Tecnológico de Costa Rica

Ingeniería en Computadores

Introducción a los Sistemas Embebidos

Sistema a la medida para el control y monitoreo de una casa inteligente por medio de servidor web

---

Estudiantes:
Malcolm Davis
David Monestel
Fabian Solano

Profesor:
Ing. Jeferson González

21 de septiembre de 2018

Índice

1. Descripción del Sistema

Este documento contiene el proceso detallado del diseño y desarrollo de una çasa inteligente"que, con el uso de un sistema a la medida implementado en un Raspberry Pi puede ser controlada desde un dispositivo móvil.
Para proponer el prototipo del sistema desarrollo en proyecto se utilizó la metodología modular, se decidió de este modo para así facilitar el diseño individual de cada uno de los componentes o soluciones de subproblemas con menor complejidad que al final se unen para componer el sistema completo [1].
La subdivisión de estos problemas se hace en subsistemas individuales que cumplen con los criterios de desarrollo y pueden ser utilizados para crear el sistema completo, más adelante se explica cada uno de ellos. Para conocer más sobre la implementación de cada modulo, puede referirse al documento de diseño que se entregó junto con este proyecto.
Además se utiliza la arquitectura cliente servidor para poder realizar la comunicación entre el dispositivo móvil del usuario, en este caso un celular inteligente con un sistema operativo Android 5.0+ y el controlador principal de la casa que se encuentra conectado a la red.
Por otra parte, y tomando en cuenta que el sistema operativo que se carga en la placa de desarrollo RaspberryPI es un sistema a la medida, se debe de ejecutar una compilación cruzada desde un computador para poder obtener los binarios que el controlador de la casa va a utilizar. El RaspberryPi funciona tanto como servidor como punto de contacto para los dispositivos que se van a controlar utilizando los pines generales de entrada y salida que la placa provee.
Como se mencionó anteriormente, la Raspberry tiene cargada una imagen de un sistema operativo hecho a la medida utilizando el Proyecto Yocto, con el mismo se logró crear una imagen básica con los componentes del sistema mínimos necesarios para ejecutar la aplicación del servidor. A esta imagen se le agregan varias capas para garantizar el funcionamiento de la aplicación, entre ellos, cabe destacar la capa üserland"que es la que permite capturar imágenes de la cámara del modulo, esta es la capa más compleja que se agregó ya que la implementación de esa funcionalidad es más compleja y se aleja del alcance del proyecto. Pero en general se intentó mantener el sistema lo más simple posible para simular el uso de un dispositivo de bajos recursos.
Los componentes descritos anteriormente se ejemplifican en la figura 1

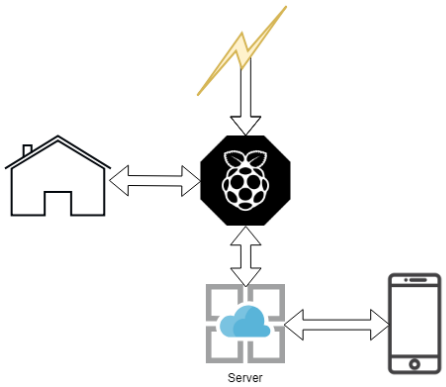

Figura 1: Diagrama del Sistema

1.1. Servidor Web

La implementación del servidor se realizó con el lenguaje de programación C con la biblioteca de sockets del sistema. Así como otras estándares del sistema operativo para poder coordinar los mensajes y generar los tokens aleatorios.
El servidor debe poder autenticar usuarios, para esto se definieron usuarios y contraseñas por defecto que se le brindan a los usuarios para acceder a su cuenta. Una vez el usuario envía las credenciales validas, se le asigna un token que se utilizará para validar las instrucciones enviadas por la aplicación, y así evitar posibles accesos no deseados a funcionalidades de sistema.
Se definió un protocolo de mensajes que utiliza tanto el servidor como los clientes que se conecten al sistema así como usuarios y contraseñas por defecto. En el cuadro 1 se puede observar la definición de los mensajes que puede enviar o recibir el servidor junto con su significado, o la acción que invoca.
El servidor se ejecuta sobre la placa de desarrollo Rapsberry Pi V2 B, aprovechando esto, se puede utilizar los pines de propósito general para controlar las luces, controlar la cámara de fotografías y además controlar las funcionalidades extras agregadas por el equipo como son el accionar de las puertas y la ducha del baño de la casa.
Las señales de tanto las puertas como las luces y la ducha se reciben en el servidor y se envían a los componentes mediante los pines de propósito general que posee la placa de desarrollo, para lograr esta comunicación se implementó una biblioteca que modifica los registros encargados del fun-

Cuadro 1: Protocolo de Mensajes

Comando	Token	Dispositivo	Estado	Acción
1	000	1-4	0/1	Abre o Cierra una puerta
2	000	1-4	0/1	Enciende o Apaga una Luz
3	000	N/A	0/1	Enciende o Apaga la Ducha
4	000	N/A	1	Toma la foto
5	000	N/A	0/1	Login
6	000	N/A	0/1	Apaga todo en la casa

cionamiento de estos pones, esta biblioteca está implementada en C, y se encarga de la definición de los pines utilizados, revisar o modificar el estado y liberarlos en el momento en que se dejan de utilizar para que otra aplicación los pueda usar.

1.2. Aplicación Móvil

La aplicación móvil es el punto de contacto que utiliza el usuario para poder interactuar con los dispositivos de la casa, desde la misma se pueden utilizar las funcionalidades mencionadas anteriormente. La aplicación se implementó desde 0 con Android puro con el programa android studio. Se genera un instalador que puede ser instalado en cualquier dispositivo con android 5.0 en adelante. Para la comunicación con el servidor se utilizaron los sockets de java. Junto con el protocolo de comunicación anteriormente definido. Se pueden observar las pantallas de la aplicación en las figuras 2, 3,4.

2. Biblioteca Utilizada

2.1. Biblioteca GPIO

Para poder utilizar la funcionalidad de entrada y salida general (GPIO) del Raspberry Pi, se implementó una biblioteca en C que accede y modifica a los registros necesarios para interactuar con los pines de entrada y salida para así activar o desactivar dispositivos conectados. Esta modificación de registros en Linux funciona como la escritura o lectura de un archivo [**?**].

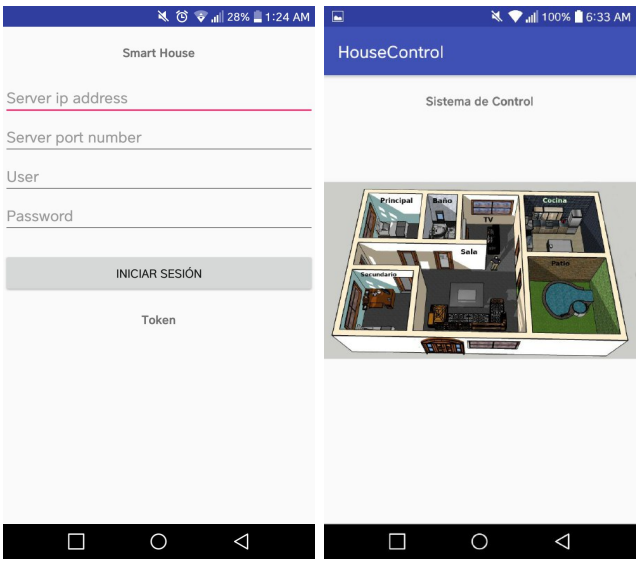

Figura 2: Inicio de Sesión y Pantalla Principal

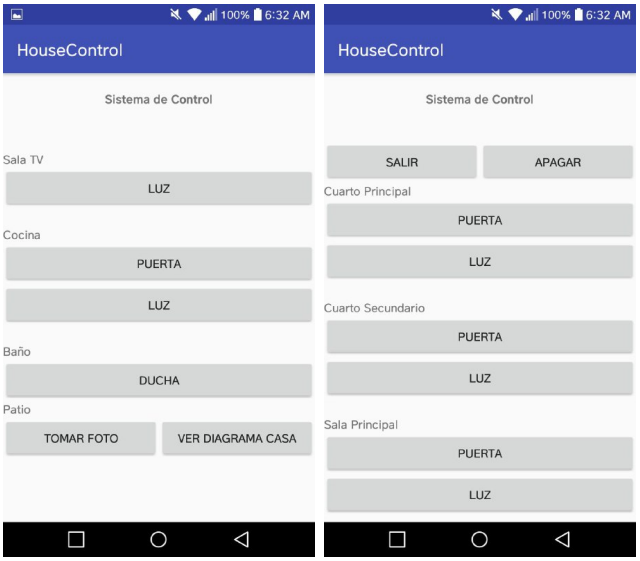

Figura 3: Sensores

Figura 4: Captura de Imágenes

2.2. Descripción de métodos de la Biblioteca GPIO

Acontinuación se enlistan los metodos implementados en la biblioteca:

- **exportGPIO:** Le dice al sistema que se va a utilizar el pin ingresado.
- **unexportGPIO:** Hace lo contrario del anterior, libera el pin.
- **clean:** Libera todos los pines que estén siendo utilizados mediante la biblioteca.
- **pinMode:** Define el tipo de modo en el cual se utilizará el pin.
- **digitalRead:** Se utiliza para conocer el estado de un pin.
- **digitalWrite:** Se utiliza para modificar el estado de un pin.
- **blink:** Método que alterna el estado de un pin.

Y utilizando un diagrama de distribución de pines para la placa de desarrollo se pueden mapear los pines conectados a los dispositivos con el software (ver figura 5).

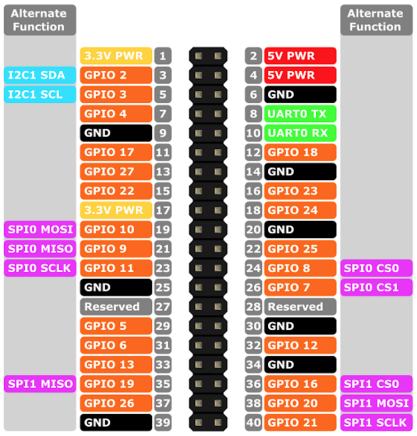

Figura 5: Distribución de pines Raspberry Pi. Obtenida de: https://medium.com/@rxseger/raspberry-pi-3-gpio-pushbuttons-leds-for-rc-and-barr-a1b947dc6b40

3. Maqueta/Dispositivos

La maqueta fue elaborada con una combinación de cortes láser sobre una lámina de MDF y cartón. Se utilizó aproximadamente DATO para la representación de la estructura. Para poder hacer estos cortes en la máquina se requiere un diseño bidimensional de los cortes que debe de hacer la máquina. Este diseño se puede observar en la figura 6. Este diseño se basó tanto del diseño propuesto por el profesor (ver figura 7), como en el modelado que realizó el equipo para planear el diseño de la maqueta(ver figura 8). En la figura 9 se puede observar el diseño de la maqueta. 8).

3.0.1. Requerimientos del sistema

Para este proyecto, los requerimientos que se pueden extraer de la especificación del profesor y además de la rúbrica de evaluación son son los siguientes:

- El sistema debe permitir el control de al menos 5 luces de la casa.
- El sistema debe permitir el monitoreo del estado de al menos 4 puertas de la casa(se implementó también modificar el estado de las mismas como un extra).

Figura 6: Layout de corte

Figura 7: Diseño propuesto por el profesor

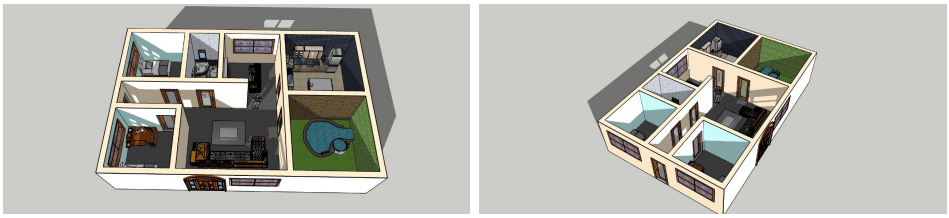

Figura 8: Modelado de la Maqueta

Figura 9: Maqueta Real

- El sistema debe de ser capaz de tomar fotografías del patio y mostrarlas en la aplicación.
- El sistema debe proveer un sistema de autenticación.
- El sistema debe de proveer una interfaz de usuario con el cual pueda interactuar las funcionalidades descritas anteriormente.
- El sistema debe de arrancar apenas se enciende el RaspberryPi sin la intervención del usuario.
- Se debe hacer uso de la compilación cruzada y un toolchain para la misma.
- Se debe de crear una imagen del sistema operativo a la medida para el sistema.
- Se debe proveer una biblioteca para el manejo del GPIO.
- Se debe presentar en físico un modelo que ejemplifique la funcionalidad de la casa.

Figura 10: Baño y Bomba

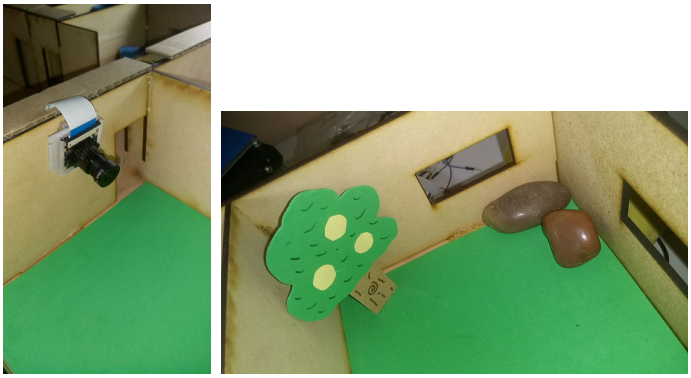

Figura 11: Area de Patio

Cuadro 2: Cumplimiento de los Requerimientos del Proyecto

Requerimiento	Estado
El sistema debe permitir el control de al menos 5 luces de la casa.	Cumple
El sistema debe permitir el monitoreo del estado de al menos 4 puertas de la casa (se implementó también modificar el estado de las mismas como un extra).	Cumple
El sistema debe de ser capaz de tomar fotografías del patio y mostrarlas en la aplicación.	Cumple
El sistema debe proveer un sistema de autenticación.	Cumple
El sistema debe de proveer una interfaz de usuario con el cual pueda interactuar las funcionalidades descritas anteriormente.	Cumple
El sistema debe de arrancar apenas se enciende el RaspberryPi sin la intervención del usuario.	Cumple
Se debe hacer uso de la compilación cruzada y un toolchain para la misma.	Cumple
Se debe de crear una imagen del sistema operativo a la medida para el sistema.	Cumple
Se debe proveer una biblioteca para el manejo del GPIO.	Cumple
Se debe presentar en físico un modelo que ejemplifique la funcionalidad de la casa.	Cumple
El modelo físico debe ser agradable estéticamente.	Cumple

- El modelo físico debe ser agradable estéticamente.

En el cuadro 2 se puede observar el estado de cada uno de estos requerimientos para el proyecto.

4. Metodología de diseño

4.1. Análisis del Problema

El principal objetivo de la computación ubicua es la creación de productos inteligentes conectados que tengan una alta disponibilidad, y que hagan la comunicación más fácil[7]. Estos dispositivos son utilizados para integrar elementos del entorno del ser humano facilitando su quehacer diario.
La domótica es la combinación de los sistemas a la medida, las tecnologías de

la información y la construcción; consiste en utilizar los mismos para crear sistemas de computación ubicua que se integran con las casas logrando optimizar el uso de los recursos y creando facilidades de control remoto de algunos componentes de ellas. Nace del auge de los sistemas a la medida y tendencias como el IoT. Este documento resume el proceso realizado para el diseño y desarrollo de una çasa inteligente". Esto con la finalidad de demostrar la utilidad de estos sistemas mediante el desarrollo de un prototipo de una casa inteligente que, con el uso de un sistema a la medida implementado en un raspberry pi puede ser controlada desde un dispositivo móvil.
Este proyecto está enfocado en desarrollar un sistema de control y monitoreo de una casa inteligente, lo que vendría siendo algo similar a un prototipo de domótica. En la actualidad la domótica está en auge pero se asocia un gran costo a la implementación de un sistema de este tipo, así que uno de los retos de este proyecto es lograr un sistema mínimo funcional que dé otras opciones económicas[2]. El sistema ayuda a optimizar el uso de recursos del hogar evitando el desperdicio dando la posibilidad de modificar el estado de los elementos remotamente. Además, puede verificar el estado de las puertas lo cual agrega seguridad y tranquilidad al usuario si se encuentra fuera de su hogar, otra de las características importantes es el poder tomar fotografías de una de las áreas de la casa.
Tomando en cuenta las características mencionadas anteriormente, el alcance del proyecto y las capacidades de las herramientas utilizadas, además se puede observar que se puede extender la cantidad de sensores, el tipo de sensores y así la funcionalidad del sistema muy fácilmente. Pero el proyecto es más una prueba de concepto para demostrar estas funcionalidades.
Para desarrollar el prototipo del sistema desarrollado en proyecto se utilizó la metodología modular, se decidió de este modo para así facilitar el diseño individual de cada uno de los componentes o soluciones de subproblemas con menor complejidad que al final se unen para componer el sistema completo [1]. La subdivisión de estos problemas se hace en subsistemas individuales que cumplen con los criterios de desarrollo y pueden ser utilizados para crear el sistema completo, más adelante se explica cada uno de ellos.

4.2. Investigación

La investigación se enfoca más que todo en los elementos de la creación de un sistema operativo a la medida, la compilación cruzada y el uso de un toolchain para poder desarrollar software desde una computadora para un

sistema embebido. Esto porque estos son los temas introducidos en el curso que aportan a este proyecto en proyecto. Además de la importancia de tener un sistema a la medida para no desperdiciar recursos de procesamiento o energía en tareas indeseadas[3, 5, 4].
Además, fuera de los temas vistos en el curso, se tuvo que investigar sobre el uso de los pines GPIO de la tarjeta de desarrollo para poder crear la biblioteca que hace el senso y actualiza el estado de los pines.

4.3. Propuestas de Diseño

Para el caso del servidor, se tenían dos propuestas una consistía en utilizar una arquitectura cliente serviror con sockets con el lenguaje C para una integración sencilla con la biblioteca elaborada; la otra propuesta era hacer un cliente servidor un poco más complejo con mqtt y conectar una aplicación de ionic+mqtt.
En la parte móvil, se tienen dos propuestas también, utilizar android puro + sockets para el desarrollo de la aplicación lo cual simplifica la conexión con el servidor sencillo pero es un poco más complejo de desarrollar; por el otro lado se tiene la implementación de la interfaz con el framework Ionic y la conexión con mqtt.

4.4. Comparación de las propuestas

Tomando en cuenta que se quiere utilizar la menor cantidad de bibliotecas externas para utilizar el menor espacio disponible se elije la primera opción de servidor, al utilizar los sockets del sistema, no se debe de portar una biblioteca compleja en el sistema a la medida lo cual facilita el proceso. Si se hubiera elegido la otra opción se tiene la desventaja mencionada anteriormente pero además el porteo de la biblioteca se complicó, en los puntos a favor de esta estrategia es que el protocolo de comunicación sería más seguro ya que por si solo mqtt ofrece una capa de seguridad extra y ofrece mayor flexibilidad para las mismas.
Para el caso de la aplicación, ya que se eligió el servidor simple, no hay mucha más opción que hacer la conexión con los sockets de java que provee Android puro. Esta es una ventaja muy grande que tiene la aplicación nativa sobre la de Ionic, pero como se mencionó anteriormente, con Ionic se pueden crear proyectos más atractivos visualmente en menor tiempo y las características

que añade la conexión mqtt también son atractivas; pero la simplicidad es la clave para los proyectos embebidos.

5. Herramientas de Ingeniería

5.1. RaspberryPi

Esta herramienta es la base del proyecto, con ella se controlan todos los elementos del hogar. Además es donde corre el servidor implementado y el sistema operativo hecho a la medida con el Proyecto Yocto.

5.2. ATMEGA328

Este microcontrolador tiene pines PWM que se utilizan para controlar el movimiento de los servos de las puertas.

5.3. Corte Láser

Para la elaboración del modelo físico se utilizó la cortadora láser que el Tecnológico de Costa Rica provee a los estudiantes. Con esta cortadora se elaboraron las divisiones de la casa y luego la misma se armó.

5.4. Yocto

El proyecto Yocto provee una herramienta para la creación de una imagen mínima de GNU/Linux para así poder crear un sistema a la medida. El proyecto Yocto está compuesto de varias herramientas como poky que hacen esto posible.

5.5. Android Studio

Android Studio es un IDE creado por Google para el desarrollo de Android, fue una herramienta de gran ayuda para probar la aplicación.

5.6. Sketchup

Para poder modelar el diseño físico de la casa se utilizó el programa Sketchup de Autodesk que permite crear modelos realistas desde 0 en muy poco tiempo.

Referencias

[1] J. Leiva, *"Diseño de Algoritmos"*, Lcc.uma.es. [Online]. Available: http://www.lcc.uma.es/ jlleivao/algoritmos/t2.pdf. [Accessed: 21- Sep- 2018].

[2] *"How Much Does It Cost To Install A Home Automation System"*, homeadvisor.com. [Online]. Available: https://www.homeadvisor.com/cost/electrical/install-or-repair-a-home-automation-system/. [Accessed: 21- Sep- 2018].

[3] Aguilas, M, 2009. *"Tutorial Linux"*. [Online] Available: TEC Digital

[4] Aguilar, M, 2009. *"Tutorial GCC"*. [Online] Available: TEC Digital

[5] Aguilar, M. 2009. *"Tutorial Make"*. [Online] Available: TEC Digital

[6] González, J. (2018). *"Tutorial Yocto"*. [Online] Available: TEC Digital

[7] *"What is Ubiquitous Computing? - Definition from Techopedia"*, Techopedia.com, 2018. [Online]. Available: https://www.techopedia.com/definition/22702/ubiquitous-computing. [Accessed: 21 - Sep- 2018].