Smart Pet Feeder

Malcolm Davis
Computer Engineering
mdavis.cr@ieee.org

David Monestel
Computer Engineering
p.david06.p@gmail.com

Fabian Solano
Computer Engineering
fasm2296@gmail.com

***Resumen*—En este proyecto se demuestra la utilidad de los sistemas a la medida, y además, se pone en practica los conceptos vistos en clase de diseño de sistemas embebidos. Esto, con el desarrollo de un prototipo de un sistema de alimentación de mascotas inteligente. El prototipo se realiza con un Raspberry pi 2.**

Palabras clave: Alimentador Inteligente, Raspberry Pi, Yocto, Multimedia, Servidor Web, Sistemas Embebidos, TEC.

I. Introducción

El desarrollo de sistemas embebidos se ha popularizado en los últimos años. El poder tener un dispositivo especializado realizando una función que le facilite la vida a las personas es muy común en estos tiempos. Además, como se ha discutido ya anteriormente en el curso, y tomando en cuenta que 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.

Por otro lado, el proyecto Yocto, que es un proyecto de colaborativo open source que provee las herramientas necesarias para crear un sistema basado en Linux a la medida, facilita mucho el proceso de desarrollar prototipos funcionales con las placas de desarrollo más conocidas**??**. Tomando en cuenta lo expuesto anteriormente, y la necesidad que tienen los dueños de mascotas en caso de requerir irse de viaje, salir tarde del trabajo o simplemente tener libertad y seguridad de que su mascota se encontrará alimentada adecuadamente a su debida hora. Con smart pet feeder, no será necesario contratar alguna persona que se encargue de alimentarlos. De esta manera se evita que el control del alimento de estos se vea perturbado por algún descontrol debido a la falta de conocimiento de esta persona a la hora de proveer el alimento necesario y adecuado a su mascota.

II. Sistema Desarrollado

Para proponer 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 [**?**]. 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.

II-A. Modelo Físico

Para el modelo físico del prototipo, es decir, el alimentador de mascotas inteligente con el cual el usuario y su mascota interactuarán, se utilizó la placa desarrollo Raspberry Pi 2. Esta placa provee puertos de entrada y salida programables (GPIO), además cuenta con una serie de núcleos de procesamiento, RAM, ROM y puerto de red, que son los elementos mínimos necesarios para el desarrollo del proyecto. Además se utilizó una placa Arduino UNO para realizar el control de pulsos PWM que permiten controlar el stepper motor para girar el dispensador y poder obtener alimento en raciones definidas por el usuario. Además, se diseño un módulo electrónico simple

utilizando el driver de motores L293D, el cual es simplemente un puente H que permite definir la polarización del motor bipolar que fue utilizado. La comunicación entre el Raspberry Pi y el Arduino se realizó mediante un pin de salida de la raspberry y un pin de entrada en el Arduino UNO. Ahora bien, la diferencia de tensiones de operación en este caso no fue un problema, ya que el arduino lee un valor superior a 1.8V como un uno lógico, y la raspberry entrega 3.3V en sus salidas digitales, por lo que no hay riesgo de tensiones de voltaje diferentes pues las entradas del Arduino operan hasta los 5V.

II-B. Servidor Web

Para poder mantener la disponibilidad de las funciones del sistema de manera remota, se implementó un servidor web en la placa de desarrollo Raspberry Pi 2. Se implementó en el lenguaje de programación C con la biblioteca de sockets del 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. El sistema utiliza un token aleatorio generado por cada sesión para así poder validar las ordenes que le son enviadas, esto evita que usuarios no autorizados puedan acceder al control de la aplicación y del sistema de alimento. El servidor también es el encargado de realizar un system call que permita ejecutar los comandos de operación de bibliotecas externas que son utilizadas, por ejemplo en este caso que se utilizó la biblioteca Alsa, que permite reproducir archivos de audio en múltiples formatos.

II-C. Biblioteca I/O

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.

II-D. Control de Alimentación

El sistema es capaz de calendarizar la distribución del alimento de la mascota. Además posee la opción de suplir alimento en un momento del tiempo con sólo presionar un botón. Cuando el sistema administra la dosis de la comida adecuada en el momento, se reproduce un mensaje de vos previamente grabado para que la mascota se acerque y consuma el producto. Además, se puede enviar este mensaje en cualquier momento.

II-E. 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 basó en la desarollada en proyectos anteriores con Android puro con el programa Android Studio, lo cual permitió ahorrar tiempo de desarollo, si se cambiaba a otras formas de generar la aplicación, por ejemplo con el uso de herramientas como Ionic. 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, que funcionan de manera asíncrona con la interfaz gráfica lo que permite manipular la aplicación de manera fluida para el usuario, sin alterar su funcionamiento.

III. RESULTADOS

Esta sección contiene los resultados obtenidos en los diferentes procesos de desarrollo del proyecto.

Se logró crear un prototipo funcional de un dispensador de alimento para mascotas automático con las herramientas básicas disponibles al momento de la elaboración del proyecto.

Se logró entender el funcionamiento de las interrupciones que se generan en los diferentes dispositivos de entrada y salida, que el procesador debe atender para el funcionamiento correcto de la calendarización de la dispensación de alimento.

Se logró crear una interfaz de usuario sencilla con la cual el mismo puede interactuar con el sistema para configurar

el prototipo, realizar funciones sencillas como dispensar el alimento y hablar con su mascota.

Se logró grabar un audio desde el dispositivo con la aplicación y enviarlo al servidor, aunque el servidor no lo pudo reproducir por problemas de compatibilidad.

Se realizaron pruebas simples de las diferentes propuestas, para observar los tiempos de reacción y poder estimar el consumo de energía y espacio de cada solución, dónde se evidenció que las primeras dos propuestas no son adecuadas debido al tiempo de respuesta y consumo de energía(ver el documento de diseño para conocer más del tema).

IV. Conclusiones

Utilizando la metodología de desarrollo modular y las herramientas que provee el proyecto Yocto y la placa de desarrollo Raspberry Pi se logró desarrollar un prototipo funcional de un dispensador de alimento para mascotas automático, que permite mejorar el cuidado de mascotas cuando los dueños se encuentra fuera de casa. Para el correcto funcionamiento de dispensador por medio de un stepper motor es necesario la utilización de una pieza en forma de cilindro que evita la obstrucción debido a las irregularidades del alimento. Por otra parte, al utilizar el proyecto Yocto se llegó a la conclusión de que es importante tener presente la versión respectiva que se está utilizando, por ejemplo sumo, que es la versión actual, ya que a la hora de realizar la clonación de repositorios para obtener distintos meta, suelen tener branches diferentes para cada una de las versiones del proyecto Yocto. En el ámbito del desarrollo de la aplicación es importante destacar que Android cuenta con un MediaRecorder para facilitar la grabación de audio mediante el micrófono integrado del dispositivo, sin embargo la cantidad de formatos de encodificación son limitidas lo que puede ocasionar problemas de incompatibilidades para su uso con bibliotecas externas.

Referencias

1 *"What is the Yocto Project?"*, yoctoproject.com, 2018. [Online]. Available: https://www.yoctoproject.org/ [Accessed: 3 - Nov- 2018].

2 *.Android Developers Supported media formats"*, Android Studio [Online]. Available: https://developer.android.com/guide/topics/media/media-formats. [Accessed: 12- Nov -2018].

3 *Raspberrypi Open Embedded Meta"*, GitHub. [Online]. Available: https://github.com/openembedded/meta-openembedded. [Accessed: 1- Nov- 2018].

4 *raspberrypi/userland"*, GitHub. [Online]. Available: https://github.com/raspberrypi/userland. [Accessed: 21- Sep- 2018]. 2008.