TEC | Tecnológico de Costa Rica

Área Académica de Ingeniería en Computadores

Curso: CE-5303 Introducción a los sistemas embebidos

Proyecto 2: Metodología de diseño de sistema

Profesor: M. Luis Chavarría Zamora

Integrantes:

Calderón Yock Sebastián - 2018161630

Acevedo Rodríguez Kevin - 2018148661

Camacho Hernández Julián - 2019201459

Cartago 2023

Resumen

Para el proceso de diseño de la solución del proyecto se siguió una metodología de tipo cascada, debido a la naturaleza invariante en el tiempo de los requerimientos. Las principales etapas de este proceso fueron: recolección de requerimientos, análisis del estado de la cuestión y herramientas disponibles, diseño, desarrollo y validación. Debido a la naturaleza del documento, se excluye la discusión de las últimas dos etapas: desarrollo y validación.

Recolección de requerimientos

Como primer tarea para el diseño de la solución, se realizó la recolección de requerimientos con base en el documento de especificación del presente proyecto. Este documento especifica una serie de requerimientos funcionales y no funcionales, para los distintos elementos fundamentales que componen el proyecto. Los requerimientos funcionales identificados se encuentran registrados en la Tabla 1 y 2.

Tabla 1. Requisitos de integración con la Raspberry Pi

Código	**Requerimiento**
RRP-01	El sistema debe ser Bare-Metal, no debe usar un OS.
RRP-02	El sistema debe usar Uboot como bootloader.
RRP-03	El sistema debe leer un sensor, de escogencia propia, en todo su rango de tensión.
RRP-04	El sistema debe contar con algún tipo de interfaz para interpretar los datos.
RRP-05	El sistema debe usar el **timer** (no usar NOPs) para el manejo temporal.
RRP-06	La comunicación de dispositivos del sistema debe ser serial (UART, SPI, I2C, etc)
RRP-07	No se pueden conectar periféricos para iniciar el sistema.

Tabla 2. Requisitos de la aplicación

Código	**Requerimiento**
RAPP-01	Debe implementarse en Rust o C y representar la interfaz por medio de LEDs o en C++, pero representar la interfaz gráfica en una pantalla.

Tabla 3. Requisitos no funcionales

Código	Requerimiento
RNF-01	Debe utilizarse la plataforma EVM Raspberrypi 2 o 3.
RNF-02	El sistema debe integrarse con otro sistema empotrado que tenga ADC.
RNF-03	El sistema debe ser a la medida (recursos limitados a la aplicación).
RNF-04	La interfaz de usuario debe ser intuitiva y fácil de entender por el usuario.

En la Tabla 3 se encuentran los requerimientos no funcionales que fueron extraídos del documento.
Estos requerimientos no funcionales no son directamente vinculantes para el funcionamiento del sistema, pero aseguran atributos de la calidad y portabilidad del sistema.

Análisis del estado de la cuestión y herramientas disponibles

Con base en los requerimientos identificados en la etapa de recolección, se identificaron los componentes fundamentales del sistema. Adicionalmente, con los componentes definidos y en concordancia con los lineamientos estipulados en la especificación del proyecto, se realizó la investigación de las herramientas disponibles para el desarrollo. Los componentes, junto con las herramientas correspondientes exploradas se listan a continuación:

- Uboot como bootloader: Universal Bootloader o Uboot es un cargador de arranque universal de código abierto. Esta herramienta está diseñada para ser compatible con diversas arquitecturas y plataformas. Es flexible, adaptable, provee una CLI y ofrece numerosas opciones de configuración.
- Raspberry Pi
    - Generación 2: esta generación de la Raspberry Pi es, respecto a la generación 3, ligeramente menos potente. El procesador es más lento y no tiene conectividad inalámbrica. Por otro lado, consume menos energía que la generación 3, debido a que no cuenta con varias funcionalidades de hardware que sí están presentes en la generación 3.
    - Generación 3: contrario a la generación 2, esta gen es más potente, su procesador es más rápido, cuenta con conexión Wi-Fi y Bluetooth.

Análogamente, consume más energía y, en algunos casos, puede requerir de mecanismos de enfriamiento externos.

- Aplicación
    - C: este lenguaje se caracteriza por su simpleza y grado de control sobre los recursos, además de su portabilidad. Estas características lo hacen un muy buen candidato para el desarrollo de una aplicación para un sistema embebido.
    - C++: este lenguaje es de más alto nivel, en comparación con C y Rust. Esto implica que el consumo de recursos probablemente sea mayor. En contraposición, las ventajas que ofrece al desarrollador son considerables, como orientación a objetos y templates.
    - Rust: la mayor ventaja que supone Rust es la seguridad en términos de acceso y manejo de memoria. Esto es ventajoso en sistemas embebidos, principalmente si son de uso crítico. Además, para concurrencia y paralelismo es bastante eficiente.
- Sensores
    - Ultrasónico: el sensor ultrasónico es un sensor relativamente simple, donde se acciona el trigger del sensor y se lee el pin de echo del mismo. Estos sensores son confiables y fáciles de implementar, además de que tienen un bajo costo monetario.
    - Temperatura: similar a los sensores ultrasónicos, los sensores de temperatura son confiables, precisos, de fácil integración y de bajo costo.
    - Acelerómetro: estos sensores son precisos, pero son sensibles a vibraciones y cambios en general. A diferencia de los otros sensores, este suele requerir de calibración para normalizar valores y hacer las lecturas más exactas.

Diseño de la solución:

Durante esta etapa, se seleccionaron las herramientas que formarían parte del desarrollo de esta solución. Debido a la naturaleza del proyecto, no había mayor margen de diseño para proponer más de una alternativa de solución, ya que la mayoría de herramientas eran de uso obligatorio y las que no lo eran, como el lenguaje de programación de la aplicación, no suponían un cambio mayor en el diseño de la solución.

Los componentes y las herramientas seleccionadas, junto con su justificación, se describen en la Tabla 4.

Tabla 4. Herramientas seleccionadas

Componente	Herramienta seleccionada	Justificación
bootloader	Uboot	El uso de Uboot fue solicitado explícitamente en el documento de especificación del proyecto.
Raspberry Pi	Generación 3	Debido a que el equipo de desarrollo no contaba con ninguna de las dos versiones permitidas de Raspberry Pi (2 o 3), está debió ser provista por el profesor del curso. Debido a disponibilidad, solo fue posible utilizar la generación 3
Aplicación	C++	Inicialmente se seleccionó Rust como lenguaje para la implementación de la aplicación, debido a que la experiencia de trabajar con este lenguaje era un beneficio considerable. Lamentablemente, durante el desarrollo del proyecto, se experimentaron diversos inconvenientes a la hora de tratar de utilizar el lenguaje. Con base en el tiempo de desarrollo perdido, se decidió optar por C++ para agilizar el proceso de desarrollo.
Sensor	Ultrasónico	Todos los sensores explorados constituían una opción factible, pero se escogió el ultrasónico debido a su simplicidad y bajo costo; además, los demás sensores se encontraban agotados o formaban parte de sensores más complejos y, por consiguiente, más costosos.

La propuesta del equipo de desarrollo se presenta en la Figura 1, que corresponde a un diagrama de componentes del sistema. En este diagrama se muestran los componentes como cajas con la etiqueta <<component>>, las relaciones y conexiones entre ellos como líneas y su multiplicidad como números al inicio de las conexiones.

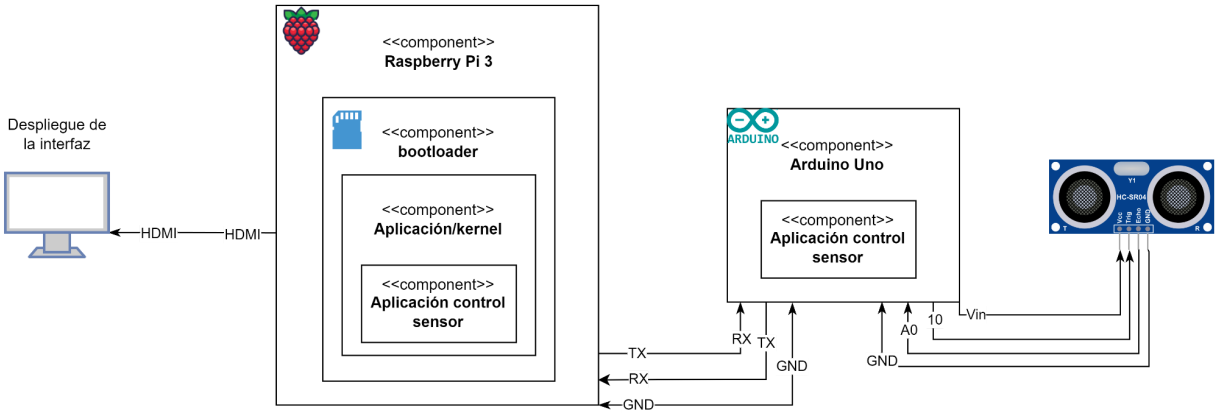

Figura 1. Diagrama de componentes