Sistema de Control de Persianas (ESP32 + Adafruit IO)

Diagrama y Especificación de Casos de Uso

Integrantes:
María Daniela Jiménez
Katherine Cardona
Brandon Bueno
Juan José Osorio

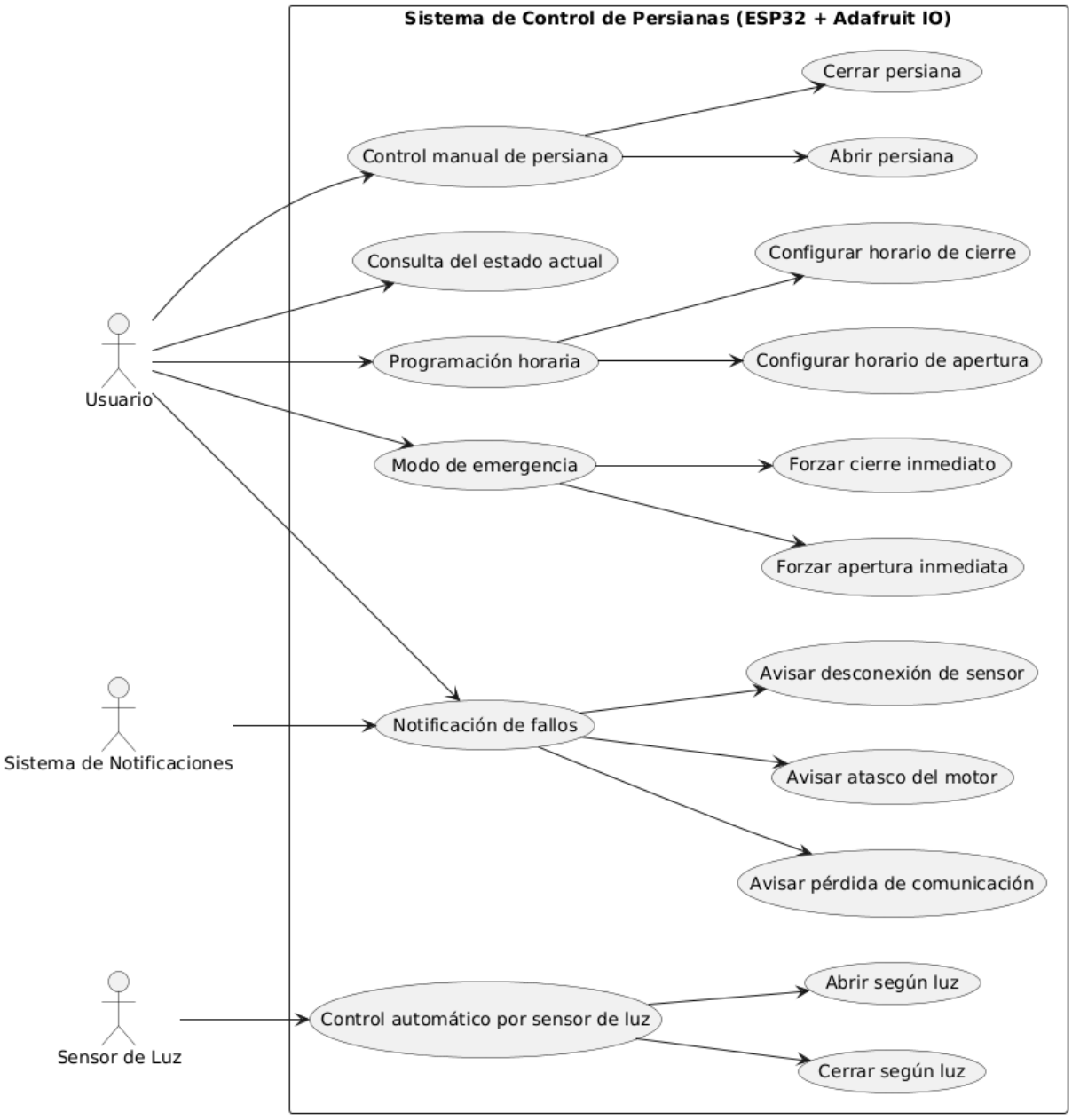
Universidad del Valle
Asignatura: Metodologías de Desarrollo de Software
2025

Introducción

El presente documento describe el análisis de los casos de uso del proyecto de control de persianas implementado con ESP32 y la plataforma Adafruit IO. El objetivo principal del sistema es ofrecer a los usuarios un control manual y automático de las persianas, incorporando además funciones de seguridad, programación horaria y notificación de fallos.

Diagrama de Casos de Uso

En la siguiente figura se presenta el diagrama de casos de uso correspondiente al sistema de control de persianas.



Especificación de Casos de Uso

CU1. Control manual de persiana

Actor principal: Usuario

Objetivo: Permitir que el usuario abra o cierre la persiana manualmente desde la aplicación.

Precondiciones: El sistema debe estar encendido y conectado.

Flujo principal:

1. El usuario abre la aplicación.
2. Selecciona la opción “Abrir persiana” o “Cerrar persiana”.
3. El sistema envía la orden al ESP32 mediante Adafruit IO.
4. El motor ejecuta la acción solicitada.
5. El estado actualizado se refleja en la app.

Flujos alternativos:

1. Si ya está abierta y se solicita apertura → se muestra mensaje “Persiana ya abierta”.
2. Si ya está cerrada y se solicita cierre → se muestra mensaje “Persiana ya cerrada”.
3. Si ocurre un fallo de comunicación → se envía notificación de error.

Postcondiciones: La persiana queda en la posición solicitada (abierta o cerrada).

CU2. Consulta del estado actual

Actor principal: Usuario

Objetivo: Ver el estado actual de la persiana (abierta, cerrada o en movimiento).

Precondiciones: El sistema debe estar en funcionamiento y conectado.

Flujo principal:

1. El usuario solicita el estado actual.
2. El ESP32 envía la información del estado a Adafruit IO.
3. La aplicación cliente actualiza el estado en pantalla.

Flujos alternativos:

1. Si no hay conexión → la aplicación muestra “Estado desconocido”.

Postcondiciones: El usuario conoce el estado real de la persiana.

CU3. Programación horaria

Actor principal: Usuario

Objetivo: Configurar horarios de apertura y cierre automáticos.

Precondiciones: Sistema encendido, conectado a la red y sincronizado con el reloj.

Flujo principal:

2. El usuario accede a la sección de programación.
3. Configura hora de apertura y hora de cierre.
4. El ESP32 guarda la programación.
5. A la hora establecida, el sistema ejecuta la orden (abrir/cerrar).

Flujos alternativos:

1. Si el horario ingresado no es válido → el sistema lo rechaza y solicita corrección.

Postcondiciones: El sistema actualiza el estado de la persiana automáticamente en los horarios configurados y la programación queda registrada en el sistema.

CU4. Modo de emergencia

Actor principal: Usuario

Objetivo: Permitir apertura o cierre inmediato de la persiana, ignorando programación y automatización.

Precondiciones: El sistema está en funcionamiento.

Flujo principal:

1. Usuario selecciona “Modo de emergencia”.
2. Puede elegir “Forzar apertura inmediata” o “Forzar cierre inmediato”.
3. El ESP32 ejecuta la orden directamente.

Flujos alternativos:

1. Si el motor está atascado → se envía notificación de fallo.

Postcondiciones: La persiana queda en la posición segura según la emergencia.

CU5. Notificación de fallos

Actor principal: Sistema de Notificaciones, Usuario

Objetivo: Informar al usuario de problemas en el sistema.

Precondiciones: El sistema detecta un error.

Flujo principal:

1. El sistema identifica la falla (desconexión, atasco del motor o pérdida de comunicación).
2. Envía la alerta a través de Adafruit IO.
3. La aplicación muestra la notificación al usuario.

Postcondiciones: El usuario está informado oportunamente de la falla y puede actuar.

CU6. Control automático por sensor de luz

Actor principal: Sensor de Luz

Objetivo: Permitir que la persiana se abra o cierre automáticamente según las condiciones de iluminación.

Precondiciones: Sensor conectado y calibrado.

Flujo principal:

1. El sensor mide la intensidad lumínica.
2. Si la luz supera el umbral → el sistema cierra la persiana.
3. Si la luz baja del umbral → el sistema abre la persiana.
4. El ESP32 actualiza el estado en la app, el sistema ejecuta la acción.

Flujos alternativos:

1. Si el sensor falla o se desconecta → se notifica al usuario.

Postcondiciones: La persiana se ajusta automáticamente de acuerdo con la luz ambiental.

Conclusiones

El análisis y diseño de los casos de uso permitió identificar de manera estructurada las principales funcionalidades del sistema de control de persianas. Mediante el diagrama UML se representaron las interacciones entre los actores y el sistema, mientras que la especificación de cada caso de uso detalló los flujos de trabajo, condiciones iniciales y resultados esperados.

Este documento constituye una base sólida para el desarrollo del sistema, sirviendo como guía tanto para la implementación como para la validación de los requerimientos planteados.