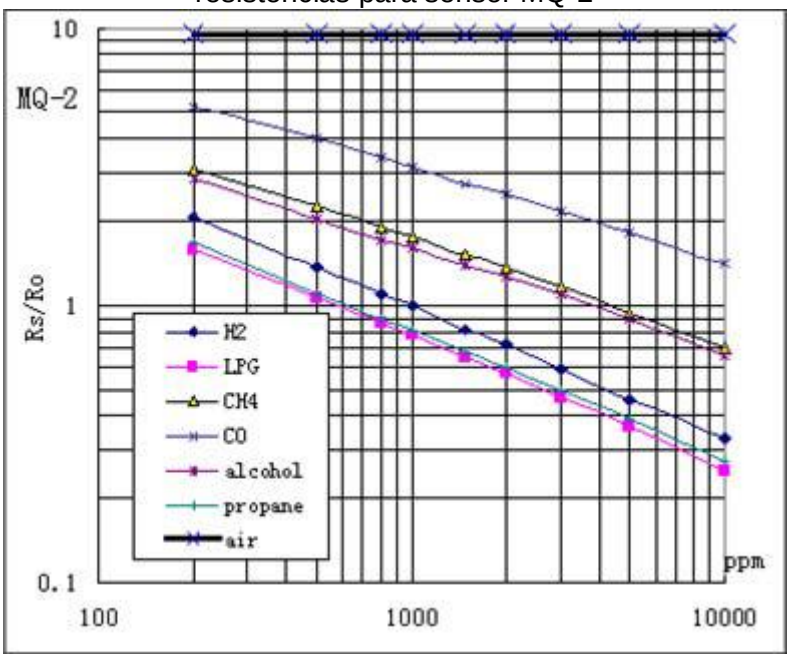
de cálculo se obtuvo la ecuación 3, la cual será útil a la hora de programar el sensor al simulador.

$$I[lx]= 411(Vo[v])+17,83 \quad (2)$$

3.3.2.4 MQ-2

El datasheet de este sensor indica que para poder configurar el sensor se debe usar una resistencia de calibración, es decir se debe ir probando con un potenciómetro hasta crear un punto de alarma. La figura 24 muestra la sensibilidad del sensor a distintas relaciones de resistencia. Para distintos gases

Para poder usar el sensor es recomendable tenerlo conectado por 12 horas para que el elemento más importante del sensor este activo.Figura 24: Relación de resistencias para sensor MQ-2



Fuente: Datasheet sensor MQ-2

La resistencia ideal para esta práctica es de 10KΩ, esta permite una lectura por debajo de un 1V para aire común en Bogotá no importando el tiempo de calentamiento, con esto se garantizará en el simulador una optima programación

de fugas o detección de humo en el ambiente. El fabricante mediante el datasheet y *Arduino Playgroud* [22] recomiendan usar el sensor con puntos de alarma, es decir como detector de gases o humo, más no para obtener una lectura fiable del nivel ppm en el aire de cualquier sustancia que el sensor pueda leer. **3.3.2.4 LM-35**

Este sensor desde el datasheet indica que tiene cambios de 10mV por grado Celsius, no fue necesario si no medir el voltaje de salida y usar un termómetro para verificar que esta línea de caracterización siempre se cumple, haciendo el sensor aún más atractivo de usar, por su sencillez de implementación.**3.3.3 Integración con la tarjeta de desarrollo**

La razón para elegir una tarjeta de desarrollo sobre un microcontrolador es su modularidad, es decir con un microcontrolador hay que crear una serie de periféricos o diseñar una tarjeta PCB desde ceros para poder hacerlo modular. Por eso se comparan las dos tarjetas de la tabla 4.

Tabla 4: comparativa de tarjetas de desarrollo

Dispositivo	ST32FDiscovery	ARDUINO UNO
Lenguaje de programación	MikroC for ARM, C for STM32Cube	Processing
Modulos(que se usarán en el proyecto)	ADC, UART.	ADC, UART
Voltaje de trabajo	0-3voltios	0-5voltios
Costo	En la página oficial $14.58USD. En Colombia entre $50.000COP y $75.000COP	En la página oficial $22,50USD. En Colombia al rededor de $69.000COP.

Fuente: Propia.

Si se evalúa la velocidad y el precio de cada tarjeta, la elegida es la tarjeta ST32FDiscovery, y fue la herramienta inicial de experimentación en el presente proyecto. Pero se elige ARDUINO UNO por dos razones simples: al programar la tarjeta ST con MikroC (Lenguaje con que fue enseñada en el programa de Ingeniería Mecatrónica de la universidad Piloto de Colombia) se presentan errores en la librería de puerto serial, ya que al valor máximo al que se puede llegar es de 900 baudios; la segunda razón es que al usar sensores con salida de 5 voltios, se

tendría que crear una etapa de escala para que la tarjeta ST pudiese leer estos valores, creando un problema a la hora de hacer un diseño modular.**3.3.3.1 Implementación con ARDUINO UNO**

El programa que se creo para la lectura y envío de las variables de los sensores tiene una lógica de contadores. Lo primero que hace el programa es verificar que la bandera que permite leer y enviar los valores del ADC sea la adecuada, de lo contrario no realizará ninguna acción. Cuando la bandera de lectura es la correcta, lee un sensor y envía su valor, seguidamente aumenta en 1 el contador; este contador tiene la función de cambiar la lectura y el envío de información de cada sensor por cada ciclo de ejecución; i.e. en el primer ciclo de ejecución recibe la bandera de lectura, seguidamente lee y envía la información del sensor de GAS, a continuación vuelve a empezar el ciclo, y espera a que llegue la bandera de lectura, si esta llega, continua con la lectura de otro sensor, siendo así hasta que lee cada sensor, finalmente volviendo al sensor de GAS.

3.3.4 Banco domótico

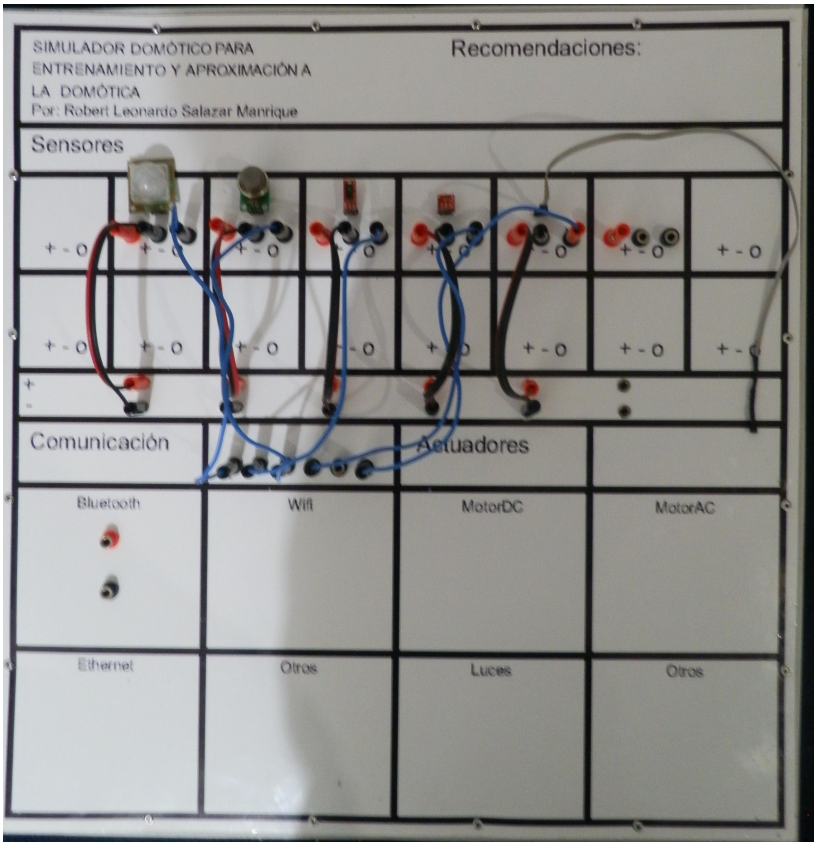
Para el diseño del banco domótico se usaron los parámetros de diseño:

- Modificable, se pueden agregar o quitar componentes.
- Modular, se puede agregar cualquier sensor, actuador o procesador sin necesidad de crear otro.
- Fácil de armar.
- Permite la conexión de cables entre los distintos módulos.

Ya una vez establecidos se procede a crear una caja con dimensiones 50cm X 50cm X 5cm en acrílico blanco.

Como dos de los parámetros de diseño buscan facilidad de creación y espacio de trabajo, se busco crear un adhesivo que tuviera indicado el espacio asignado a cada módulo. Por lo tanto se dejan líneas delimitando el espacio para : alimentación, sensores, actuadores y comunicación, puesto que el procesador debe ir debajo no se indica donde va este. El adhesivo se creó en Autocad y fue impreso en un papel que pegó directamente sobre el panel acrílico de 50cm X 50cm.

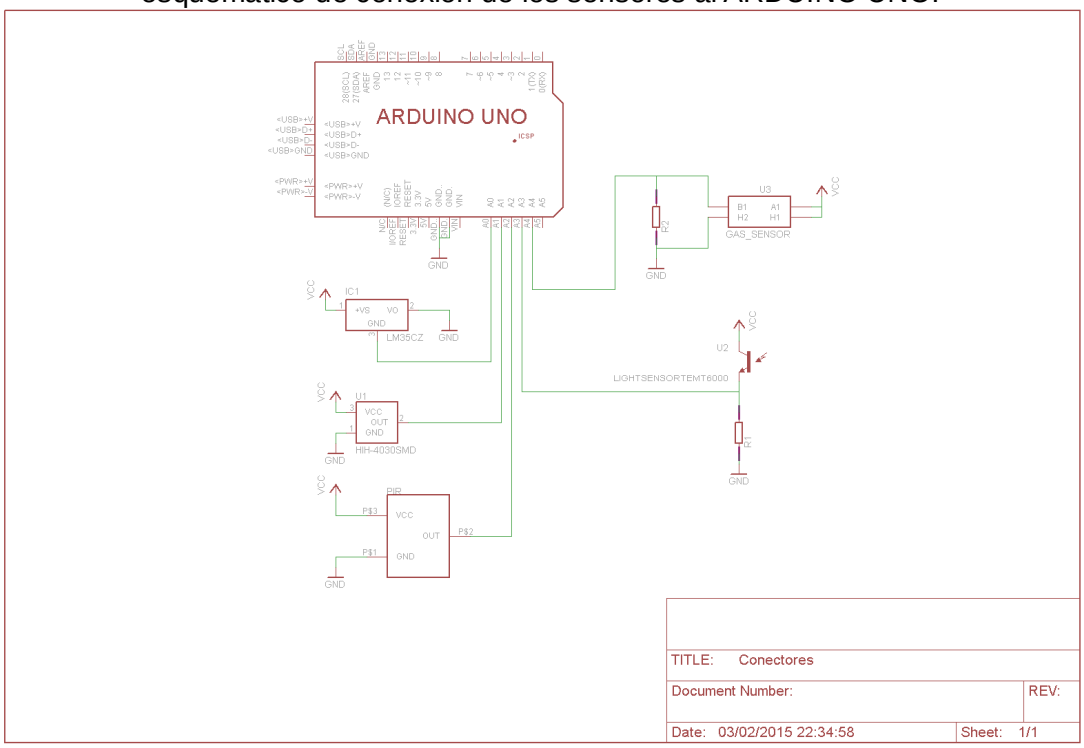
Una vez puesto el adhesivo en su lugar se procede a taladrar el panel, de modo que se creen los agujeros que tendrán las borneras de conexión, en este caso la broca usada es de ¼". Posteriormente se procede a realizar las conexiones con los sensores, de esta forma se puede conectar y desconectar cada uno de una forma rápida. En la figura 25 se observa una fotografía del panel con los sensores conectados a la alimentación y a los terminales analógicos del ARDUINO UNO, además de las borneras de Rx-Tx del módulo Bluetooth. Figura 25: Panel domótico listo para la práctica de sensores.



Fuente: propia.**3.3.4.1 Conexiones**

Las conexiones necesarias para que el panel funcione se deben hacer por debajo del panel, donde el esquemático de la figura 26 indica como deben quedar los sensores para que el simulador sincronice de la forma correcta. Haciendo que la alimentación se reparta en las líneas del medio, y los seis puntos análogicos del ARDUINO UNO en las borneras ubicadas encima de el espacio destinado al Wifi.

Finalmente como se había mencionado los puntos Rx-Tx del módulo Bluetooth a las dos borneras en el espacio delimitado a este protocolo de comunicación. La alimentación en este caso, por la baja demanda de potencia por parte de los sensores fue tomada desde el ARDUINO, si desea usar fuente externa para la alimentación de los sensores lo único que se debe hacer es acoplar las tierras y conectar el positivo de la fuente a la línea de alimentación.Figura 26: Plano esquemático de conexión de los sensores al ARDUINO UNO.



Fuente: Propia.

3.4 CONTROL REMOTO

Una parte pensada para el banco domótico y hacerlo algo más parecido a los sistemas domóticos actuales, fue la creación de un control remoto implementado en una plataforma móvil (Tablet, o SmartPhone), comunicado inalámbricamente al simulador.**3.4.1 Capa física**

El cuadro de la tabla 1 compara los protocolos inalámbricos más usados para

aplicaciones *Indoors*, Como se puede observar el que tiene mejor transferencia de datos es Wi-fi, pero implementar una red Wi-fi requiere más recursos que implementar una red ZigBee o Bluetooth. Como parametro de diseño se quiso implementar una plataforma móvil, se descarta inmediatamente Zigbee, ya que los celulares o smartphone vienen integrados con modulos bluetooth y Wi-fi, pero no con modulos Zigbee.

Tabla 5: comparativa de protocolos de comunicación.

Standard	**Bluetooth**	**Zigbee**	**Wi-Fi**
Frecuencia de banda	2.4GHz	868/915 MHz; 2.4 GHz	2.4 GHz; 5 GHz
Rango nominal	10m-30m	10-100 m	100 m
Máximo de células por nodo	8	> 65000	2007
Ancho de banda	1MHZ	0.3/0.6 MHz; 2MHz	22MHz

Fuente:Propia.

Por ser un módulo más optimo para esta aplicación y por tener módulos incluidos en plataformas móviles se usa Bluetooth. Se dice que más optimo ya que se requieren largas distancias para la comunicación, y el sistema no requiere una tasa de transferencia demasiado elevada. **3.4.2 Conexión de módulos**

Para que el computador pudiese comunicarse mediante Bluetooth fue necesario el uso de dos módulos de comunicación: uno Bluetooth y uno serie RS-232. Los módulos usados fueron el RN-41 y el PL2303.**3.4.2.1 Módulo RS-232**

Para poder usar el puerto serie desde cualquier programa es recomendable ver que puerto se está usando desde el Administrador de archivos de Windows, en la pestaña *Puertos(COM y LPT)* aparece el puerto serie asignado al dispositivo una vez instalado.**3.4.2.2 Módulo Bluetooth**

Para poder usar este modulo en transmisión con un celular es necesario configurarlo desde un programa terminal, en este caso se usa TeraTerm. El dispositivo debe estar previamente emparejado como cualquier otro dispositivo Bluetooth®. Esto se puede hacer desde el panel de control buscando Bluetooth®, allí nos da la opción de agregar un dispositivo Bluetooth®.

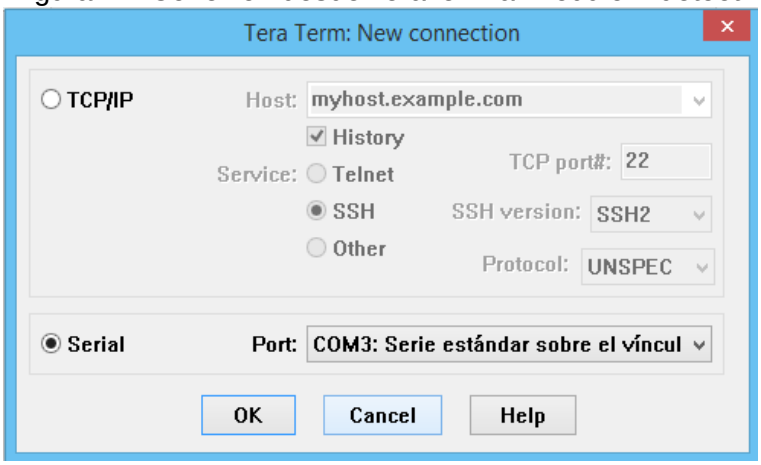
El siguiente paso es conectar el modulo a una batería de 5V. Una vez conectado el módulo, un LED verde de *Status* comienza a titilar al rededor de una vez por segundo. Que el módulo titile de esa forma es debido a que este espera un minuto, si se pasa de ese minuto no se puede entrar a modo de configuración, por lo tanto hay que reiniciarlo.

Desde TeraTerm se debe iniciar la conexión con el módulo Bluetooth, se elige la opción *Serial* como indica la figura 27, allí se debe elegir el puerto asignado al módulo Bluetooth en el administrador de dispositivos. Una vez se establezca la conexión el LED “connected” prenderá automáticamente.

Cuando la conexión este establecida se procede a entrar en modo configuración, esto es posible al tipear $$$(tres signos pesos) en el terminal. El módulo indicará que ha entrado en modo configuración enviando las letras “CMD”. Si no llega a enviar las letras que se esperan hay que volver a repetir el proceso desde el inicio.

Ya una vez en modo configuración se deben usar mayúsculas. Hay dos comandos básicos: escribir H seguido de enter, o D seguido de enter. H envía la ayuda para configuración, y D envía el estado actual en configuración del Bluetooth®.

Figura 27: Conexión desde TeraTerm al módulo Bluetooth



Fuente:Propia.

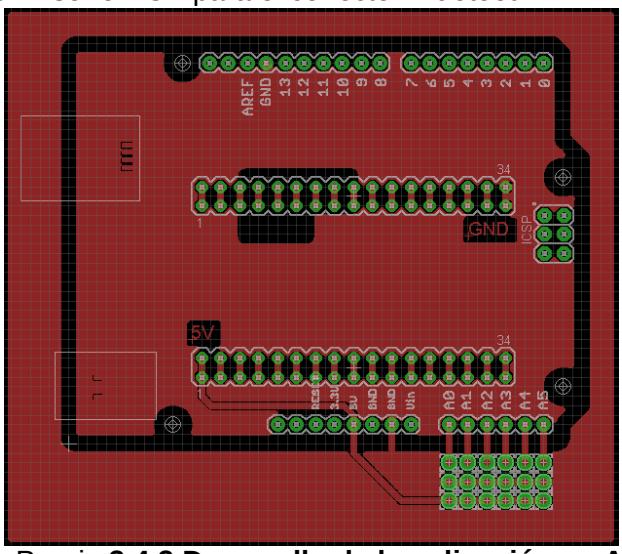
Un ejemplo de configuración puede ser cambiar el nombre del dispositivo. Para este fin se escribe:"SU, Mi Bluetooth" seguido de la letra enter. En caso de ser positivo el cambio el modulo muestra "AOK" de no ser así, enviara "ERR". Para visualizar si el cambio realizó efecto ve el estado actual del Bluetooth® mediante el comando D.

La configuración realizada para esta práctica es la que muestra la figura[], donde se visualizan las variables de uso, donde los valores más importantes son: PinCod: que es el código para emparejar el dispositivo, Baudrt que es la velocidad de transferencia y el último parámetro configurado es el nombre del dispositivo.Figura 28: Configuración del módulo RN-41

Fuente: Propia.

La ventaja del dispositivo RN-41 es que permite el uso de aplicaciones Android y iOS. Para eso es necesario conocer el UUID (código único por aplicación con Bluetooth). El UUID por defecto en ese módulo es: “00001101-0000-1000-8000-00805F9B34FB”; se puede ver que es un número de 128 bits, que puede ser modificado en los primeros 8 valores desde la configuración Bluetooth, pero para efectos prácticos se dejará el valor por defecto.

Finalmente se diseño una tarjeta para conectar el módulo Bluetooth a manera de Shield de ARDUINO, esta fue realizada en Eagle. Ya que Eagle no posee librerías para este módulo se usaron conectores estándar a la médida del módulo RN-41. La figura 29 muestra el diseño final de la PCB creada para la conexión de este módulo. Figura 29: Diseño PCB para el conector Bluetooth



Fuente: Propia.**3.4.2 Desarrollo de la aplicación en Android**

El parámetro de diseño más importante para el control remoto es el uso de un móvil, en la tabla 6 se realiza una comparación de las tres plataformas móviles más utilizadas actualmente, se exponen los puntos más relevantes para seleccionar una entre las tres.

Tabla 6: comparativa de plataformas móviles

Plataforma	Windows Phone	iOS	Android
Lenguaje de programación	C#.NET, VB.NET	Objective C, Swift	Java

Plataforma de desarrollo	Visual Studio 2012+	XCode	Android Studio, Eclipse.
Bluetooth	Si	Si	Si
Costo de desarrollo	Se puede desarrollar en una versión gratuita, para licencia comercial se debe obtener visual Studio con valor minimo de $699USD	Se puede usar XCode gratuitamente para desarrollo, para subir aplicaciónes Apple cobra $100USD anuales.	No requiere licencia, para subir aplicaciones se debe pagar a google $24USD.

Fuente: propia.

La plataforma elegida es Android, sobre todo por el libre uso de sus herramientas de desarrollo, a pesar de que Apple y Microsoft soportan programas educativos, el uso comercial de cualquiera de las dos herramientas requiere pago. Por parte de Android no se requiere si no el pago para poder subir las aplicaciones a la nube de Google Play™.

Para desarrollar en Android Studio se recomienda revisar la plataforma destino, esto es de gran importancia ya que entre más bajo el nivel API más gente lo podrá usar, pero no serán accesibles todas las librerías creadas por Android para las versiones siguientes. La tabla 7 muestra el porcentaje de dispositivos con cada versión de android.

La aplicación del simulador se desarrolló en android Gingerbread API 10, puesto que se puede dirigir al 99,5% del público. Cabe destacar que Android Studio viene integrado con simuladores de la aplicación, pero el módulo Bluetooth no se puede simular allí, esto se debe a que el dispostivo simulado no posee ningún módulo Bluetooth incorporado, es por eso que todas las pruebas de desarrollo Bluetooth se deben realizar en un celular o una tablet real. En este caso se uso un Samnsung galaxy Ace de primera generación para el desarrollo, y un Huawei g526 para las pruebas.

Tabla 7: Porcentaje de distribución de las versiones de Android en el mercado.

Version	Codename	API	Distribution

2.2	Froyo	8	0.5%
2.3.3 - 2.3.7	Gingerbread	10	9.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	7.8%
4.1.x	Jelly Bean	16	21.3%
4.2.x		17	20.4%
4.3		18	7.0%
4.4	KitKat	19	33.9%

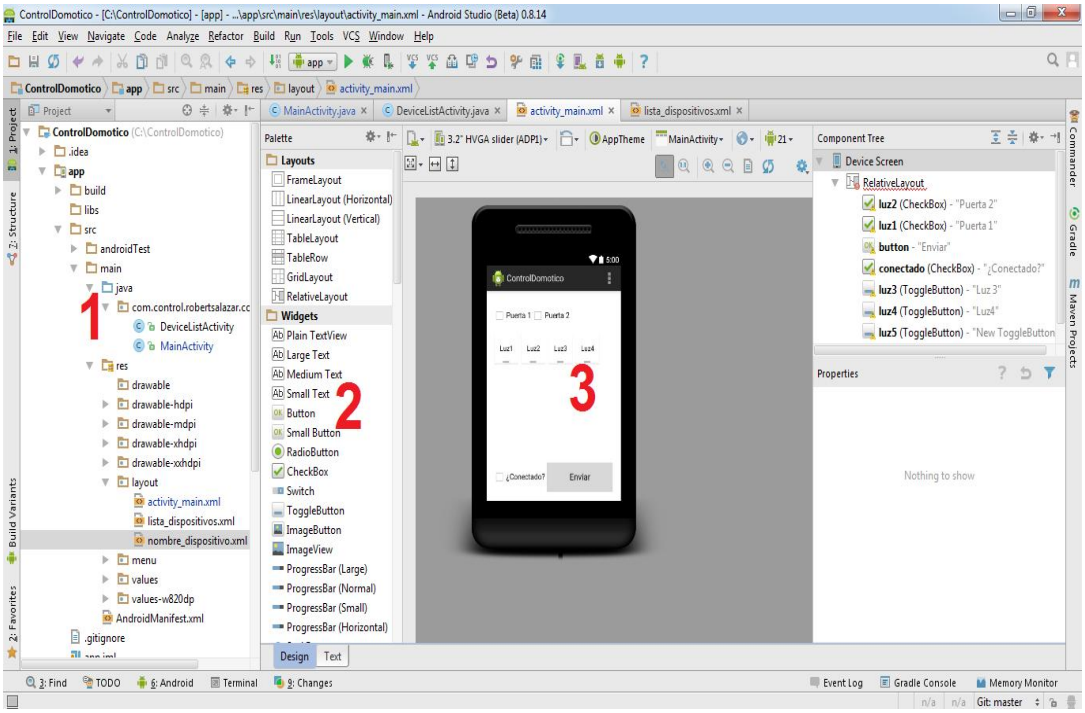
Fuente: Documentación de Android Developer [18]**3.4.2.1 Creación de la interfaz**

Si desea crear una aplicación desde cero, remítase primero al anexo D que muestra el proceso de creación de un proyecto en Android Studio.

El primer paso para crear una aplicación es crear la interfaz. La figura 30 indica: 1) el sistema de archivos que maneja Android Studio, 2) los elementos visuales que se pueden agregar a la aplicación, 3) el diseño de la pantalla de la aplicación.

El diseño de la aplicación fue realizada en base a las variables que se manejan en el simulador: 4 luces y 2 puertas. Por eso se ubican 2 *checkbox* y 4 botones, que funcionarán como switch para estas 6 variables.

Una vez culminado el diseño visual de la aplicación se prosigue por programar la acción de los botones y de las aplicación en general en un archivo .java.Figura 30: Interfaz de usuario de Android Studio.



Fuente: propia

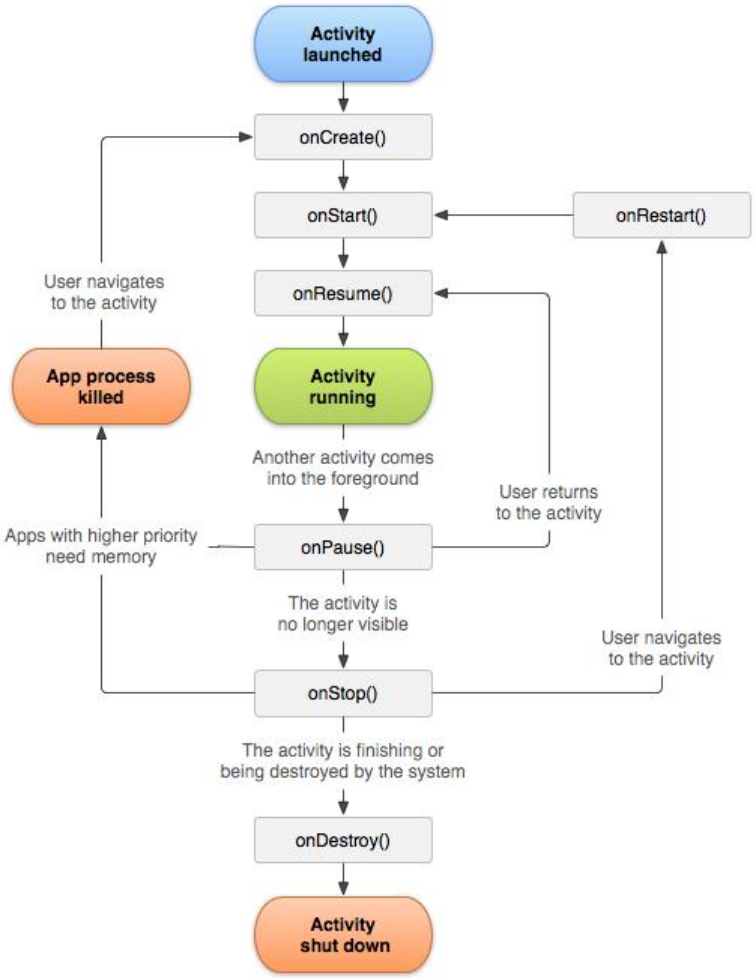
3.4.2.2 Programación de las *Activities*

En el desarrollo de aplicaciones para Android se debe tener claro el concepto de *Activity*, el cual consiste en cada pantallazo que se tenga en la aplicación, de modo que la interfaz mostrada en la figura 29 viene siendo la *Activity* principal de la aplicación.

La documentación de Android al explicar que es una *Activity* usa el diagrama de flujo de la figura 31, este va mostrando que tiene varios métodos que se van ejecutando al correr el *Activity* en la aplicación.

Para la programación de la *Activity* principal de la aplicación del simulador se usan los métodos: *onCreate(), onStart() y onStop().* Esto no quiere decir que unicamente se usen estos métodos, ya que para realizar más acciones en la aplicación no solo se usan estos métodos.

Figura 31: Diagrama de flujo para la programación de *Activities*



Fuente:Documentación de Android Developer [18].

El método *onStart()* unicamente se encarga de verificar si el modulo Bluetooth de el celular o tablet está habilitado, de no ser así envía una caja solicitando al usuario que encienda este módulo.

El método onCreate realiza las tareas indicadas en la figura 32. Cabe mencionar que es un proceso estándar programar las acciones de los botones en este

método, ya que no hay un método que se ejecute periódicamente en la aplicación.

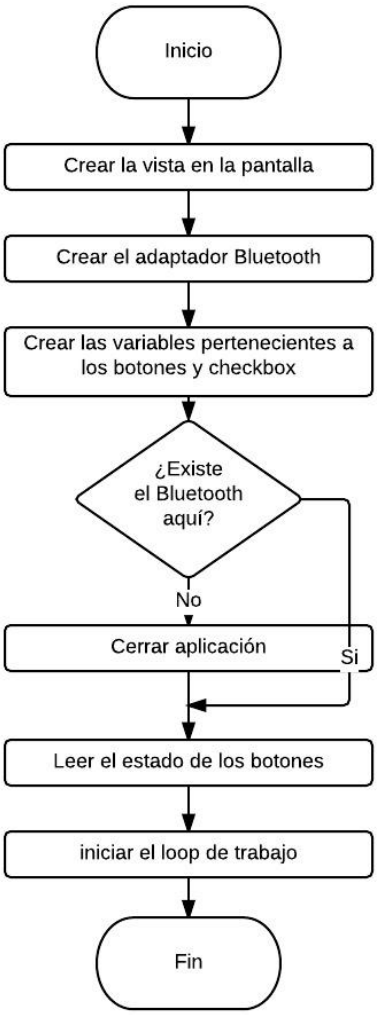
Como Android no posee un método estándar que se repita periódicamente hay que iniciarlo desde el método *onCreate()* y detenerlo en el método *onDestroy()*. En este método se envía el identificador de cada variable manejada desde el control remoto, seguidamente de su estado. El identificador de las variables para está aplicación, son letras mayusculas que van desde la 'A' hasta la 'F', y el estado es On/Off, representados por una 'L' y una 'M' respectivamente.

Se creó además una *Activity* complementaria a la principal, esta se puede activar desde: el botón menú->Conectarse a. El diseño de está *Activity* es el mostrado en la figura 33, y está encargada de mostrar los dispositivos previamente emparejados con el celular, y al elegir uno enviar la dirección MAC de ese dispositivo Bluetooth a la *Activity* principal.

Existe un método encargado de manejar el resultado proveniente desde otras *Activities,* este método toma la dirección MAC proveniente de la *Activity* secundaria y empieza una conexión asíncrona con el dispositivo Bluetooth. Esto es posible sí y solo sí el identificador UUID coincide.

Si desea ver el código fuente de la aplicación en Android desarrollada para el simulador, puede remitirse al repositorio en GitHub: https://github.com/Robmotor/ControlDomotico

Figura 32: Diagrama de flujo del método *onCreate()*



Fuente: Propia.

Figura 33: Diseño de la interfaz de la actividad secundaria.

Fuente: propia.**3.5 INTEGRACIÓN CON EL SIMULADOR**

El último proceso para la realización del simulador es integrar los 3 componentes principales del sistema. Primero es la integración del Motor de videojuegos con el módulo Bluetooth y luego con los sensores.**3.5.1 Integración de Unity3D con el control remoto**

La integración del control remoto basta con crear un *Script* en Unity3D, mediante: click derecho en *Console-Create-C# Script.* Este abrirá el código en MonoDevelop. Para programar en Unity 3D se debe tener en cuenta los dos métodos principales: *void Start ()* y void Update ().

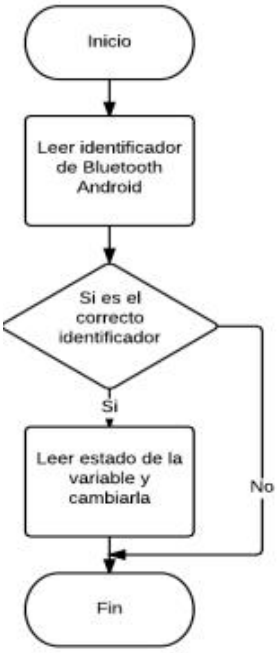
En el método *void Start()* unicamente se inicia el puerto serie, ya que las variables que se usen en el código es preferible usarlas como variables globales en esa clase.

Como se piensan manipular luces y animaciones se deben agregar los objetos *GameObject* al código, estos crean una referencia al objeto en la escena. Se declaran como :

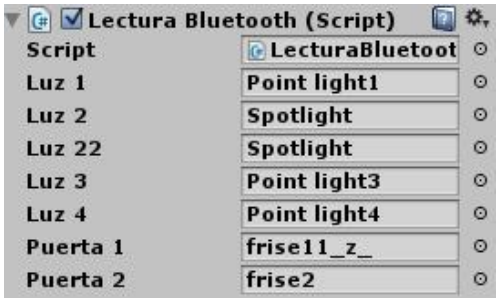
public GameObject “nombre del objeto que será usado en el código”;
Este tipo de objeto se debe crear por cada variable que se piense manejar.

El método *void Update()* ejecuta el diagrama de flujo de la figura 34, el cual se encarga de leer el identificador que provenga desde el control remoto, y cambiar el estado de esa variable. Para cambiar la luz es necesario primero referenciar el *GameObject* de la escena y ejecutar la línea:
“nombre del *GameObject*”.light.enabled = “Valor Booleano”;
Al igual que para ejecutar una animación se debe correr la línea con el Gameobject ya asignado:
“nombre del *GameObject*”.animation.Play("Nombre de la animación");
Finalmente hay que compilar el código, y desde *Console* se debe arrastrar el Script al *Character Controller,* una vez realizado esto se puede ver en *Inspector* que se pueden vincular los objetos desde *Hierarchy* como indica la figura 35.

Figura 34: diagrama de flujo del método *Update()* para la comunicación Bluetooth.



Fuente: PropiaFigura 35: Script compilado y añadido al *Character Controller*



Fuente: Propia.

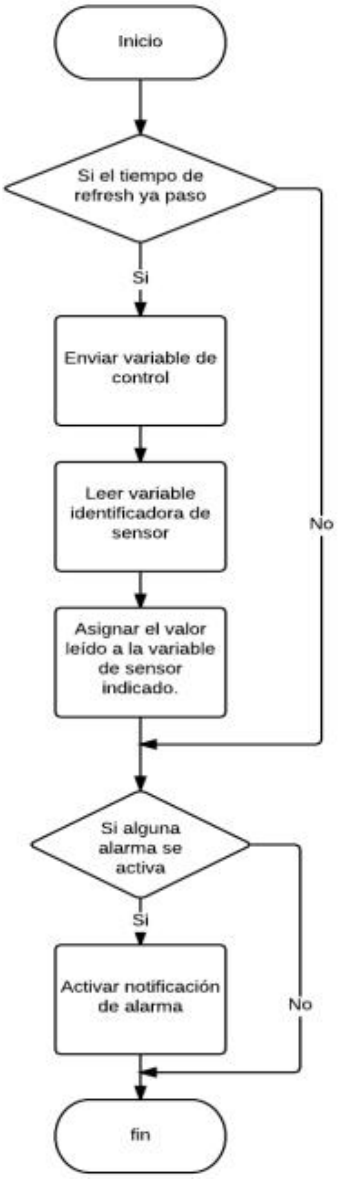
Los comandos que crean, leen y escriben el puerto serial son los que Microsoft proporciona en la documentación de C# para Visual Studio.**3.5.2 Integración de Unity3D con la tarjeta de desarrollo**

Ya que se desea mostrar en pantalla el valor de cada uno de los sensores que lea la tarjeta se deben crear los elementos *GUIText* en la escena. Para crearlos se debe ir a *GameObject-Create Other- GUIText,* estos elementos no se programan en ninguno de los dos métodos anteriormente mencionados, pero si se deben referenciar mediante un *Public GameObject* en el *Script.* Cuando se elije un *GUIText* en Hierarchy, mediante *Inspector* se pueden cambiar las propiedades de color, tamaño de letra y ubicación de ese objeto para tener una Interfaz más fluida.

Antes de programar el método *OnGUI()* se debe programar el método *Update(),* ya que este estará encargado de leer el valor del ADC de 10 bits que envía la tarjeta, y traducir este valor mediante las ecuaciones halladas para cada sensor. La lógica usada en este método se ve en el diagrama de flujo de la figura 36. Como se había mencionado el sensor envía el carácter que inicia la comunicación, seguidamente empieza a leer y calcular el valor de cada sensor.

El método OnGUI() es el encargado de cambiar los textos, cajas y botones que se creen en la interfaz de Unity3D, en este caso unicamente se piensa cambiar el valor que tienen unos textos en pantalla. Para eso se usa la línea de código:
“Nombre del *GameObject*”.guiText.text = "Sensor de “Nombre del sensor”: " + “Valor del sensor”.ToString("N");
Esta línea de código toma el valor previamente leído, lo convierte en un *String* de solo dos decimales y lo deja en un mensaje introductorio.

Figura 36: diagrama de flujo del método Upadte para la comunicación con la tarjeta de desarrollo



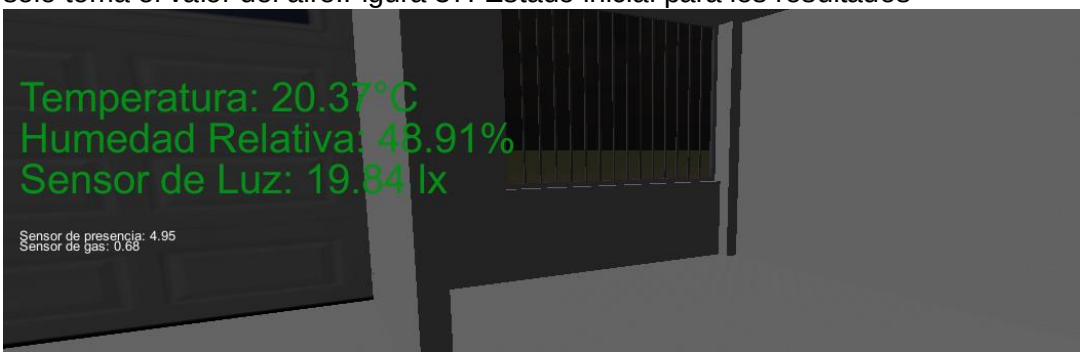
Fuente:propia

4. RESULTADOS, CONCLUSIONES E IMPACTOS4.1 Resultados

Para entender mejor los resultados se recomienda ver los anexos B y C, que son el manual de usuario y una práctica inicial creada específicamente para el sistema de entrenamiento con los elementos de los que hasta ahora se compone. Para los resultados se asume que todas conexiones están realizadas.**4.2.1 Práctica de sensores**

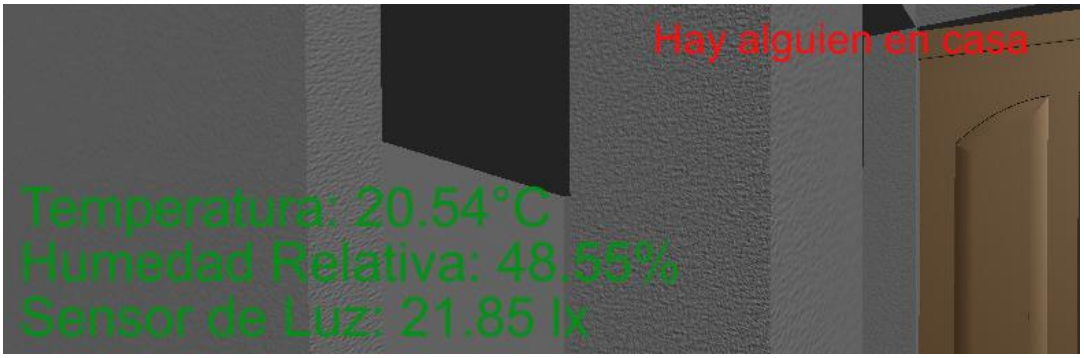
Para el uso de cada sensor se muestra un estado inicial de lectura de variable, y un proceso de modificación de esta variable, y como cambio la variable después de ese proceso. Se denota que se apagan las luces :1, 2 y 3. Ya que la dos cambia proporcionalmente al cambio de luz en el sensor de intensidad luminíca.

En la figura 37 se ve el estado inicial de la simulación, donde: La temperatura ambiente es de 20,37°C, la humedad relativa es de 48,91%, el sensor de luz no detecta si no 19,84lx, el sensor de presencia no detecta a nadie y el sensor de gas solo toma el valor del aire.Figura 37: Estado inicial para los resultados



Fuente: Propia.**4.2.1.1 Sensor de presencia**

Para el sensor de presencia se genera movimiento por encima de este, haciendo que detecte el movimiento de la mano, en la parte superior de la interfaz en ejecución el programa muestra el mensaje en rojo: “Hay alguien en casa”, tal cual muestra la figura 38.Figura 38: Demostración del sensor de presencia



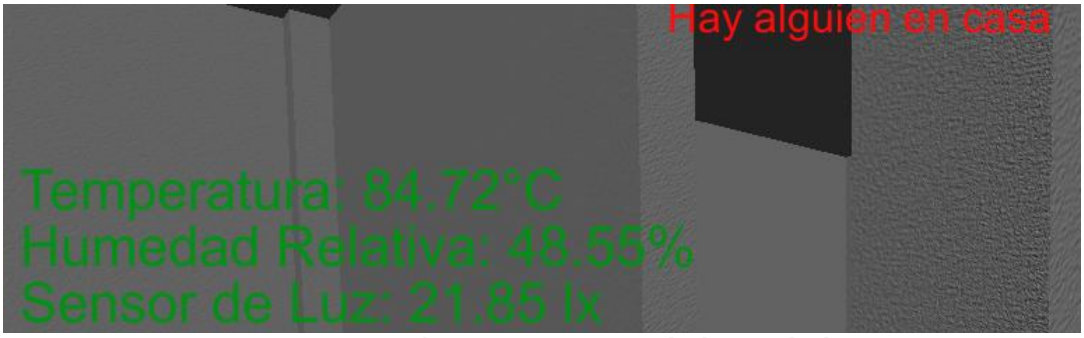
Fuente:Propia

4.2.1.2 Sensor de gas

A este sensor se le habilito una especie de alarma, que cuando detecte el gas por encima del nivel permitido, este envía un color rojo de ambiente, tal como muestra la figura 39. También se observa que el sensor de presencia se activa por que al tener los sensores tan cerca, se alcanza a activar el otro por el simple movimiento.Figura 39: Demostración del sensor de gas.

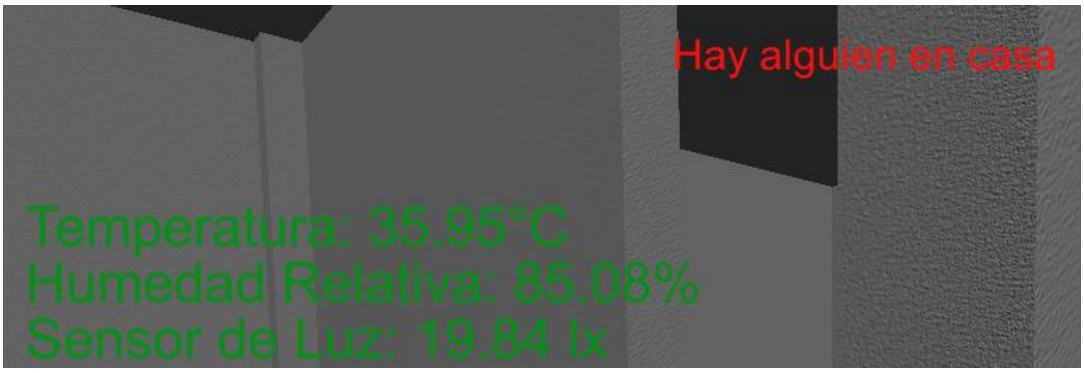
Fuente: Propia.**4.2.1.1 Sensor de temperatura**

Con el sensor de temperatura se busca elevar su valor mediante el uso de un mechero convencional. Esto es posible ya que este sensor en el Datasheet tiene un rango de trabajo de 0°C a 150°C. Como se observa en la figura 4 el valor de la temperatura se ha elevado. Se ve que el sensor de presencia aun detecta movimiento y no se ha estabilizado en el punto que no detecta a nadie.Figura 40: Demostración del sensor de temperatura



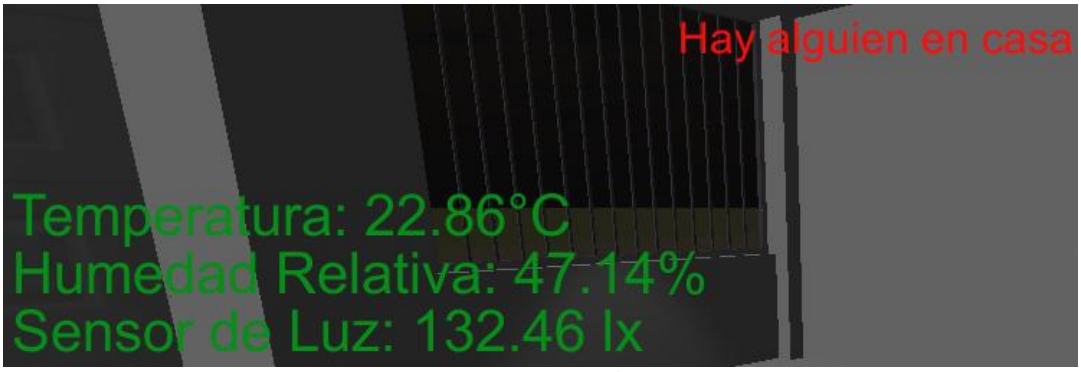
Fuente: Propia.**4.2.1.1 Sensor de humedad**

El proceso para manipula cambio para la humedad relativa del aire es soplar sobre este sensor, ya que el aire inhalado pasa por los pulmones, al ser exhalado este tiene una concentración de humedad mayor a la del aire en el ambiente. En la figura 41 se observa el cambio de la variable de humedad, también se observa en esa imagen que el sensor de temperatura aun se está enfriando.Figura 41: Demostración del sensor de humedad



Fuente: Propia**4.2.1.1 Sensor de intensidad lumínica**

En todas las anteriores demostraciones se puede ver que el valor de intensidad de luz está entre los 20lx, esto se debe a que la luz del cuarto estaba apagada mientras se realizaba la práctica. Para este sensor se usaron dos demostraciones, un bombillo de baja potencia, y uno de potencia regular, estas demostraciones se pueden ver en las figuras 42 y 43 respectivamente. Donde se puede ver que la magnitud de la luz es proporcional al valor leido en lux. Figura 42: Primer demostración del sensor de Intensidad Lumínica

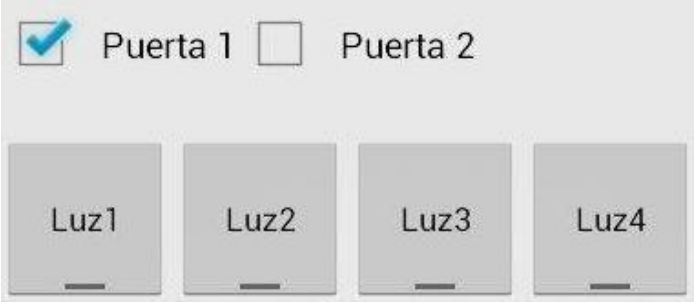


Fuente: Propia

Figura 43: Segunda demostración del sensor de Intensidad Lumínica

Fuente: Propia.**4.1.2 Control Remoto**

En el control remoto se quería evaluar la respuesta del simulador de Unity3D con la aplicación de Android. Para eso se debe conectar el módulo Bluetooth e iniciar el simulador. Una vez hecho esto se procede a probar las animaciones de las puertas. **4.1.2.1 Puertas** la evaluación en este punto es si la puerta está chuleada como muestra la figura 44 se abre, de lo contrario se cierra. Además la animación solo debe ocurrir una vez por selección.Figura 44: Botón seleccionado para abrir la primer puerta



Fuente: Propia.

El resultado de la acción anterior es abrir la puerta 1. La figura 45 es un pantallazo de cuando se ejecuta esta acción, mientras se tienen el resto de variables apagadas, y se mantienen conectados los sensores.Figura 45: Prueba de control remoto con la primer puerta

Fuente: Propia

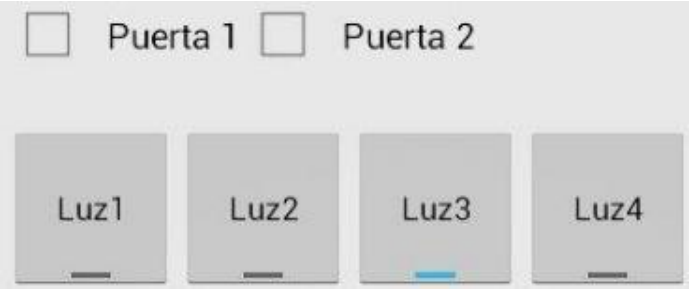
La segunda prueba fue abrir ambas puertas, estas hacen la animación al tiempo como muestra la figura 46, como en la prueba anterior, todas las luces estaban apagadas.Figura 46: Prueba del control remoto con ambas puertas

Fuente: Propia.

Con las pruebas realizadas se demuestra que el control remoto sirve para correr las animaciones de las puertas sin ningún problema y leyendo los sensores.**4.1.2.1 Luces**

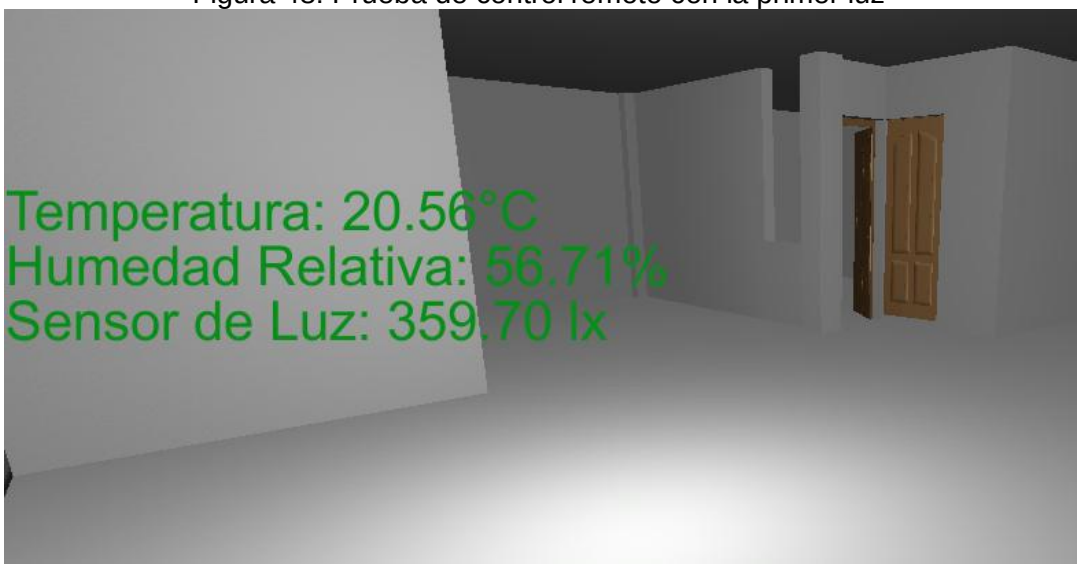
Para las luces se hizo un procedimiento similar. En la primer prueba se prendió unicamente la luz 1 que corresponde al hall de entrada. El control se muestra en la figura 47, y la respuesta se puede observar en la figura 48, donde también se muestra que se puede abrir la puerta 1 al mismo tiempo.

Figura 47: Botón seleccionado para prender la primer luz



Fuente: Propia

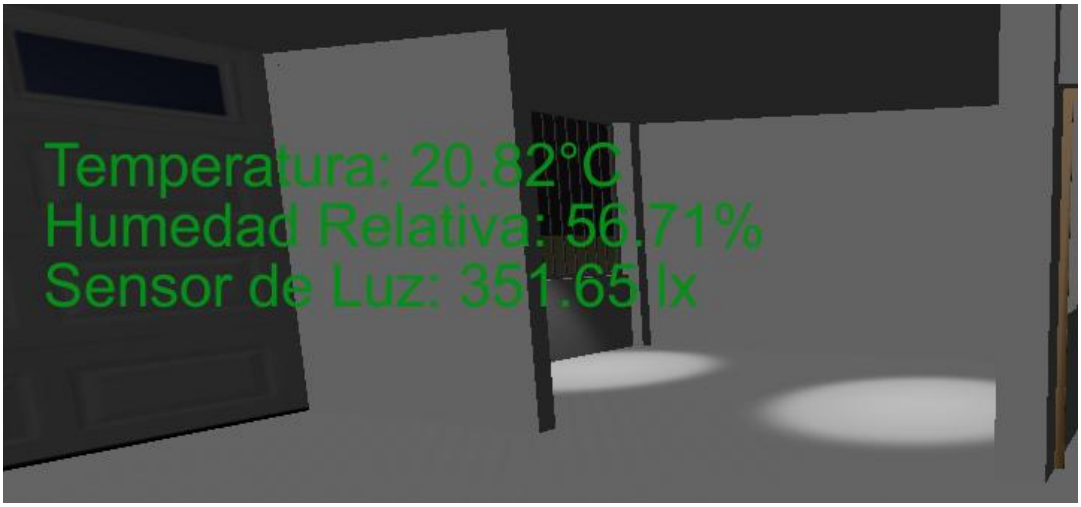
Figura 48: Prueba de control remoto con la primer luz



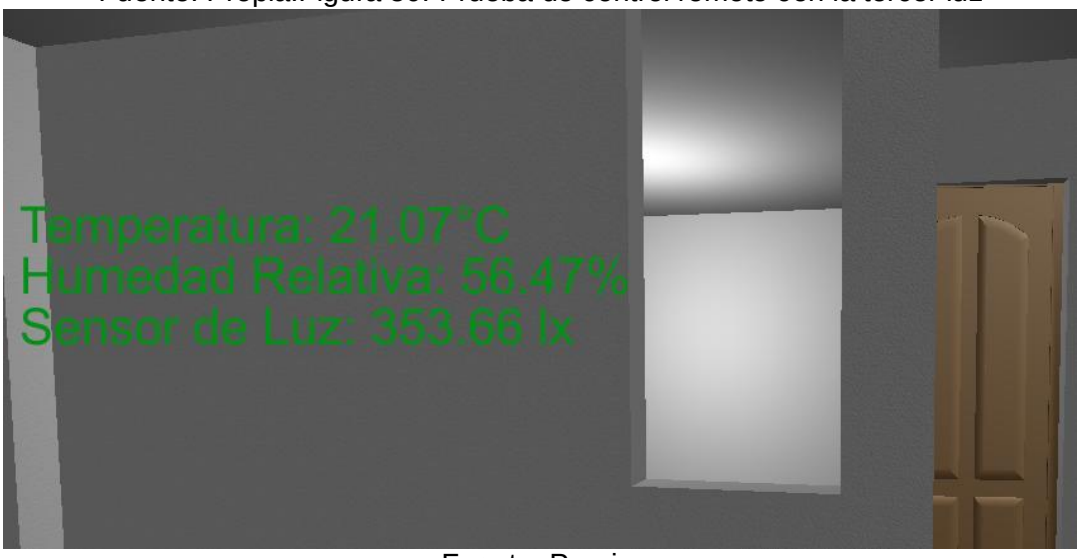
Fuente: Propia.

Para las siguiente pruebas se realizó el mismo contro pero para cada luz individual, de esa forma la figura 49 muestra el simulador cuando unicamente está activado el botón 2. Así la figura 50 muestra al simulador cuando unicamente esta prendido el botón 3, y finalmente la figura 51 muestra lo que sucede cuando unicamente se deja al switch 4 activado

Figura 49: Prueba de control remoto con la segunda luz

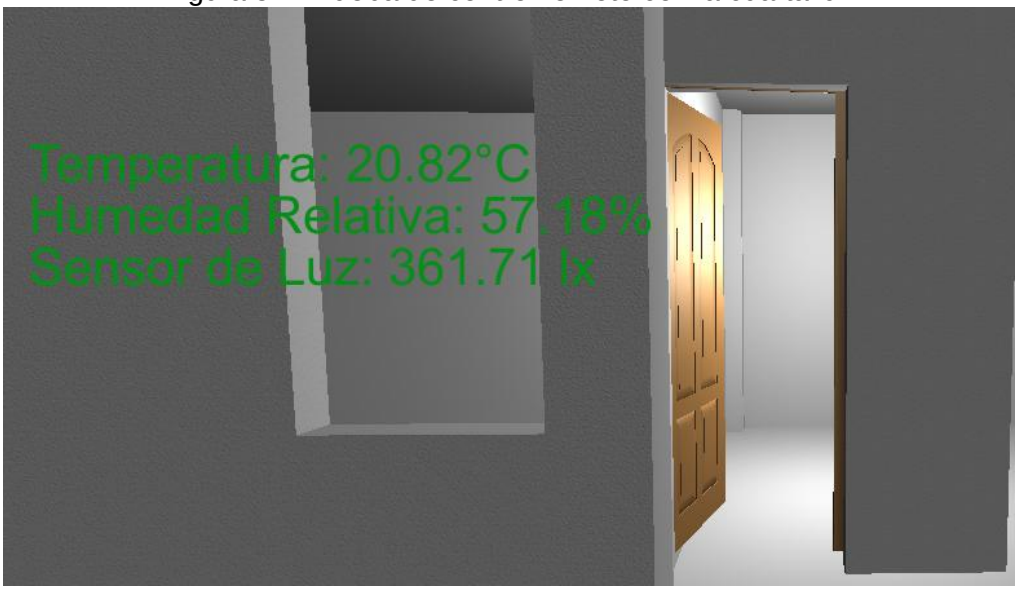


Fuente: Propia.Figura 50: Prueba de control remoto con la tercer luz



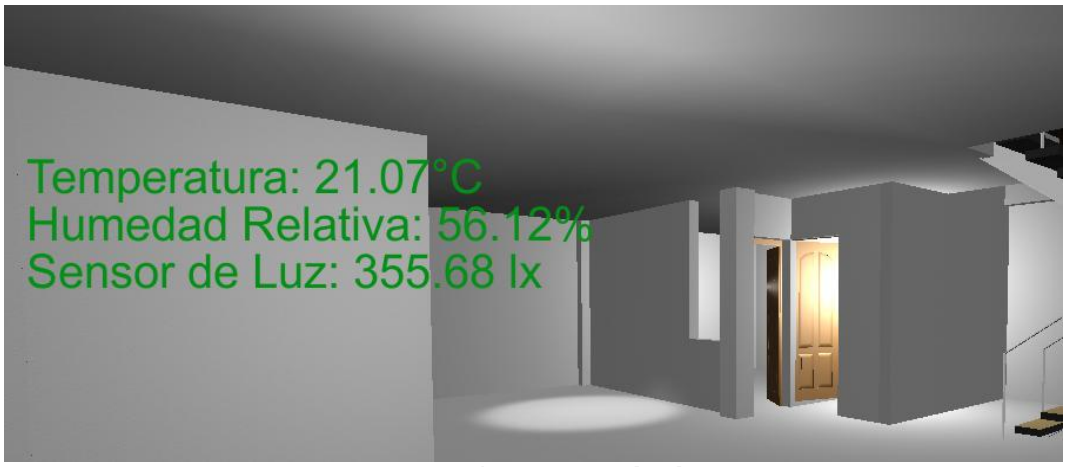
Fuente: Propia.

Figura 51: Prueba de control remoto con la cuarta luz



Fuente: Propia.

Como una última prueba se quiso activar todos los componentes que puede manejar el simulador, de esta forma las 4 luces están activas y las dos puertas abiertas, así como también los sensores están leyendo los valores actuales. Tal como muestra la figura 52.Figura 52: Prueba de control remoto con todos los botones accionados



Fuente: Propia.**4.3 Conclusiones**

El simulador cumplió con los objetivos puestos, es un simulador de bajo costo realizado con dispositivos alcanzables por un estudiante de ingeniería, tanto por precio como por la facilidad de uso. Este tiene variables que toma del mundo real, y el control remoto de unas de sus variables es realizado por un celular con Bluetooth®.

Se crearon el .exe del simulador y el apk de la aplicación en Android, dando muestra de los alcances de un simulador con bajo presupuesto. Por la parte de Android Studio es gratuita la aplicación y todo lo que se puede desarrollar con ella, pero por Unity3D existe la gratuidad para desarrollar pero no para distribuir. Esa es una de las razones por las cuales se eligió este simulador de videojuegos. Siendo una versión gratuita del programa, las características con las que el simulador puede llegar visualmente en una versión paga son mayores.

Se usaron sensores de bajo costo y cumplieron con la función que debían, el usar sensores de protocolo implicaría un mayor costo de inversión. Además para poder identificarlos no se presentaron mayores problemas, puesto que encontrar el punto de calibración o sistemas homólogos en un ambiente universitario no es complejo.

Crear un entorno virtual donde se diseñen elementos automáticos solo es cuestión de experimentación, es decir se pueden aplicar conceptos que en el mundo real serían complejos o costosos, además de dar la posibilidad de poder crear un espacio único.

Las posibilidades de desarrollo de un sistema de este tipo son muy grandes, ya el manejo de la información desde Unity3D es el que da las posibilidades de aplicación, por lo tanto si se desea mejorar el sistema se debe tener muy en cuenta eso, ya que es el corazón del simulador.

Resumiendo el simulador cumplió con las expectativas, es funcional y se puede repetir este proceso para un simulador con características similares, pero como se ha demostrado aun se pueden mejorar todos los componentes del mismo.

4.4 Trabajos futuros

Este simulador propone una nueva forma de poder abordar este tipo de problemas, y como se observo en las conclusiones se puede trabajar en muchos aspectos del mismo.

El primer trabajo futuro que se debe realizar es agregar actuadores, de está forma ya se pueden implementar controladores físicos, y no solo manejo de variables

dentro de la simulación. Esto aumenta las posibilidades de lo que un ingeniero mecatrónico puede experimentar con el entorno.

Se propone es realizar un control de variables unificado, es decir que la unidad que monitoreé los sensores, sea capaz de manejar la información que se obtiene desde el dispositivo Android, y que también pueda manejar algunos actuadores, en otras palabras crear un banco de sensores y actuadores propio de este simulador con el que se puedan experimentar teorías domóticas como el uso de escenas o cualquier cosa que se le ocurra al usuario.

Otro trabajo sería crear bloques para Unity3D que sean capaces de crear un entorno domótico fácilmente, y conocimientos mínimos sobre Unity3D. Esta idea se ve sintetizada en crear librerías de uso, e.g. una librería de luces que ya puedan comunicarse con la base que la controla.

Como lo más usado en domótica es el protocolo IP, es un trabajo futuro de gran interés para el sistema domótico. La idea es crear sockets de comunicación IP en los 3 dispositivos para crear una interfaz de intercambio de datos estable, optima y con retroalimentación. Además que al ser un protocolo basado en IP se puede usar inalambricamente también desde cualquier dispositivos con acceso a WIFI o cableado Ethernet mediante un Router.

Como otro posible trabajo futuro y no menos importante, es crear más prácticas. Crear situaciones donde el estudiante se vea más envuelto en la teoría de la domótica y sus aplicaciones.

BIBLIOGRAFÍA

[1] The Little Guys, Recuperado en Septiembre, 2014, “Living Room Automation Simulator :The Little Guys ”, , Disponible en: http://www.thelittleguys.com/living-room/

[2] Domótica Viva S.L. , Recuperado en Septiembre, 2014, “Demo interactiva EIB/KNX ”, , Disponible en: http://www.domoticaviva.com/demo/demo_knx.htm

[3] Gonzáles, J. , Recuperado en Septiembre, 2014, “Aplicación web para la generación y simulación de eventos domóticos”, , Disponible en: https://riunet.upv.es/bitstream/handle/10251/14238/memoria.pdf?sequence=1

[4]Conte, G.; Scaradozzi, D.; Donnini, R.; Pedale, A., "Building simulation/emulation environments for home automation systems," *Control & Automation (MED), 2011 19th Mediterranean Conference on* , vol., no., pp.31,38, 20-23 June 2011; doi: 10.1109/MED.2011.5983093; URL: http://ieeexplore.ieee.org.ezproxy.unal.edu.co/stamp/stamp.jsp?tp=&arnumber=5983093&isnumber=5982972

[5] Jun-Ming Su, Chih-Fang Huang, An easy-to-use 3D visualization system for planning context-aware applications in smart buildings, Computer Standards & Interfaces, Volume 36, Issue 2, February 2014, Pages 312-326, ISSN 0920-5489, http://dx.doi.org/10.1016/j.csi.2012.07.004. URL: http://www.sciencedirect.com/science/article/pii/S0920548912000931

[6] REAL GAMES, Recuperado en Septiembre, 2014, “HOME I/O”, Disponible en: http://www.realgames.pt/home-io/

[7]Ming Sun; Jingye Wang, "Methodology of simulation science and technology," System Simulation and Scientific Computing, 2008. ICSC 2008. Asia Simulation Conference - 7th International Conference on , vol., no., pp.989,994, 10-12 Oct. 2008, URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4675509&isnumber=4675314

[8] Smith, R.D., 1998, “Simulation Article ”, Disponible en: http://www.modelbenders.com/encyclopedia/encyclopedia.html

[9] Vallverdúl,J. , “What are Simulations? An Epistemological Approach”, Procedia Technology , Vol. 13, 2014, páginas 6–15, disponible en: http://www.sciencedirect.com/science/article/pii/S2212017314000139.

[10] Kamat, V.R.; Martinez, J.C., "Practical 3D animation of multiply articulated construction equipment,"*Simulation Conference, 2004. Proceedings of the 2004 Winter*, vol.2, páginas1229,1237 vol.2, 5-8 Dec. 2004, Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1371454&isnumber=29990

[11] Fei Yao; Lingguo Cui; Senchun Chai; Guolong Zhang, "Distributed interactive 3D combat simulation system for the army training and testing," Control and Decision Conference (CCDC), 2013 25th Chinese, pp.3143,3147, 25-27 May 2013, Disponible en: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6561486&isnumber=6560882

[12] Jiang Jie; Kuang Yang; Shen Haihui, "Study on the Virtual Natural Landscape Walkthrough by Using Unity 3D," Computational and Information Sciences (ICCIS), 2013 Fifth International Conference on , vol., no., pp.1,4, 21-23 June 2013, Disponible en:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6642924&isnumber=6642914

[13]Jingming Xie, "Research on key technologies base Unity3D game engine," *Computer Science & Education (ICCSE), 2012 7th International Conference on* , vol., no., pp.695,699, 14-17 July 2012 doi:10.1109/ICCSE.2012.6295169,URL: http://ieeexplore.ieee.org.ezproxy.unal.edu.co/stamp/stamp.jsp?tp=&arnumber=6295169&isnumber=6295013

[14] Liyanage C. De Silva, Chamin Morikawa, Iskandar M. Petra, State of the art of smart homes, Engineering Applications of Artificial Intelligence, Volume 25, Issue 7, October 2012, Pages 1313-1321, ISSN 0952-1976, Disponible en: http://www.sciencedirect.com/science/article/pii/S095219761200098X.

[15]Dan Ding, Rory A. Cooper, Paul F. Pasquina, Lavinia Fici-Pasquina, Sensor technology for smart homes, Maturitas, Volume 69, Issue 2, June 2011, Pages 131-136, ISSN 0378-5122,

Disponible en: http://www.sciencedirect.com/science/article/pii/S0378512211000983.

[16]Unity Technologies,Página principal de unity3d,URL: http://unity3d.com/

[17]SIG, Página oficial sobre información del Bluetooth®, URL:http://www.bluetooth.org

[18]Google developer, Toda información relacionada con desarrollo de aplicaciones en Android, URL:http://developer.android.com/index.html

[19]**{Jalc}** Developer Live ,Conectar Android y Arduino por Bluetooth Cap.3, 31 de Mayo de 2014 , URL: jalcdeveloper.com/projects/chapters/arduino-android-blue-cap3/

[20]ASOCIACIÓN ESPAÑOLA DE DOMÓTICA E INMÓTICA ,Sobre Domótica, URL:http://www.cedom.es/sobre-domotica/normativa-y-certificacion.

[20]K. Pothuganti A. Chitneni; A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi ; ISSN 2231-1297; Volume 4; Number 6 (2014); pp. 655-662; URL:http://www.ripublication.com/aeee_spl/aeeev4n6spl_18.pdf

[21]Emant Pte Ltd Co, Measure Light Intensity using Light Dependent Resistor (LDR), URL:http://www.emant.com/316002.page.

[22]Arduino, MQ Gas sensors, URL:http://playground.arduino.cc/Main/MQGasSensors

ANEXO A

DATASHEETS

Ambient Light Sensor
Vishay Semiconductors
www.vishay.com

Description

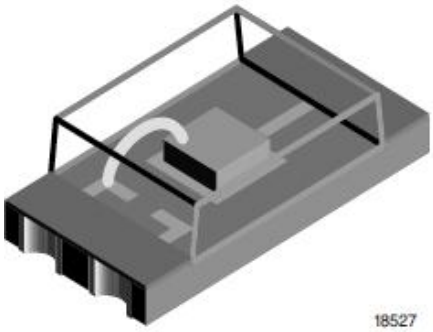
TEMT6000 is a silicon NPN epitaxial planar phototransistor in a miniature transparent mold for surface mounting onto a printed circuit board. The device is sensitive to the visible spectrum.

Features

- Adapted to human eye responsivity
- Wide angle of half sensitivity □ = ± 60°
- SMD style package on PCB technology
- Suitable for IR reflow soldering
- Lead free component
- Component in accordance to RoHS 2002/95/EC and WEEE 2002/96/EC

Applications

Ambient light sensor for display backlight dimming in:
Mobile phones
Notebook computers
PDA's
Cameras
Dashboards



Humidity Sensors
HIH-4030/31 Series

DESCRIPTION

Honeywell has expanded our HIH Series to include an SMD (Surface Mount Device) product line: the new HIH 4030/4031.

The HIH 4030/4031 complements our existing line of non-SMD humidity sensors. SMD packaging on tape and reel allows for use in high volume, automated pick and place manufacturing, eliminating lead misalignment to printed circuit board throughhole.

The HIH-4030/4031 Series Humidity Sensors are designed specifically for high volume OEM (Original Equipment Manufacturer) users.

Direct input to a controller or other device is made possible by this sensor's near linear voltage output. With a typical current draw of only 200 µA, the HIH-4030/4031 Series is often ideally suited for low drain, battery operated systems.

Tight sensor interchangeability reduces or eliminates OEM production calibration costs. Individual sensor calibration data is available.

The HIH-4030/4031 Series delivers instrumentation-quality RH (Relative Humidity) sensing performance in a competitively priced, solderable SMD.

The HIH-4030 is a covered integrated circuit humidity sensor.

The HIH-4031 is a covered, condensation-resistant, integrated circuit humidity sensor that is factory-fitted with a hydrophobic filter allowing it to be used in condensing environments

including industrial, medical and commercial applications.

The RH sensor uses a laser trimmed, thermoset polymer capacitive sensing element with on-chip integrated signal conditioning.

The sensing element's multilayer construction provides excellent resistance to most application hazards such as condensation, dust, dirt, oils and common environmental chemicals.

Sample packs are available. See order guide.

FEATURES

- Tape and reel packaging allows for use in high volume pick and place manufacturing (1,000 units per tape and reel)
- Molded thermoset plastic housing
- Near linear voltage output vs %RH
- Laser trimmed interchangeability
- Low power design
- Enhanced accuracy
- Fast response time
- Stable, low drift performance
- Chemically resistant

POTENTIAL APPLICATIONS

- Refrigeration equipment
- HVAC (Heating, Ventilation and Air Conditioning) equipment
- Medical equipment
- Drying
- Metrology
- Battery-powered systems

MQ-2 GAS SENSOR

HANWEI ELETRONICS CO.,LTD MQ-2 http://www.hwsensor.com
TEL: 86-371- 67169070 67169080 FAX: 86-371-67169090 E-mail: sales@hwsensor.com

TECHNICAL DATA MQ-2 GAS SENSOR

FEATURES

Wide detecting scope Fast response and High sensitivity
Stable and long life Simple drive circuit

APPLICATION

They are used in gas leakage detecting equipments in family and industry, are suitable for detecting
of LPG, i-butane, propane, methane ,alcohol, Hydrogen, smoke.

SENSITVITY ADJUSTMENT

Resistance value of MQ-2 is difference to various kinds and various concentration gases. So,When using
this components, sensitivity adjustment is very necessary. we recommend that you calibrate the detector for
1000ppm liquified petroleum gas<LPG>,or 1000ppm iso-butane<i-C4H10>concentration in air and use value of
Load resistance that(RL) about 20 KΩ(5KΩ to 47 KΩ).
When accurately measuring, the proper alarm point for the gas detector should be determined after
considering the temperature and humidity influence.

LM35 Precision Centigrade Temperature Sensors

FEATURES DESCRIPTION

The LM35 series are precision integrated-circuit
- **Calibrated Directly in ° Celsius (Centigrade)**
- **Linear + 10 mV/°C Scale Factor** temperature sensors, with an output voltage linearly proportional to the Centigrade temperature. Thus the
- **0.5°C Ensured Accuracy (at +25°C)** LM35 has an advantage over linear temperature
- **Rated for Full −55°C to +150°C Range** sensors calibrated in ° Kelvin, as the user is not
- **Suitable for Remote Applications** required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The
- **Low Cost Due to Wafer-Level Trimming** LM35 does not require any external calibration or
- **Operates from 4 to 30 V** trimming to provide typical accuracies of ±¼°C at
- **Less than 60-µA Current Drain** room temperature and ±¾°C over a full −55°C to +150°C temperature range. Low cost is assured by
- **Low Self-Heating, 0.08°C in Still Air** trimming and calibration at the wafer level. The low
- **Nonlinearity Only ±¼°C Typical** output impedance, linear output, and precise inherent
- **Low Impedance Output, 0.1 Ω for 1 mA Load** calibration of the LM35 make interfacing to readout or

control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 draws only 60 µA from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 is rated to operate over a −55°C to +150°C temperature range, while the LM35C is rated for a −40°C to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface-mount smalloutline package and a plastic TO-220 package.

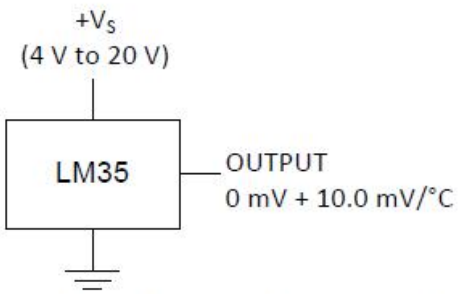


Figure 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)

PIR Motion Sensor
SEN-08630ROHS

Description: This is a simple to use motion sensor. Power it up and wait 1-2 seconds for the sensor to get a snapshot of the still room. If anything moves after that period, the ‘alarm’ pin will go low.

This unit works great from 5 to 12V (datasheet shows 12V). You can also install a jumper wire past the 5V regulator on board to make this unit work at 3.3V. Sensor uses 1.6mA@3.3V.

The alarm pin is an open collector meaning you will need a pull up resistor on the alarm pin. The open drain setup allows multiple motion sensors to be connected on a single input pin. If any of the motion sensors go off, the input pin will be pulled low.

The connector is slightly odd but has a 0.1" pitch female connector making it compatible with jumper wires and 0.1" male headers.

Note: The colors of the wires can vary greatly. Looking at the unit from the top, the middle wire is ground, the left-most wire is the alarm, and the right-most is the power.

ANEXO B

MANUAL DE USUARIO

El sistema de entrenamiento contiene:

- Banco domótico.
- Módulo Bluetooth RN-41 con tarjeta de alimentación para ARDUINO.
- Módulo USB-Serial PL-2303.
- Tarjeta ARDUINO UNO.
- Cables de conexión entre las borneras del banco y la tarjeta de desarrollo.
- Cables bananas para la conexión entre borneras.
- Sensor de gas MQ-2..
- Sensor de presencia PIR-SE-10.
- Sensor de intensidad lumínica TEMT6000.
- Sensor de humedad relativa HIH-4030.
- Sensor de temperatura LM-35.
- Cable Jumper-hembra a banana-macho.

Además de tener los archivos:

- Simulador_domotico.apk
- Carpeta Simulador domótico

Antes de iniciar cualquier actividad con el sistema de entrenamiento instale el archivo .apk a un dispositivo android con versión GingerBread o superior. Instale la versión gratuita de Unity3D y abra el proyecto que se encuentra en la carpeta

Simulador domótico. Una vez abierto el proyecto abra la escena Tocaima.Unity.

Prepare todas las conexiones entre el Banco domótico y los demás dispositivos de forma que:

- Las 6 borneras contiguas al título comunicación sean destinadas para un único puerto ADC de la tarjeta ARDUINO UNO.

- Todas las tierras, excepto la de los módulos de los sensores y actuadores estén acopladas.

- La fuente elegida alimente la línea de Vcc en el banco domótico.

- Los pines Rx-Tx del módulo Bluetooth RN-41 se conecten a las borneras asignadas en el cuadro titulado Bluetooth.

Conecte los pines Rx-Tx del banco al módulo PL-2030 de forma que el Rx de uno se conecte con el Tx del otro.

Desde los Scripts en Unity3D “LecturaSensores.cs” y “LecturaBT.cs” verifique que los puertos COM que usan estos Scripts coincidan con la tarjeta ARDUINO UNO y el módulo Prolific UBS-to-Serial respectivamente.

Uso de la aplicación en Android:

1. Previamente empareje el dispositivo al módulo RN-41
2. Una vez la aplicación abierta la aplicación se ve la pantalla como en la figura 1.
3. Presione la tecla menú de su dispositivo Android y aparecerá la opción “Conectarse a” como muestra la figura 2. Selecciónela.
4. Elija el dispositivo con nombre: “SIMULADOR DOMOTICO” como muestra la figura 3.
5. Espere a que conecte al módulo, este le indicará que existe conexión al prender un LED azul.
6. Ya una vez conectado, chulee la caja ”¿Conectado?”. Para empezar a cambiar las variables cambie los *Switch* y las *Checkbox* y presione Enviar. Si no presiona enviar no se enviarán los cambios al simulador.

Figura 1.

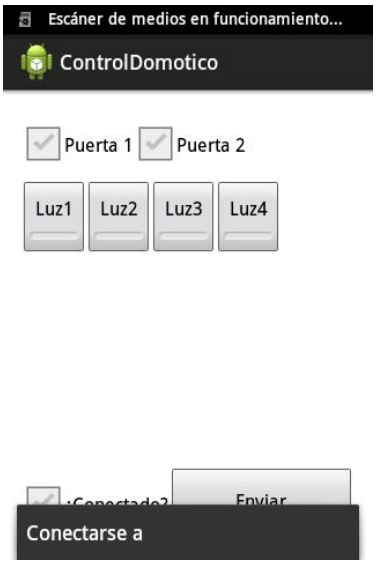


Figura 2.

Figura 3.

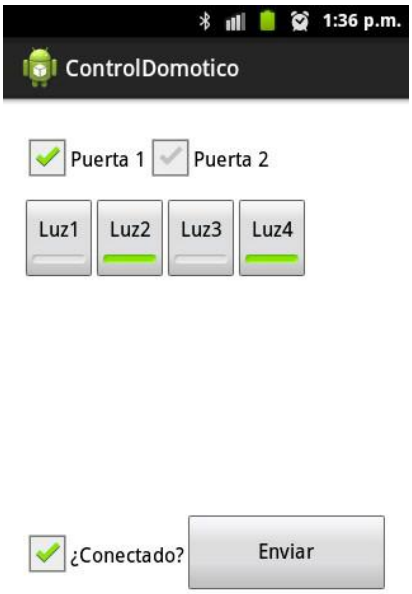


Figura 4.

Recomendaciones y precauciones.

- No inicie la aplicación de Unity3D si el modulo Bluetooth no está conectado al celular.
- Si el módulo Bluetooth está conectado asegúrese que la aplicación tenga la caja “¿Conectado?” seleccionada.
- No use bruscamente las borneras del Banco domótico.
- Use el Banco en una superficie firme.
- Verifique las conexiones antes de asignar cualquier práctica.

ANEXO C

PRÁCTICA 1: SENSORES.

*Para esta práctica no es necesario el uso de la aplicación de Android

Objetivo.

Acercar al estudiante a algunos sensores usados en domótica.

Nota: Antes de usar el Banco domótico asegúrese con su tutor que este este listo para usar.

Procedimiento.

- Vea los sensores que le fueron entregados y por medio de la referencia busque el Datasheet.
- Ubique cada uno de los sensores en las casillas de los sensores, ubicando en orden de izquierda a derecha: Presencia, Gas, Humedad, Intensidad Lumínica y temperatura.
- Con ayuda de los Datasheet conecte cada uno de los sensores realice las conexiones de los sensores a la alimentación y a tierra.
- La salida de los sensores conéctelos en el mismo orden que se mencionó anteriormente a las borneras asignadas al controlador.
- Corra el simulador en Unity3D.

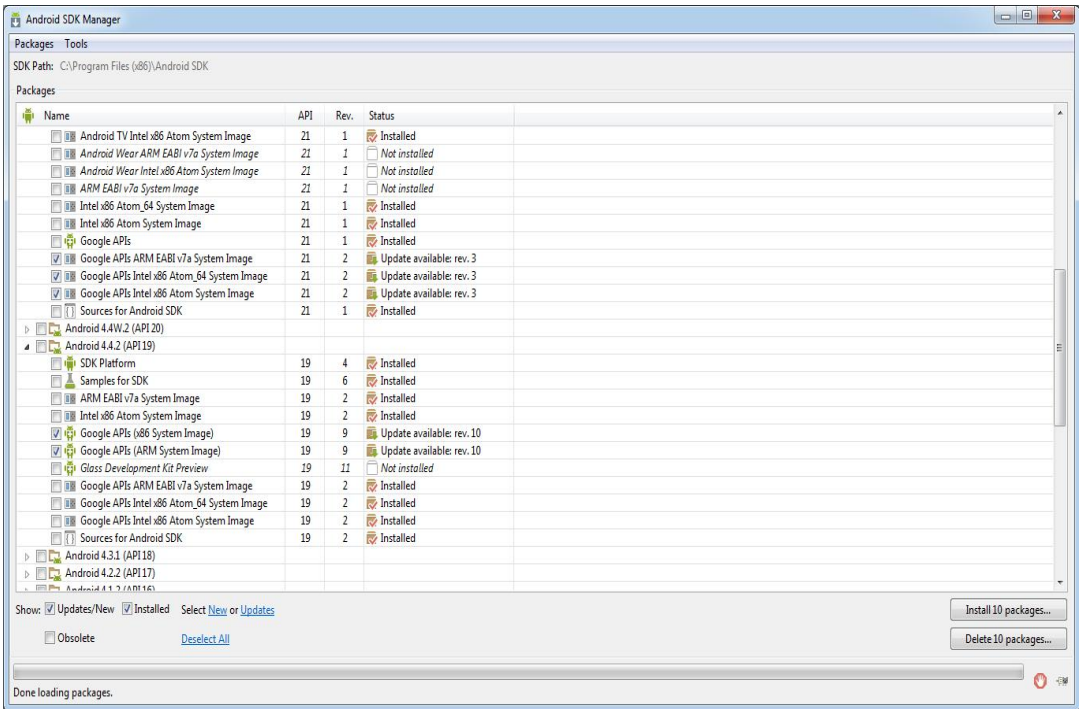
Preguntas

- ¿Qué variable mide cada sensor?
- ¿Es está acertada?
- ¿Para qué se pueden usar estos sensores en domótica u en otros campos?
- ¿Qué otro sensor se puede agregar al sistema?

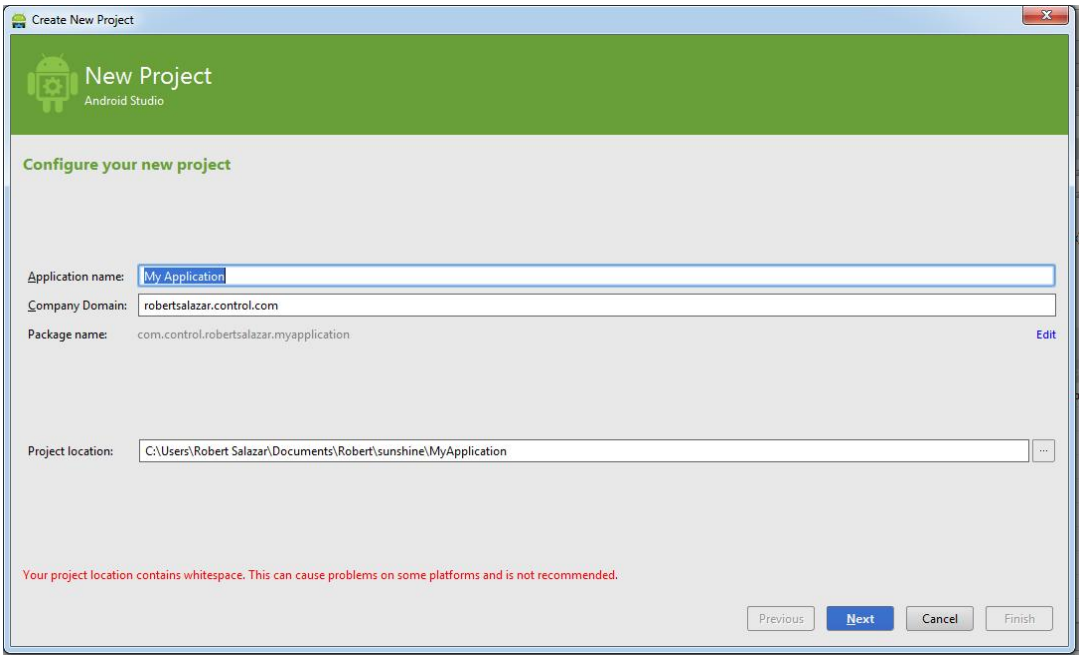
ANEXO D

CREACIÓN DE UN PROYECTO EN ANDROID STUDIO

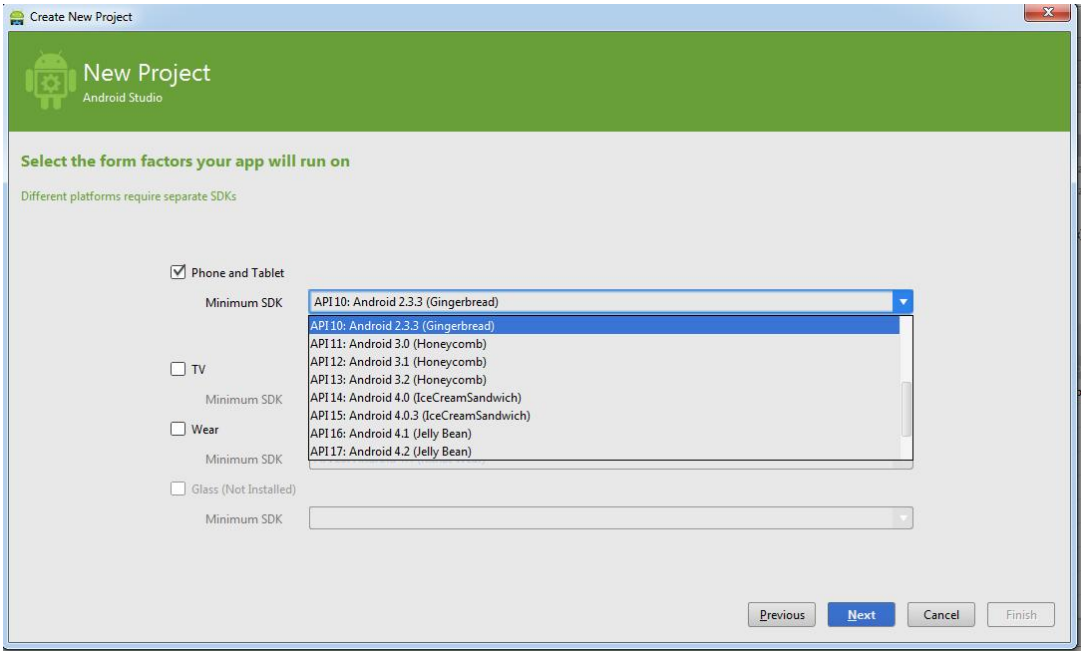
Android Studio esta disponible gratuitamente para cualquier plataforma. Descarguelo e instalelo desde la página oficial. Una vez instalado le recomiendo que abra el Android SDK manager y descargue las herramientas completas del último API. Solo es necesario el ultimo API ya que puede programar para otras plataformas desde este.



Una vez instaladas proceda a crear un nuevo proyecto, se le abrirá una ventana pidiendo el nombre que le desea dar a la aplicación. Se recomienda editar el “Package Name” para que este sea unico en el mundo. La última opción es la ubicación del proyecto, en lo posible cree una ubicación desde C: para que no existan conflictos de nombres con espacios, ya que al compilar algunas cosas esto puede crear problemas.

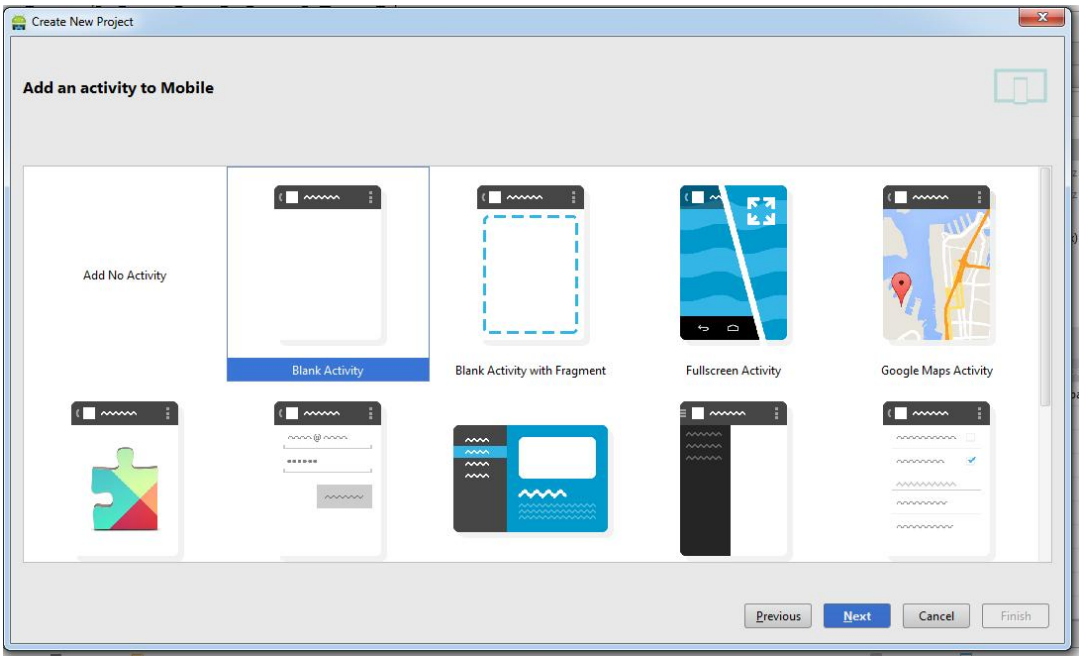


Lo siguiente es elegir la plataforma y API en la cual se desea trabajar. Como se puede apreciar en la imagen las plataformas son: Celular y tablet, Smart Tv, Wear(Reloj), y Google glass. Al elegir la plataforma seguidamente nos pide el API, como se ve en la imagen en el desplegable se elige el API que desea como objetivo, es decir la minima API en la que el celular podra correr.



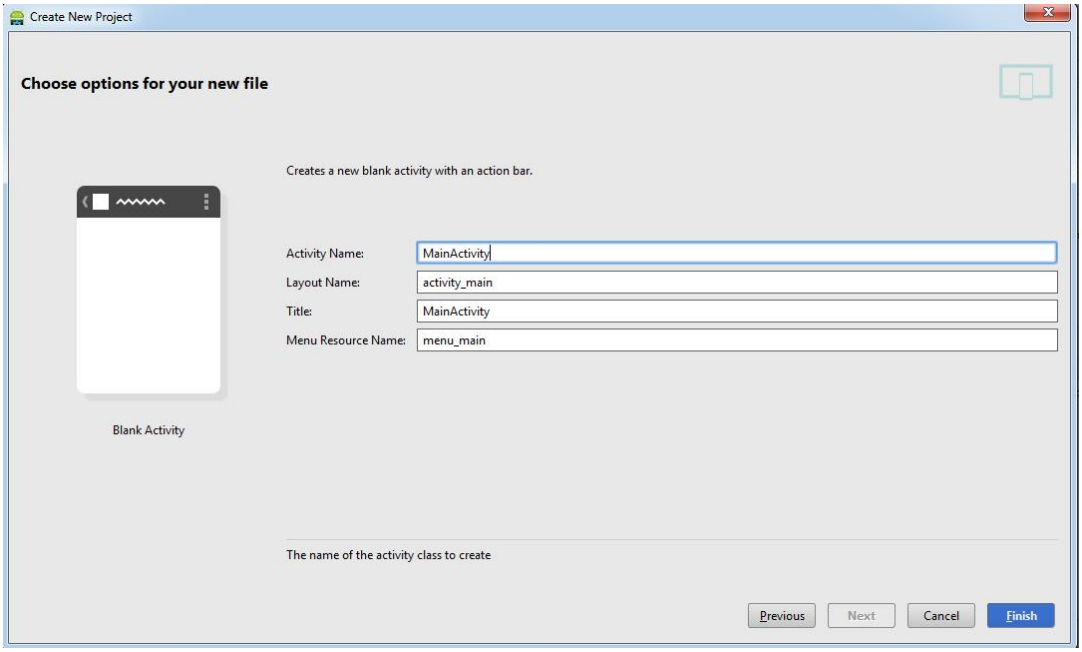
Seguido nos pide la actividad principal de la aplicación, si se desea crear una simple actividad con botones y checkbox se puede crear una “blank activity” o una “Blank activity with fragment”, el “fragment” es simplemente un contendor dentro de la actividad, allí también se pueden colocar los botones y demás. El resto de actividades son para aplicaciones más especificas, se recomienda revisar la documentación de Android developer para mayor información sobre estas.

Cabe mencionar que el resto de tipos de “activity” principales están disponibles desde distintas API, por ejemplo para el API 10 la mayoria de las “activity” mostradas no pueden correr, allí mismo al elegirla nos muestra en rojo que está “activity” no es compatible con la API elegida anteriormente.



La siguiente y ultima ventana que se mostrara es el de “activity”, esta nos pide que nombremos la clase que se creara en JAVA (Activity Name), el nombre del layout usado para la activity(Layout Name), el titulo de la “activity” y el nombre del menu desplegable en esa misma activity.

Así se crea una nueva aplicación en Android Studio, esta se podrá programar con más “activity” y con el tipo de aplicación que desee implementar. Para una vista más amplia del código implementado remitase a la siguiente URL: https://github.com/Robmotor/ControlDomotico esta dirección es un repositorio creado en GitHub para contener todos los códigos aquí mencionados.



Finalmente ya al presionar Finish nos mandará al editor del layout, allí podremos editar el layout a gusto. Para correr la aplicación en nuestro celular presionamos el botón triangular de correr. Allí nos preguntara si deseamos correrlo en el celular o en el simulador integrado, para poderlo correr en el celular es necesario primero ir a configuraciones> aplicaciones>Desarrollo, allí nos aparece un checkbox con la opción de “Depuración de USB” este debe estar activo para poder correrá la aplicación desde Android Studio en un celular real.