UNIVERSIDAD DE TARAPACÁ

FACULTAD DE INGENIERÍA

Departamento de Ingeniería en Computación e Informática

Sistema de monitoreo y control de un acuario

“AquaPI”

Autor(es): Bruno Améstica

Jorge Cáceres

Katalina Oviedo

Cristhian Sánchez

Asignatura: Proyecto II

Profesor(es): Diego Aracena

ARICA, 02 DICIEMBRE 2024

Historial de Cambios

Fecha	Versión	Descripción	Autor(es)
10/09/2024	1.0	Versión preliminar del formato	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
11/09/2024	1.1	Revisión y modificación del plan	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
20/09/2024	1.2	Estimación de costos	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
22/09/2024	1.3	Planificación de la gestión de riesgos	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
30/09/2024	1.4	Revisión final informe I	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
16/10/2024	1.5	Corrección informe I	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
30/10/2024	1.6	Planificación de los procesos técnicos	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
04/11/2024	1.7	Revisión final informe II	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
27/11/2024	1.8	Corrección informe II	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez
02/12/2024	1.9	Revisión final informe III	Bruno Améstica Jorge Cáceres Katalina Oviedo Cristhian Sanchez

Tabla de contenidos

Índice de tablas

Índice de figuras

1. Panorama General

1.1. Resumen del Proyecto

Supervisar un acuario puede ser una tarea complicada debido a la necesidad de mantener controlados diversos factores ambientales como la temperatura, pH, niveles de amoníaco, nitritos, nitratos, oxígeno disuelto, luz y conductividad, todos esenciales para la salud de los organismos acuáticos. Sin una supervisión constante y precisa, pequeñas variaciones en alguno de estos parámetros pueden provocar situaciones peligrosas que, si no se abordan rápidamente, pueden resultar en la muerte de los organismos en el acuario.

La complejidad aumenta al considerar las distintas especies que pueden habitar el acuario, cada una con necesidades específicas y tolerancias variables a los diferentes parámetros del agua. Esto hace que el monitoreo manual no solo sea tedioso y consume mucho tiempo, sino también propenso a errores, lo que puede llevar a decisiones ineficaces y potencialmente perjudiciales.

Para solucionar esta problemática, se propone un sistema IoT llamado "AquaPI", que integra sensores especializados para monitorear en tiempo real las condiciones del acuario. Estos sensores detectan parámetros críticos como la temperatura del agua, niveles de pH y la intensidad de la luz. Todos los datos recopilados se centralizan en una Raspberry Pi, que actúa como servidor y procesa la información.

La solución también incluye una interfaz de usuario accesible desde una aplicación remota, que permite supervisar el acuario en cualquier momento y lugar, notificando al usuario si se detecta alguna alteración en los parámetros críticos del acuario. Además, el sistema puede automatizar acciones correctivas, como activar sistemas de calefacción o aireación, reduciendo la necesidad de intervención manual y minimizando el riesgo de errores humanos.

En resumen, la solución propuesta no solo facilita la supervisión continua y precisa de un acuario, sino que también proporciona un nivel de control más avanzado. Al integrar sensores avanzados con tecnologías IoT y un servidor central, "AquaPI" transforma el mantenimiento de acuarios en una tarea mucho más manejable y eficiente, garantizando un entorno saludable y estable para todos los organismos acuáticos, eliminando los riesgos e inconvenientes de realizar las mediciones de estos datos cruciales de forma manual.

1.1.1. Propósito

El propósito del proyecto es mejorar el cuidado de los organismos acuáticos al facilitar el monitoreo y control automatizado de los parámetros ambientales del acuario, brindando a los usuarios mayor precisión y autonomía en el mantenimiento, reduciendo errores y esfuerzos manuales.

1.1.2. Alcance

El sistema de monitoreo y control para el acuario contará con sensores de temperatura, niveles de pH y luz para la medición de parámetros vitales en el acuario, mediante la conexión de un Raspberry Pi que recibirá los parámetros por parte de los sensores para analizar y comprobar que estos valores se encuentren en el rango deseado dependiendo de los requerimientos del usuario, de no encontrarse los valores en el rango deseado se activarán los accionadores para estabilizar los parámetros necesarios en el acuario.

1.1.3. Objetivo general

Desarrollar un sistema de monitoreo y control de un acuario, que asegure la estabilidad de parámetros ambientales esenciales para la supervivencia de los seres biológicos que residen en él, como peces y plantas, minimizando la necesidad de intervención manual y optimizando los procesos de mantenimiento del acuario.

1.1.4. Objetivos específicos

- Investigar los parámetros óptimos de los factores ambientales del acuario para su correcta supervisión.
- Definir los recursos necesarios para el desarrollo del proyecto como los sensores, accionadores y mediadores electrónicos que comuniquen el hardware.
- Diseñar el modelado del proyecto para su realización.
- Planificar el desarrollo del proyecto para un avance eficiente.
- Implementar los conocimientos necesarios para el desarrollo del sistema.
- Realizar pruebas para asegurar que el monitoreo y control automatizado funcione correctamente.

1.1.5. Suposiciones y restricciones

- **Suposiciones:**
    - Se asume que todos los equipos (filtros, calefactores, bombas de aire) funcionarán correctamente y que habrá un plan de mantenimiento regular.
    - Se asume que el sistema de monitoreo podrá alertar a los usuarios de cambios críticos en tiempo real.
- **Restricciones:**
    - Puede haber limitaciones en el conocimiento del usuario sobre el manejo y el mantenimiento adecuado del sistema de control y monitoreo.
    - El sistema puede ser susceptible a fallos tecnológicos o cortes de energía, lo que puede afectar el monitoreo continuo.

1.1.6. Entregables del Proyecto

Los entregables del proyecto son los siguientes:

- Maqueta del sistema
- Presentación de la maqueta
- Informes del proyecto
- Presentaciones del proyecto
- Redmine UTA (wiki, bitácoras y carta Gantt)
- Poster promocional
- Manual de usuario
- Sistema “AquaPI”

2. Organización del proyecto

2.1. Personal y entidades internas

Para asegurar el cumplimiento de los objetivos del proyecto, cada integrante ha sido asignado a un rol específico con responsabilidades claras. A continuación, se describen los roles de cada miembro del equipo.

- **Jefe de grupo:** es el responsable de liderar, coordinar y supervisar el desarrollo de tareas necesarias en el equipo, tomando decisiones para el cumplimiento de los objetivos planteados en un plazo estipulado utilizando métodos de comunicación efectivos con los integrantes del equipo.
- **Programador:** son los encargados de crear la arquitectura del software, implementar la lógica para la aplicación del usuario y el sistema de control, y realizan pruebas para detectar y solucionar fallos.
- **Diseñador:** diseña presentaciones, diagramas e interfaces para representar información relevante de forma visual, asegurando calidad y atractivo en el material gráfico.
- **Documentador:** se ocupa de mantener al día la documentación del proyecto, registrando la información clave de manera continua.
- **Técnico de Hardware:** se encarga de seleccionar y ensamblar los componentes físicos (sensores y actuadores), realizar pruebas de funcionamiento y resolver problemas relacionados con el hardware.

2.2. Roles y responsabilidades

Los roles que tomarán cada integrante del equipo serán:

Rol	Responsable
Jefe	Cristhian Sánchez
Programador	Bruno Améstica, Jorge Cáceres, Cristhian Sánchez y Katalina Oviedo
Diseñador	Jorge Cáceres
Documentador	Katalina Oviedo
Técnico de Hardware	Bruno Améstica, Jorge Cáceres

Tabla 1: Roles y responsabilidades.

2.3. Mecanismos de comunicación y organización

Para garantizar una buena organización y comunicación en el desarrollo del proyecto, se han establecido las siguientes herramientas y plataformas:

- **Discord:** utilizado para realizar reuniones de trabajo mediante llamadas, permitiendo la coordinación en tiempo real. En estas reuniones se informa sobre el progreso del proyecto y se trabaja en conjunto, además de realizar sesiones de lluvia de ideas para mejorar diferentes aspectos del proyecto.
- **WhatsApp:** se utilizará para coordinar los horarios de las reuniones que se realizan en Discord, además de compartir información relevante, como links, documentos, referencias, entre otros.
- **Redmine:** se utilizará para guardar documentación, guía para el seguimiento de una metodología mediante la carta Gantt, realización de tareas mediante el calendario, presentar mediante la wiki la información general del proyecto.
- **Google Drive:** herramienta destinada a la organización de archivos, documentos, links, videos y referencias relacionadas con el proyecto, facilitando el acceso y la colaboración en tiempo real.

3. Planificación de los procesos de gestión

3.1. Planificación inicial del proyecto

En la parte de hardware, se utilizarán los siguientes productos:

- Arriendo de equipos
- Smartphone
- Raspberry Pi 3
- Sensores
- Accionadores

Mientras que para la parte de software:

- Licencia Microsoft Office
- Visual Studio Code
- Unity Hub
- Blender

3.1.1. Planificación de estimaciones

- **Costos de Hardware**

Producto	Cantidad	Costo por unidad	Costo total
Notebook	4	$50.000 (arriendo por mes)	$800.000 (4 meses)
Raspberry Pi 3	1	$81.508	$81.508
Smartphone	1	$64.990	$64.990
Sensor de temperatura DS18B20	1	$6.904	$6.904
Sensor de pH	1	$20.000	$20.000
Sensor de luz	1	$1.950	$1.950
Cámara raspberry	1	$38.483	$38.483
Pantalla LCD	1	$5.140	$5.140
Adaptador Wi-fi	1	$8.142	$8.142
Project Kit Electrónica	1	$13.590	$13.590

Bomba de agua	1	$5.490	$5.490
Termo calentador	1	$8.490	$8.490
Barra luz led	1	$11.990	$11.990
		Total	**$1.056.397**

Tabla 2: Costos de Hardware.

- **Costos de Software**

Producto	**Costo**	**Costo total**
Visual Studio Code	Gratuito	Gratuito
Unity Hub	Gratuito	Gratuito
Blender	Gratuito	Gratuito
Microsoft 365	$7.990/mes (2-6 personas por licencia)	$31.960 (4 meses)
	Total	**$31.960**

Tabla 3: Costos de Software.

3.1.2. Planificación de Recursos Humanos

Rol	**Cantidad por rol**	**Costo/Hora**	**Horas mensuales totales**	**Costo total**
Jefe de proyecto	1	$9.231 CLP/hora	48	$443.088
Programador	4	$5.231 CLP/hora	48	$1.004.352
Diseñador	1	$3.836 CLP/hora	48	$184.128
Documentador	1	$5.538 CLP/hora	48	$265.824
Técnico de Hardware	2	$3.077 CLP/hora	48	$295.392
			Total por 1 mes	**$2.192.784**
			Total por 4 meses	**$8.771.136**

Tabla 4: Costos de RRHH.

3.1.3. Costos totales

Elemento	Costo
Hardware	$1.056.397
Software	$31.960
Recursos Humanos	$8.771.136
Costo total del proyecto	**$9.859.493**

Tabla 5: Costos totales.

3.2. Distribución de tiempos

3.2.1. Carta Gantt

Con el propósito de planificar y gestionar de manera eficiente los tiempos y actividades, se elaboró una carta Gantt, la cual facilita la visualización y gestión del desarrollo del proyecto AquaPI.

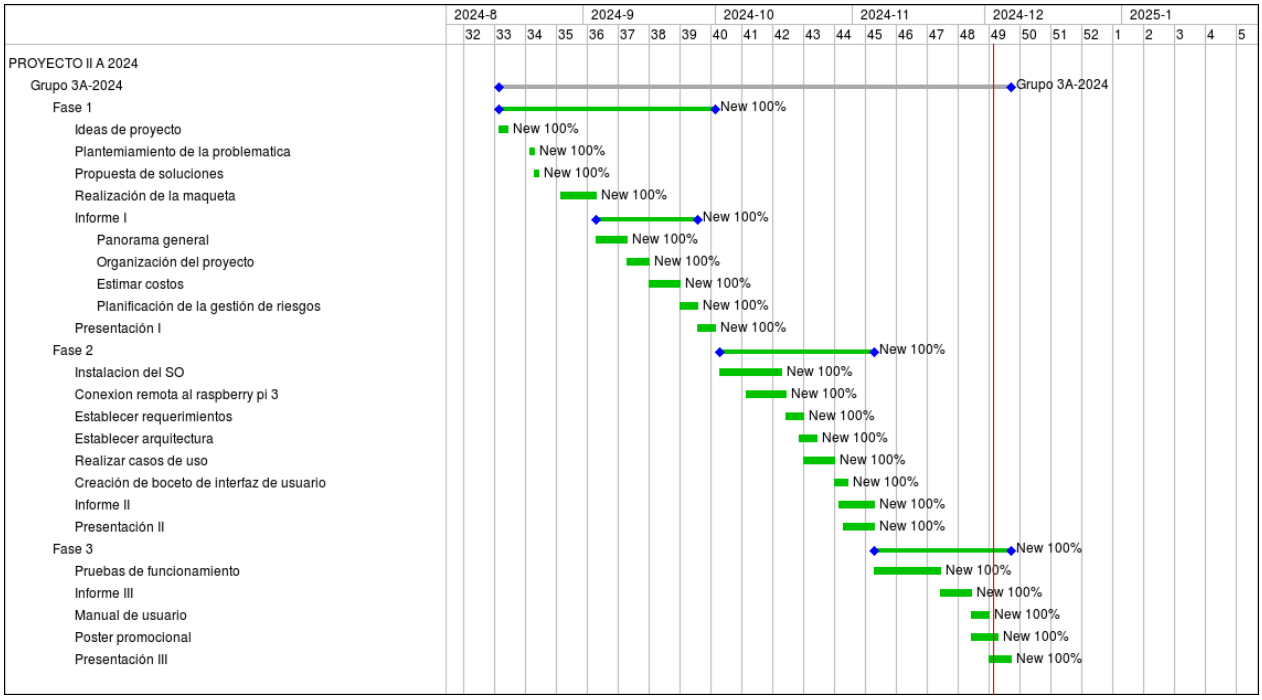


Ilustración 1: Carta Gantt.

3.2.2. Asignación de tiempo

El proyecto está planificado para desarrollarse a lo largo de 16 semanas, con una distribución de trabajo semanal que incluye 6 horas de trabajo en clases y 6 horas de trabajo autónomo, lo que da un total estimado de 192 horas de trabajo. Estas 16 semanas se han dividido en 3 fases, las cuales se encuentran en la siguiente tabla.

FASE	Semanas comprendidas	Fecha
1	Desde la 1º hasta la 6º semana	Del 20 agosto hasta 1 octubre.
2	Desde la 7º a la 12º semana	Del 2 octubre hasta el 5 noviembre.
3	Desde la 13º a las 18º semana	Del 6 noviembre hasta el 5 diciembre.

Tabla 6: Fases del proyecto.

3.3. Planificación de la gestión de riesgos

Se elabora la siguiente tabla para establecer las acciones a tomar ante los posibles riesgos, según su nivel de impacto. Los niveles de impacto son:

1. Catastrófico
2. Crítico
3. Marginal
4. Despreciable

Riesgos	Nivel de impacto	Probabilidad de ocurrencia	Acción remedial
Mala distribución de tareas en el equipo	4	7%	Realizar reuniones de seguimiento para evaluar la comodidad de trabajo de cada integrante del grupo de su área asignada, agilizando las asignaciones de tareas.
Daño del material de hardware utilizado	2	10%	Tener las precauciones necesarias en el uso de los diferentes materiales de

			hardware para evitar situaciones no deseadas.
Poca disponibilidad para las reuniones en equipo	3	25%	Asignar horarios dinámicos para realizar reuniones importantes.
Falta de algún material para el avance del proyecto	2	20%	Evaluar en profundidad los recursos necesarios para el avance de cada una de las etapas a realizar a lo largo del Proyecto mediante una investigación exhaustiva.
Mala administración del tiempo	3	10%	Planificar las tareas a realizar en tiempos estipulados por herramientas como “Google Calendar”.
Retraso en la entrega de trabajos	2	15%	Tener medidas preventivas como la realización de subtarea en tiempos previos a la entrega del trabajo más grande.
Falta de conocimiento	3	60%	Asignar tiempo adicional para estudiar y adquirir conocimientos de las herramientas a utilizar.
Incompatibilidad entre componentes del hardware	2	30%	Revisar las especificaciones de cada dispositivo para cumplir con todos los requisitos específicos de cada uno.
Mala comunicación entre integrantes	4	50%	Motivar, estimular y comprometer a los integrantes del equipo a informar y comunicar todo error de proyecto o idea adicional para el proyecto.

Tabla 7: Gestión de riesgos.

4. Planificación de los procesos técnicos

4.1. Modelo de proceso

4.1.1. Requerimientos

Los requerimientos funcionales y no funcionales son esenciales para el desarrollo y diseño de sistemas, brindando una base importante para crear soluciones tecnológicas que satisfacen tanto las necesidades como las expectativas de sus usuarios.

Requerimientos funcionales

1. **Monitoreo de parámetros**: Registrar en tiempo real la temperatura, pH, y niveles de luz en el acuario.
2. **Automatización mediante accionadores**: Activar calefactores, regular luces led, ajustar el pH si los valores se desvían de los niveles deseados.
3. **Alerta de anomalías**: Notificar al usuario cuando algún parámetro ambiental salga del rango establecido.
4. **Interfaz de usuario**: Proveer una aplicación remota para visualizar el estado del acuario y recibir notificaciones.
5. **Configuración de parámetros**: Permitir al usuario establecer rangos óptimos para cada parámetro de acuerdo a los requerimientos de las especies en el acuario.
6. **Registro de alertas y parámetros**: El sistema debe registrar las alertas ocurridas y promediar por un lapso de tiempo los parámetros en tiempo real.

Requerimientos no funcionales

1. **Interfaz intuitiva:** Facilitar la navegación para que los usuarios puedan monitorear y ajustar configuraciones del acuario sin dificultad.
2. **Tiempos de respuesta bajos:** Minimizar los tiempos de respuesta en la conexión remota para un rápido envío de los parámetros del acuario y envío de alertas.
3. **Asegurar la escalabilidad:** Permitir la integración de sensores adicionales o accionadores para adaptarse a futuros requerimientos o mejoras del sistema.

4.1.2. Descripción de la arquitectura

La arquitectura del proyecto se basa en la estructura cliente-servidor y se representa de la siguiente manera:

Cliente: El usuario utilizará la aplicación que permitirá el monitoreo y control del sistema. A través de esta aplicación el usuario podrá recibir alertas en tiempo real sobre el estado de los parámetros críticos del entorno.

Servidor: El servidor está implementado en una Raspberry Pi, sobre ella se ejecuta un software que actúa como intermediario entre el hardware (sensores y actuadores) y el usuario.

Además, tanto el cliente como el servidor deben estar conectados a la misma red Wi-Fi, lo que facilita la comunicación y transferencia de datos en tiempo real entre ambos.

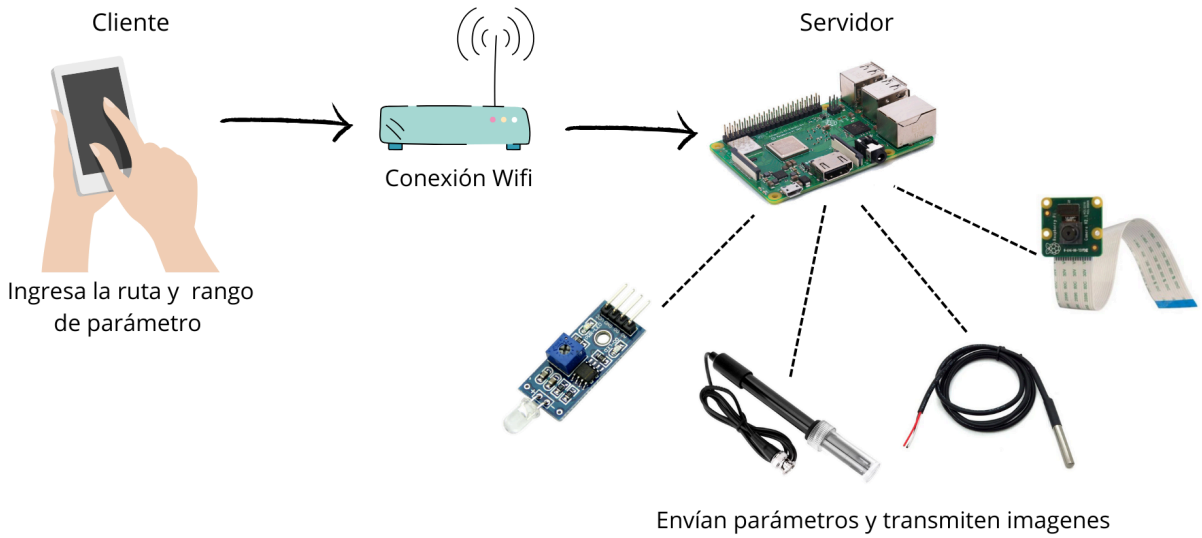


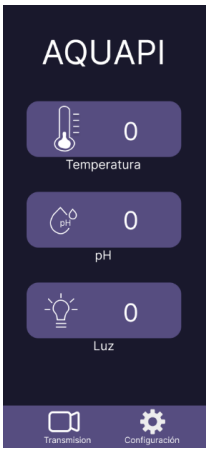
Ilustración 2: Descripción de arquitectura.

4.1.3. Diseño de interfaz de usuario

Interfaz Establecer conexión:

Se ingresa la ruta para establecer la conexión remota.

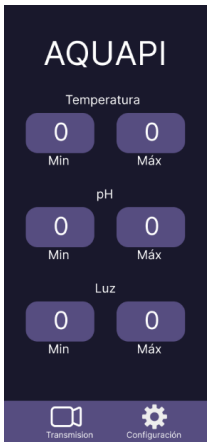
Ilustración 3: Interfaz establecer conexión.



Interfaz Monitoreo de parámetros:

Establecida la conexión remota, se muestran los datos de los parámetros críticos arrojados por los sensores.

Ilustración 4: Interfaz monitoreo de parámetros.



Interfaz Configuración de umbral de parámetros:

En el apartado de configuración se muestran las casillas para ingresar el umbral de valores para cada uno de los parámetros críticos, de modo que funciona como una restricción que activa una alerta en caso de que los sensores arrojen mediciones que se escapen de ese rango definido de valores.

Ilustración 5: Interfaz configuración de umbral de parámetros.

Interfaz Transmisión:

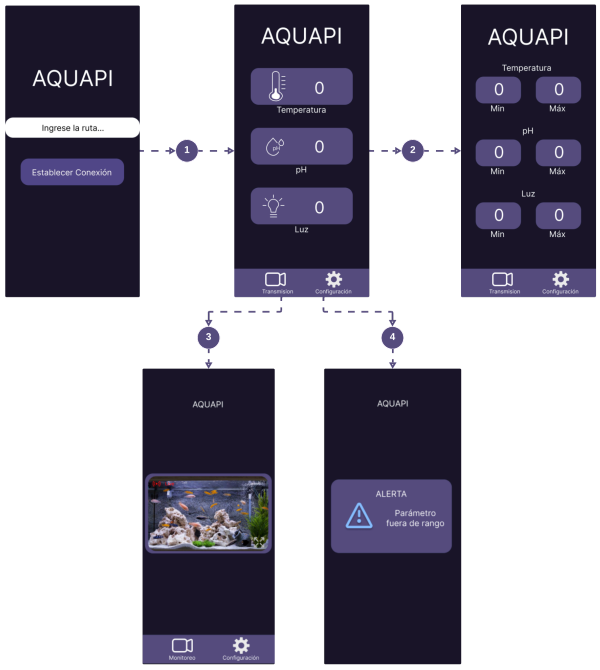
En el apartado de transmisión se muestra un directo en vivo del acuario mediante la cámara.

Ilustración 6: Interfaz transmisión.

Interfaz Alerta:

El usuario recibirá una alerta en caso de que algún parámetro esté fuera de rango.

Ilustración 7: Interfaz alerta.



Secuencia de pantallas

La secuencia de pantallas de la aplicación AquaPI muestra el flujo de interacción del usuario para monitorear y controlar los parámetros del acuario.

Ilustración 8: Secuencia.

4.1.4. Caso de uso general

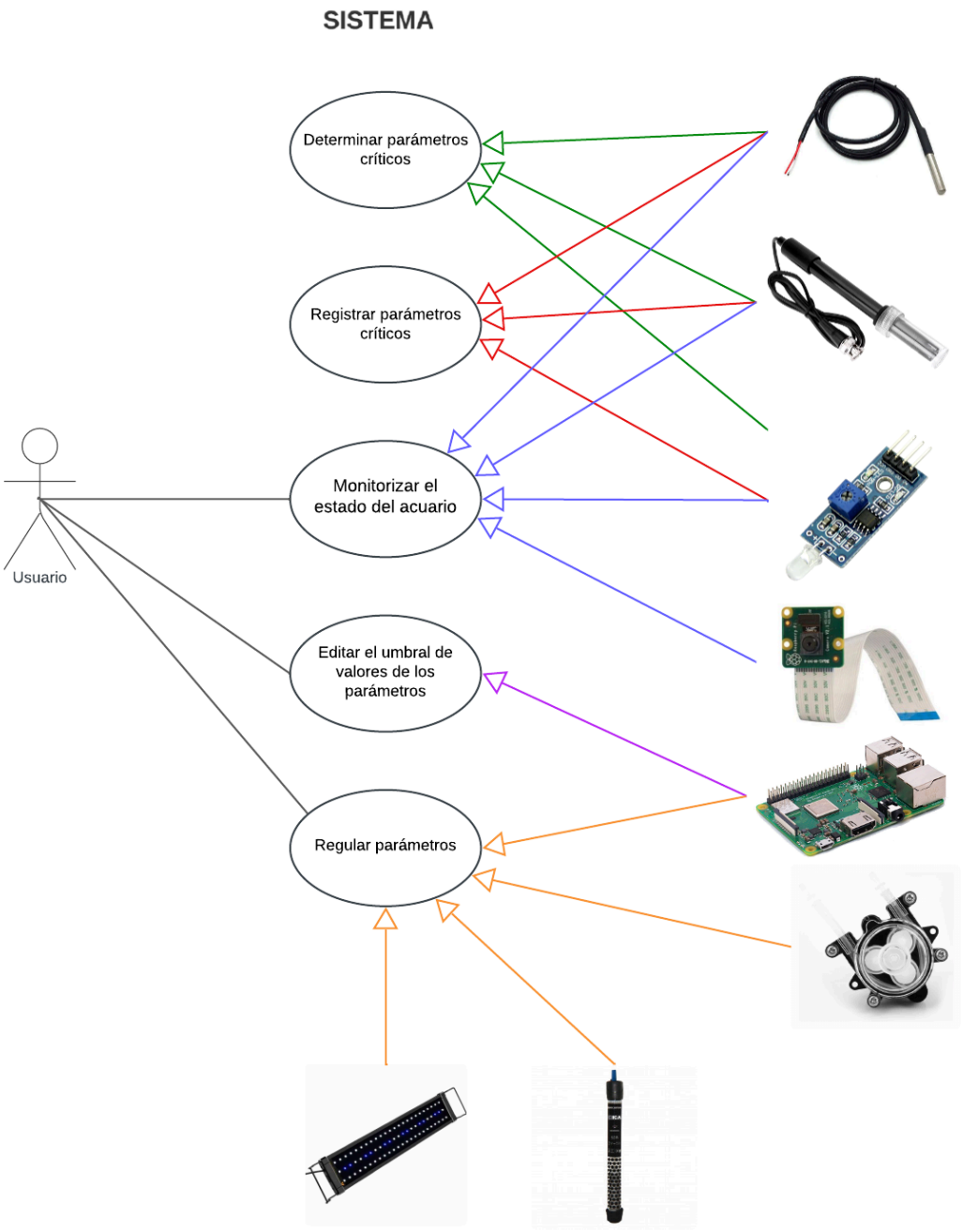


Ilustración 9: Caso de uso general.

4.1.5. Casos de uso

<table>
<tr><td colspan="2">Nombre CUS: Determinar parámetros críticos</td></tr>
<tr><td colspan="2">Autor/Fecha: Katalina Oviedo 30/10/2024</td></tr>
<tr><td colspan="2">Descripción: La Raspberry Pi determina si los parámetros se encuentran dentro de los umbrales preestablecidos en función de los datos obtenidos.</td></tr>
<tr><td colspan="2">Actor: Raspberry pi</td></tr>
<tr><td colspan="2">Precondición: Los datos de los parámetros deben estar registrados en el sistema.</td></tr>
<tr><td>Flujo Principal: Raspberry pi

1. La Raspberry Pi pide los datos de los parámetros al sistema.

3. Analiza y verifica si los datos de los parámetros están dentro de los umbrales preestablecidos.</td><td>Flujo Principal: Sistema

2. El sistema envía los datos de los parámetros.</td></tr>
<tr><td colspan="2">Postcondiciones: No aplica.</td></tr>
</table>

Tabla 8: CUS Determinar parámetros críticos.

<table>
<tr><td colspan="2">Nombre CUS: Registrar parámetros críticos.</td></tr>
<tr><td colspan="2">Autor/Fecha: Cristhian Sánchez 30/10/2024</td></tr>
<tr><td colspan="2">Descripción: Los sensores, al estar conectados al sistema, envían los datos para que estos sean procesados, definiendo el estado de los parámetros críticos para almacenarlo.</td></tr>
<tr><td colspan="2">Actor: Sensor temperatura, sensor de luz, sensor de pH</td></tr>
<tr><td colspan="2">Precondición:</td></tr>
<tr><td>Flujo Principal: Sensores

2. Miden y envían las mediciones de:

● temperatura(Celsius).
● nivel de pH.
● nivel de luz.</td><td>Flujo Principal: Sistema

1. Solicita que los sensores midan los parámetros críticos.

3. Almacena los parámetros críticos de los sensores, la fecha y hora del registro.</td></tr>
<tr><td colspan="2">Postcondiciones: Se registra la temperatura, nivel de pH y el nivel de luz, junto con la fecha y hora del registro.</td></tr>
</table>

Tabla 9: CUS Registrar parámetros críticos.

Nombre CUS: Regular parámetros.	
Autor/Fecha: Bruno Améstica 30/10/2024	
Descripción: La Raspberry analiza los datos, que tienen actualmente los sensores de pH y Temperatura, al detectar una irregularidad en los sensores, alertará al usuario con un mensaje en su dispositivo y solicitará que ingrese el id del sensor, para activar a los actuadores que regulan el pH y/o Temperatura en el acuario.	
Actor: Usuario	
Precondición: Los datos de los parámetros deben estar registrados en conjunto de su respectivo umbral de valores en el sistema.	
Flujo Principal: Usuario 2. Recibe la alerta en su dispositivo, el cual tiene un mensaje: “Alerta, Temperatura, pH o nivel de luz irregulares”. 3. Se ingresa el ID sensor de pH.	Flujo Principal: Sistema 1. Un parámetro escapa de su umbral definido, manda una alerta junto con el id del sensor. 4. Se activa la bomba peristáltica, para estabilizar y regular el pH en el acuario.
Flujo Alternativo: 3.1.1 Se ingresa el ID sensor de temperatura.	Flujo Alternativo: 3.1.2 Se activa el calefactor, para estabilizar y regular la Temperatura en el acuario.
3.2.1 Se ingresa el ID sensor de Luz	3.2.2 Se realizarán cambios en el nivel de iluminación en el panel LED, para estabilizar y regular la luz en el acuario.
Postcondiciones: No aplica.	

Tabla 10: CUS Regular parámetros.

<table>
<tr><td colspan="2">Nombre CUS: Monitorizar el estado del acuario</td></tr>
<tr><td colspan="2">Autor/Fecha: Cristhian Sánchez 30/10/2024</td></tr>
<tr><td colspan="2">Descripción: El usuario revisa el estado del acuario según los datos de los diferentes sensores.</td></tr>
<tr><td colspan="2">Actor: Usuario</td></tr>
<tr><td colspan="2">Precondición: Debe estar registros los parámetros críticos en el sistema y la cámara ha de estar conectada.</td></tr>
<tr><td>Flujo Principal: Usuario

2.- Selecciona ver las mediciones de los parámetros críticos.</td><td>Flujo Principal: Sistema

1.- Muestra las siguientes opciones:

● Ver las mediciones de los parámetros críticos.
● Ver acuario.

3.- Muestra los parámetros críticos entregados por los sensores.</td></tr>
<tr><td>Flujo Alternativo: Usuario
2.1 Selecciona ver acuario.</td><td>Flujo Alternativo: Sistema

2.2 Muestra la transmisión del acuario mediante la cámara.</td></tr>
<tr><td colspan="2">Postcondiciones: No aplica.</td></tr>
</table>

Tabla 11: CUS Monitorizar el estado del acuario.

Nombre CUS: Editar el umbral de valores de los parámetros	
Autor/Fecha: Jorge Cáceres 30/10/2024	
Descripción: Permite al Usuario editar el umbral de parámetros de cada sensor.	
Actor: Usuario.	
Precondición: Los datos de los umbrales de valores de parámetros deben estar en el sistema.	
Flujo Principal: Usuario 1. El Usuario solicita al sistema el umbral de valores de los parámetros de todos los sensores 3. El Usuario selecciona el sensor que desee editar mediante su id. 4. El Usuario modifica el umbral de valores de los parámetros 7. El Usuario confirma los cambios.	Flujo Principal: Sistema 2. El sistema por cada sensor muestra el umbral de valores de los parámetros y el id del sensor 5. El sistema válida que los valores ingresados sean válidos y estén dentro de los límites permitidos 6. El sistema muestra el nuevo umbral de valores a cambiar y solicita la confirmación
Flujo Alternativo:	Flujo Alternativo: 6.1 En caso de que los valores no sean válidos, el sistema mostrará un mensaje de error y solicitará los valores de nuevo.
Postcondiciones: El nuevo umbral de valores de parámetros se registran en el sistema	

Tabla 12: CUS Editar el umbral de valores de los parámetros.

4.1.6. Diagramas de secuencia

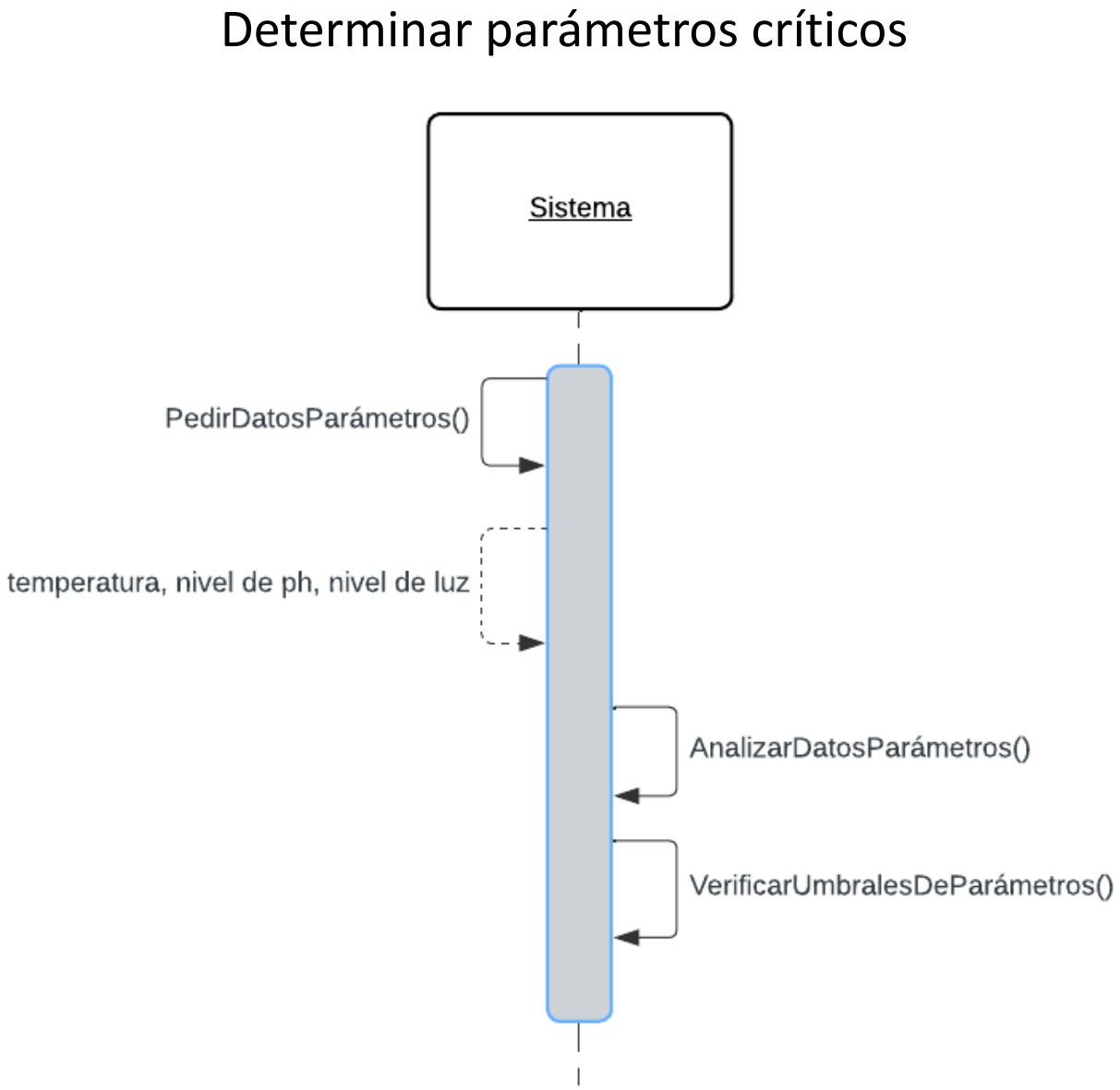


Ilustración 10: Diagrama determinar parámetros críticos.

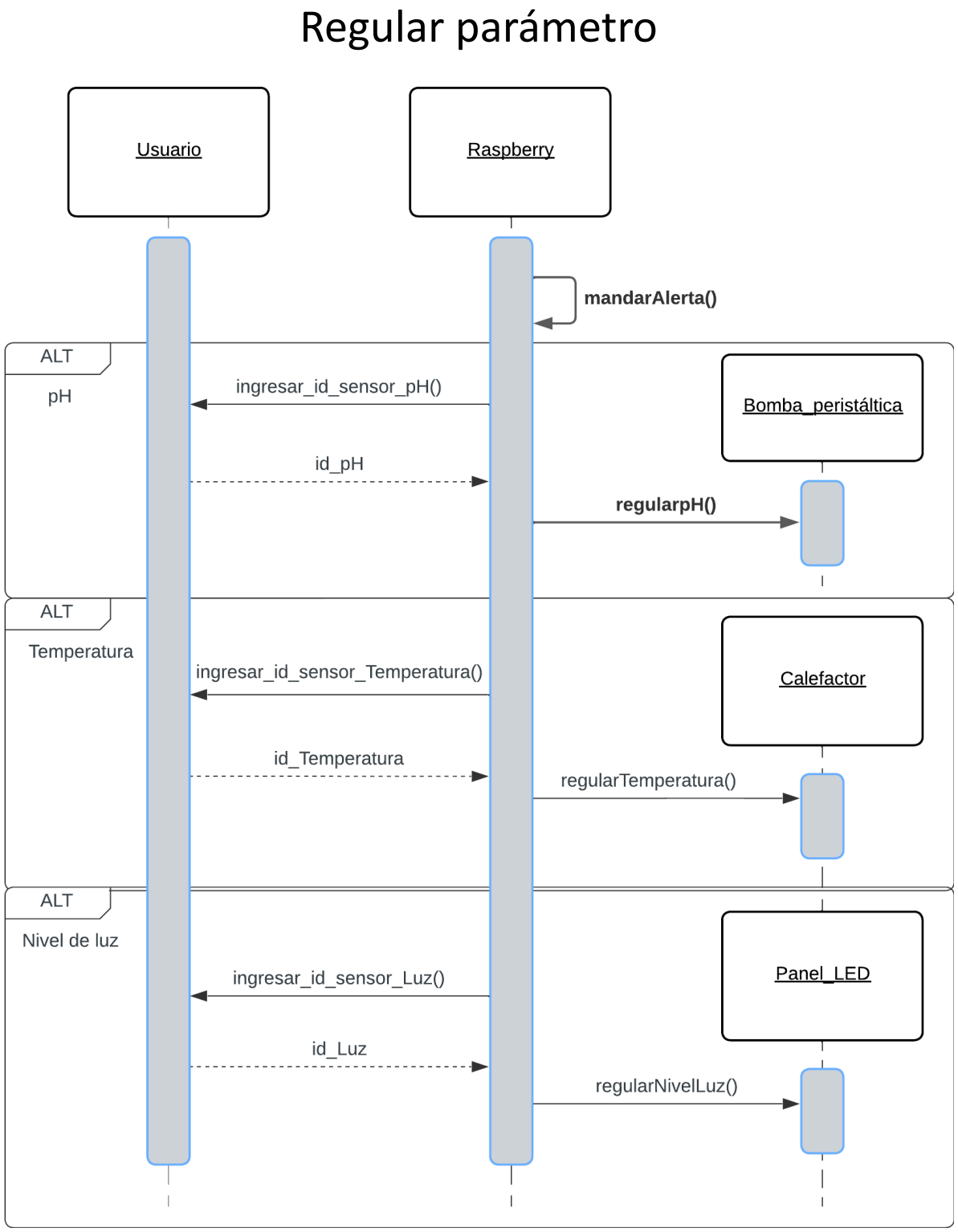


Ilustración 11: Diagrama regular parámetro.

Registrar parámetros críticos

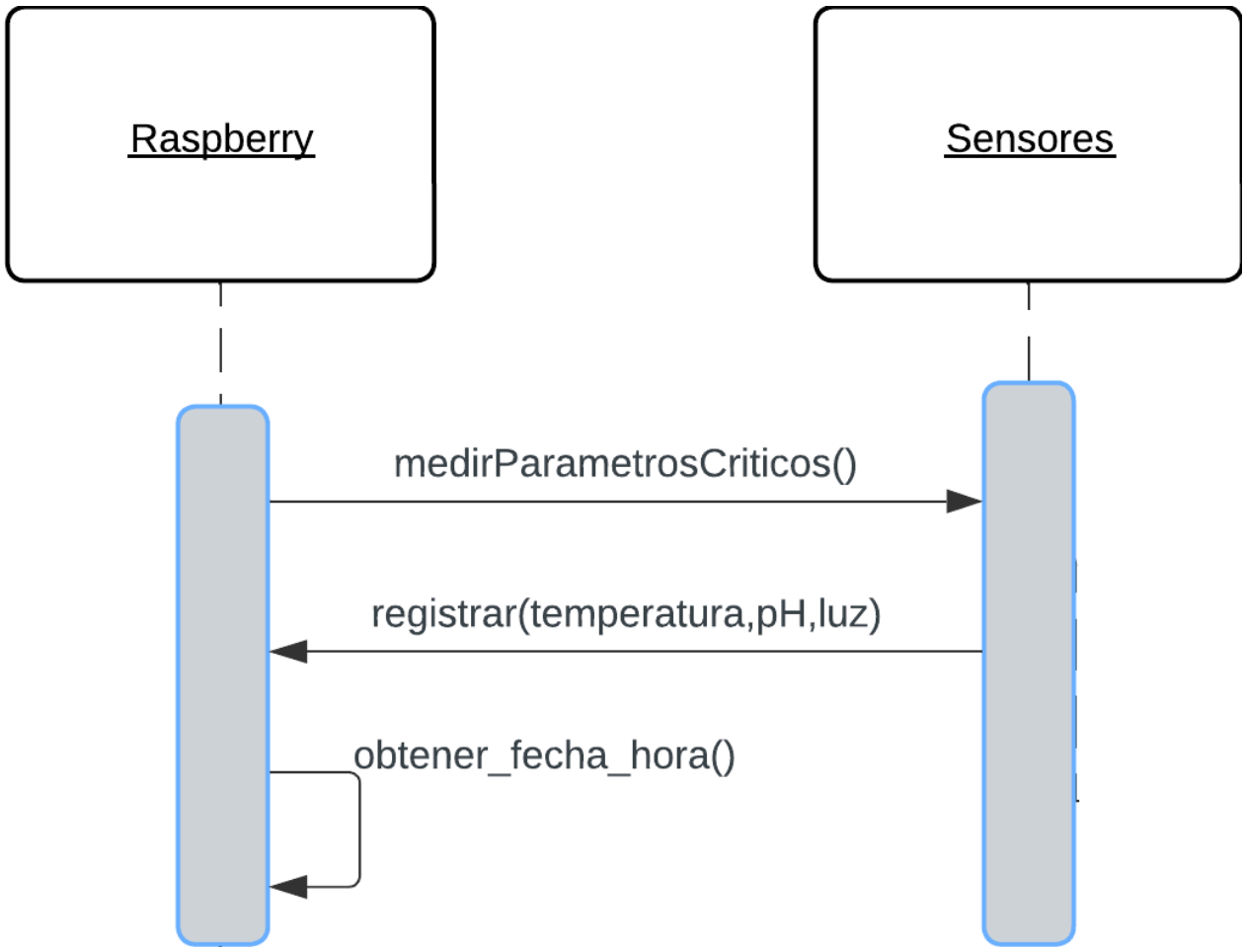


Ilustración 12: Diagrama registrar parámetros críticos.

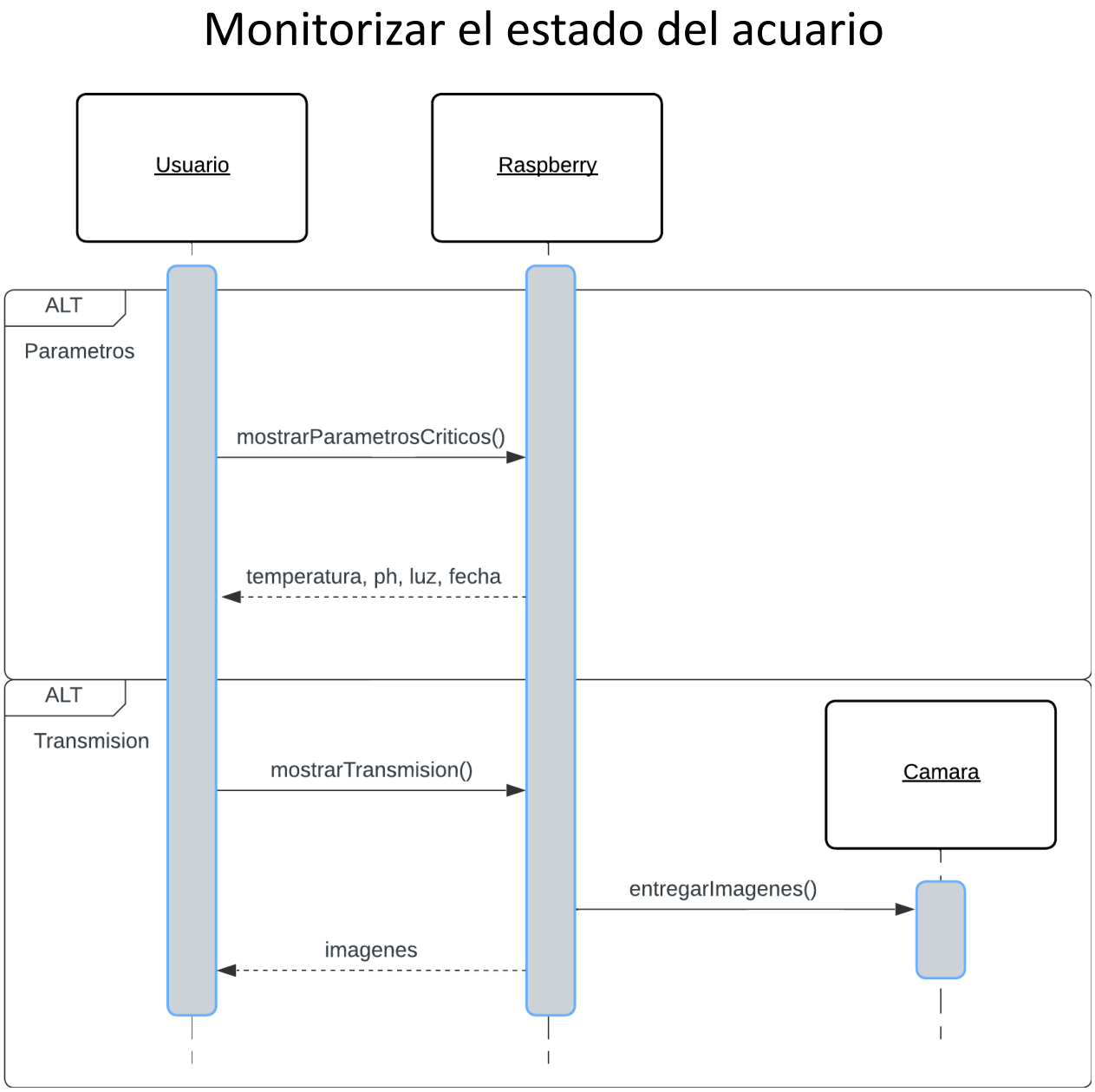


Ilustración 13: Diagrama monitorizar el estado del acuario.

Editar el umbral de valores de los parámetros

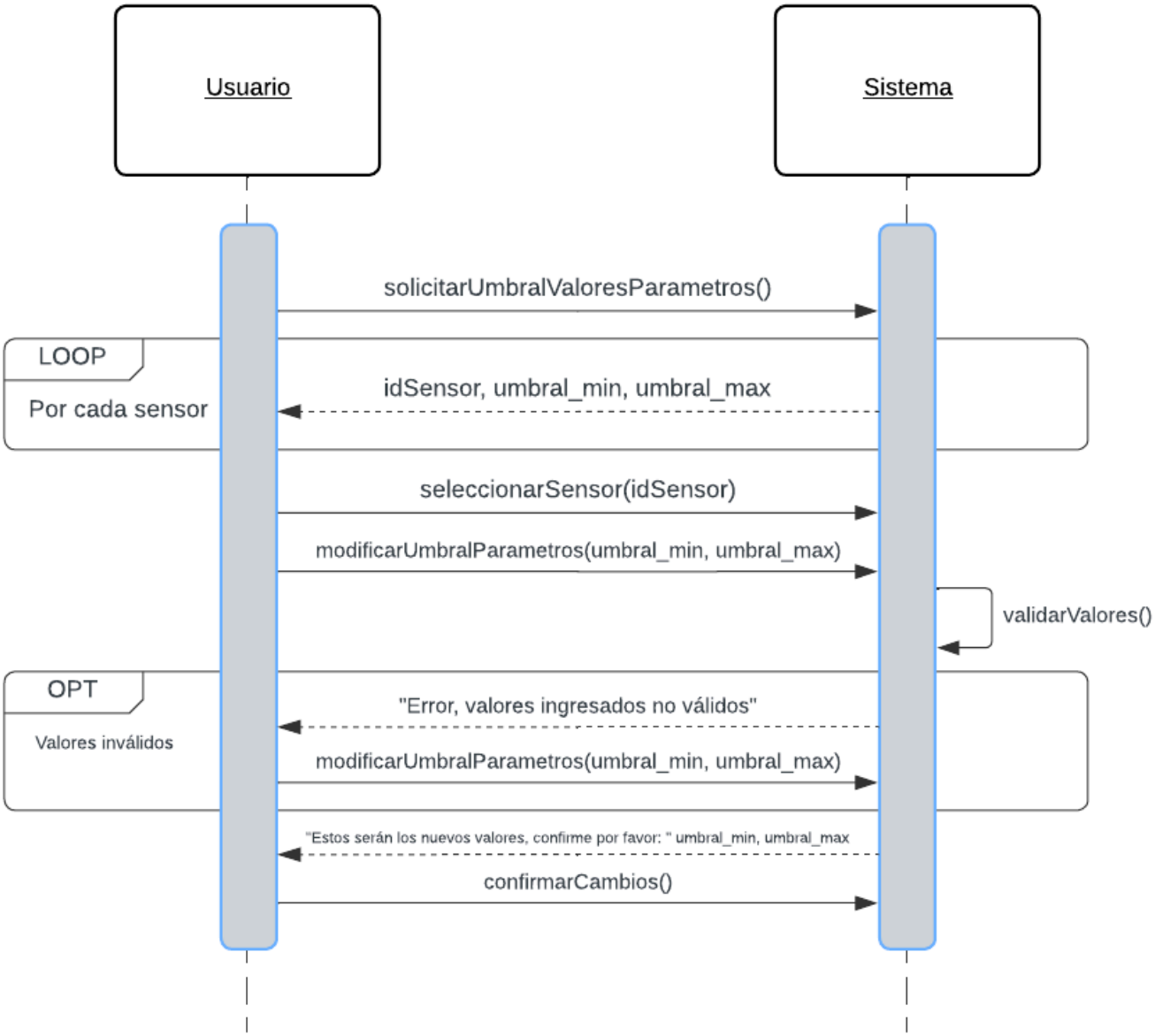


Ilustración 14: Diagrama editar el umbral de valores de los parámetros.

4.2. Herramientas y técnicas

Para llevar a cabo el desarrollo y ejecución del proyecto, será necesario contar con distintas herramientas. Las herramientas necesarias para la implementación son las siguientes:

Visual Studio Code: Un entorno de desarrollo integrado (IDE) que facilita la escritura, depuración y administración de código. Utilizaremos Visual Studio Code para desarrollar y editar el código Python y para gestionar el proyecto en general.

Arduino IDE: Herramienta para programar y depurar microcontroladores, utilizando un lenguaje basado en C++ simplificado. En este proyecto, se usará para desarrollar firmware que controle sensores y actuadores, y para configurar la comunicación con la Raspberry Pi, asegurando la transmisión de datos hacia Python.

Python: Un lenguaje de programación versátil y de alto nivel. Se utilizará Python como el lenguaje principal para desarrollar la lógica de la aplicación.

C/C++: Lenguajes de programación eficientes y versátiles. C se enfoca en bajo nivel y rendimiento, mientras que C++ añade programación orientada a objetos y características avanzadas para mayor flexibilidad.

Para obtener los datos de los sensores se utilizará la librería RPi.GPIO, la cual es una librería de Python que permite obtener entradas de los dispositivos de entrada y salida de propósito general.

5. Implementación

5.1. Plan de integración

1. Integración de Hardware y Software

Se verificará la correcta conexión de los sensores de temperatura, pH y luz con la Raspberry Pi, realizando pruebas de calibración para asegurar que las lecturas sean precisas. Dado que algunos de estos sensores generan señales analógicas, se integrará un Arduino como intermediario para convertir esas señales a digitales, permitiendo su envío y procesamiento por la Raspberry Pi. Además, se instalará el sistema operativo en la Raspberry Pi, junto con los controladores y librerías necesarias. Posteriormente, se configurarán las librerías adecuadas para garantizar una comunicación fluida entre los sensores y la Raspberry Pi.

2. Integración de la Interfaz de Usuario

La interfaz de usuario será desarrollada utilizando PySide6, lo que permitirá crear una aplicación accesible y fácil de usar. La funcionalidad de la aplicación incluirá la conexión con la Raspberry Pi a través del módulo socket. Al ingresar la IP y presionar el botón de conexión, la interfaz establecerá la conexión con la Raspberry Pi. Los datos de los sensores (temperatura, pH y luz) se recibirán en tiempo real y se mostrarán en la interfaz. Además, los usuarios podrán ajustar los valores de los parámetros (mínimos y máximos) directamente desde la interfaz, lo que les permitirá controlar los rangos de los sensores de forma sencilla.

3. Pruebas del sistema

Se realizarán pruebas para asegurar que el sistema AquaPi funcione correctamente en diversos escenarios. En las pruebas de simulación de parámetros del acuario, se verificará que el sistema reciba y muestre correctamente la información de los parámetros como la temperatura, el pH y la luz. Además, se comprobará que el sistema envíe notificaciones al usuario si alguno de los parámetros está fuera de los rangos establecidos. También se simularán fallos en los sensores para asegurarse de que el sistema pueda detectar estos problemas y generar alertas de mantenimiento predictivo. Finalmente, se realizarán pruebas de conectividad para verificar que la conexión entre la Raspberry Pi y la interfaz sea exitosa, y se asegurará que todos los elementos interactivos de la interfaz, como botones y cuadros de texto, funcionen correctamente, con actualizaciones en tiempo real y transiciones fluidas entre pantallas.

5.2. Modelo de implementación

Para la implementación del proyecto AquaPi, se utilizaron diversas librerías que facilitaron el desarrollo de la aplicación y la integración de los diferentes componentes del sistema. A continuación, se detallan las principales librerías utilizadas:

PySide6: Librería principal utilizada para el desarrollo de la interfaz gráfica de usuario (GUI) en AquaPi. PySide6 es un conjunto de herramientas para crear aplicaciones en Python utilizando Qt. Esta librería permitió construir la interfaz intuitiva que permite al usuario monitorear y controlar los parámetros del acuario.

Socket: La librería socket en Python se utilizó para establecer la comunicación cliente-servidor entre la aplicación AquaPi (cliente) y la Raspberry Pi (servidor). Utilizando el protocolo TCP/IP, la aplicación recibe datos de los sensores (temperatura, pH y luz) en tiempo real desde el servidor (Raspberry Pi). Esta librería fue fundamental para asegurar la transmisión de datos eficiente y sin interrupciones entre los dispositivos.

Re (expresiones regulares): Se utilizó la librería re para validar la IP ingresada por el usuario en la pantalla de conexión. Esta librería permite verificar que la IP tenga un formato válido antes de intentar establecer la conexión con el servidor.

Serial: La librería serial se utilizó para la comunicación entre la Raspberry Pi y los dispositivos conectados a través de los puertos seriales. En AquaPi, se utilizó para recibir y enviar datos desde los sensores de pH y luz, que están conectados al puerto serial.

Signal: Signal se utilizó para crear señales personalizadas en la aplicación. Cuando se reciben nuevos datos de los sensores desde la Raspberry Pi, Signal emite señales que son captadas por la interfaz para actualizar los valores de los parámetros (temperatura, pH, luz) en tiempo real.

OpenCV (cv2): cv2 es la interfaz de Python para OpenCV, una librería de visión por computadora utilizada para el procesamiento de imágenes y videos. En AquaPi, se utilizó para capturar, procesar y visualizar imágenes en tiempo real, como la transmisión de video en vivo desde las cámaras conectadas al sistema.

5.3. Módulos implementados

Módulo Sensores

Este código de Arduino se encarga de leer los valores generados por los sensores de pH y luz conectados a la placa y enviar estos datos a través del puerto serie para su posterior procesamiento

```
// Pines de los sensores
const int pHSensorPin = A0;  // Sensor de pH
const int lightSensorPin = A1;  // Sensor de luz

void setup() {
  Serial.begin(9600);  // Inicializa la comunicación serial
  pinMode(pHSensorPin, INPUT);  // Configura el pin de pH como entrada
  pinMode(lightSensorPin, INPUT);  // Configura el pin de luz como entrada
}

void loop() {
  // Lectura del sensor de pH
  int rawValue = analogRead(pHSensorPin);  // Lee el valor del sensor de pH
  float voltage = rawValue * (5.0 / 1023.0);  // Convierte a voltaje (suponiendo 5V de referencia)

  // Convertir el voltaje a pH
  float pH = (14.0 / 5.0) * voltage;

  // Lectura del sensor de luz
  int lightValue = analogRead(lightSensorPin);  // Lee el valor del sensor de luz

  // Envía los datos al puerto serie
  Serial.print(" ");
  Serial.print(pH);
  Serial.print(", ");
  Serial.println(lightValue);

  delay(1000);  // Espera 1 segundo antes de la siguiente lectura
}
```

Ilustración 15: Módulo Sensores.

Estas funciones permiten al sistema obtener los valores de temperatura del sensor DS18B20, así como los datos de pH y luz provenientes del Arduino, proporcionando la información esencial para realizar el monitoreo en tiempo real del acuario.

```python
# Funciones para leer el sensor de temperatura DS18B20
def read_temp_raw():
    with open(device_file, 'r') as f:
        lines = f.readlines()
    return lines

def read_temp():
    lines = read_temp_raw()
    while lines[0].strip()[-3:] != 'YES':
        time.sleep(0.2)
        lines = read_temp_raw()
    equals_pos = lines[1].find('t=')
    if equals_pos != -1:
        temp_string = lines[1][equals_pos + 2:]
        temp_c = float(temp_string) / 1000.0
        temp_f = temp_c * 9.0 / 5.0 + 32.0
        return temp_c, temp_f

# Función para leer los sensores de pH y luz desde Arduino
def read_ph_and_light():
    try:
        if arduino.in_waiting > 0:
            line = arduino.readline().decode('utf-8').strip()  # Lee los datos del Arduino
            return line
    except Exception as e:
        print(f"Error al leer del Arduino: {e}")
    return None
```

Ilustración 16: Módulo Sensores 2.

La función receive_camera_stream() establece una conexión con el servidor de la cámara, recibe los paquetes de datos en formato JPEG, los decodifica y los muestra en tiempo real en una ventana de OpenCV

```
# Función para recibir y mostrar video
def receive_camera_stream():
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.connect((host, port_camera))
        try:
            buffer = b''
            while True:
                buffer += s.recv(4096)
                start = buffer.find(b'\xff\xd8')  # Inicio del JPEG
                end = buffer.find(b'\xff\xd9')    # Fin del JPEG
                if start != -1 and end != -1:
                    frame = buffer[start:end+2]
                    buffer = buffer[end+2:]
                    frame = cv2.imdecode(np.frombuffer(frame, np.uint8), cv2.IMREAD_COLOR)
                    if frame is not None:
                        cv2.imshow("Video en Vivo", frame)
                        if cv2.waitKey(1) & 0xFF == ord('q'):
                            break
        except KeyboardInterrupt:
            print("Desconectado del servidor de cámara.")
        finally:
            cv2.destroyAllWindows()
```

Ilustración 17: Módulo Sensores 3.

Módulo servidor

Este código configura un servidor TCP en la Raspberry Pi que se encarga de leer los datos de los sensores de temperatura, pH y luz, procesarlos y enviarlos en tiempo real al cliente conectado. De esta manera, permite un monitoreo continuo y eficiente de las condiciones del acuario.

```python
# Configuración del servidor
host = ''
port = 5560

def setupServer():
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    print("Socket creado.")
    try:
        s.bind((host, port))
    except socket.error as msg:
        print(msg)
    print("Socket vinculado.")
    return s

def setupConnection(s):
    s.listen(1)
    conn, address = s.accept()
    print(f"Conectado a: {address[0]}:{address[1]}")
    return conn

def send_sensor_data(conn):
    while True:
        try:
            # Lee los datos de los sensores
            temp_c, temp_f = read_temp()
            ph_and_light = read_ph_and_light()

            # Formatea los datos
            sensor_data = f"{temp_c:.2f}, "
            if ph_and_light:
                sensor_data += f"{ph_and_light}"
            else:
                sensor_data += "No se recibieron datos de pH y luz.\n"

            # Envía los datos al cliente
            conn.sendall(sensor_data.encode('utf-8'))
            time.sleep(1)
        except (ConnectionResetError, BrokenPipeError):
            print("Cliente desconectado.")
            break

s = setupServer()

while True:
    conn = setupConnection(s)
    send_sensor_data(conn)
```

Ilustración 18: Módulo Servidor.

Módulo Interfaz

La clase DataReceiver se encarga de establecer la conexión con el servidor, recibir los datos de los sensores (temperatura, pH y luz), procesarlos y emitir una señal para actualizar la interfaz de usuario en tiempo real.

```
class DataReceiver(QThread):
    # Señales para actualizar los valores en la interfaz
    data_received = Signal(float, float, int)

    def __init__(self, host, port):
        super().__init__()
        self.host = host
        self.port = port
        self.running = True

    def run(self):
        try:
            # Conexión al servidor
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((self.host, self.port))

            while self.running:
                # Recibir datos del servidor
                data = s.recv(1024)
                if not data:
                    break

                # Decodificar y procesar los datos
                decoded_data = data.decode('utf-8').strip()
                print(decoded_data)
                try:
                    # Validar que los datos contengan exactamente dos comas (indicativo del formato esperado)
                    if decoded_data.count(',') == 2:
                        temp, ph, light = map(float, decoded_data.split(','))
                        self.data_received.emit(temp, ph, int(light))
                        print("FUNCIONA")
                    else:
                        # Ignorar los datos que no tienen el formato correcto
                        print("ERROR")
                except ValueError:
                    # Manejar cualquier error inesperado en el procesamiento
                    print("ERROR2")
        except Exception as e:
            print("Error en la conexión:", e)

    def stop(self):
        self.running = False
        self.quit()
```

Ilustración 19: Módulo Interfaz.

La función verificar_conexion asegura que el usuario ingrese una IP válida y, si la validación es exitosa, intenta conectar con el servidor de sensores. Si la conexión es exitosa, se establece un flujo continuo de datos a la interfaz. Si la IP es incorrecta o hay un error en la conexión, se muestra un mensaje de error en la interfaz.

```
def verificar_conexion(self, ip):
    if not self.validar_ip(ip):
        QMessageBox.critical(self, "Error de IP", "La IP ingresada no es válida.")
        return

    try:
        self.data_receiver = DataReceiver(ip, 5560)
        self.data_receiver.data_received.connect(self.actualizar_datos)
        self.data_receiver.start()
        self.mostrar_pantalla_monitoreo()
    except Exception as e:
        QMessageBox.critical(self, "Error de Conexión", f"No se pudo conectar al servidor: {e}")

def validar_ip(self, ip):
    # Verificar si la IP ingresada tiene un formato válido
    ip_regex = re.compile(r"^(?:[0-9]{1,3}\.){3}[0-9]{1,3}$")
    return ip_regex.match(ip) is not None
```

Ilustración 20: Módulo Interfaz 2.

6. Problemas encontrados y soluciones propuestas

1. Problema con el sensor de pH y de luz
Los sensores de pH y de luz generan señales analógicas, pero la Raspberry Pi no puede recibir señales analógicas directamente, ya que solo puede procesar señales digitales.

Solución:
Para resolver este problema, se implementó un **Arduino** como intermediario. El Arduino recibe las señales analógicas de los sensores, las convierte en digitales y luego las envía a la Raspberry Pi para su procesamiento y monitoreo.

2. Problema de compatibilidad del GrovePi
Durante la integración del GrovePi con la Raspberry Pi, se presentó un problema de **compatibilidad de librerías**. Las librerías necesarias para que el GrovePi funcionara correctamente no eran compatibles con las versiones más recientes de Raspberry Pi OS, lo que impedía la comunicación efectiva entre los sensores conectados al GrovePi y la Raspberry Pi.

Solución:
A pesar de los intentos de actualizar las librerías y explorar diversas soluciones, no se logró resolver el problema de compatibilidad. Como solución alternativa, se decidió utilizar un **Arduino** como intermediario. El Arduino recibió las señales de los sensores, las procesó y luego envió los datos digitales a la Raspberry Pi, permitiendo que los sensores funcionaran correctamente sin depender del GrovePi.

3. Problema con la conexión de la interfaz con la Raspberry Pi
Durante la integración de la interfaz de usuario con la Raspberry Pi, se presentaron dificultades en la conexión entre ambos. La interfaz no lograba comunicarse de manera eficiente con la Raspberry Pi, lo que causaba retrasos en la actualización de los datos y dificultaba la interacción en tiempo real con el sistema.

Solución:
Tras realizar una investigación exhaustiva sobre las posibles soluciones, se decidió utilizar el **módulo Socket** para establecer una comunicación directa y eficiente entre la Raspberry Pi y la interfaz de usuario. Este módulo permite crear una conexión estable mediante TCP/IP, asegurando que los datos se transmitan de forma rápida y en tiempo real.

7. Comparación de requerimientos

A continuación, se presenta una tabla que evalúa detalladamente los requerimientos establecidos para el sistema, y su cumplimiento durante el proceso de implementación. Esta comparación permite evaluar el progreso del proyecto y verificar si los objetivos iniciales han sido alcanzados en cada etapa del desarrollo.

Requerimientos	Cumplido	No cumplido
Monitoreo de parámetros	x	
Automatización mediante accionadores		x
Alerta de anomalías	x	
Interfaz de usuario	x	
Configuración de parámetros	x	
Registro de alertas y parámetros	x	

Tabla 13: Comparación de requerimientos.

8. Trabajo futuro

El proyecto AquaPi ha logrado un progreso importante en su desarrollo, sin embargo, aún hay diversas áreas que pueden ser mejoradas y ampliadas para optimizar su rendimiento y ampliar su alcance y funcionalidad.

Expansión de Sensores: Se podría agregar soporte para más sensores, como oxígeno disuelto, nitratos, amoníaco y salinidad, lo que permitiría monitorear más parámetros clave del acuario. Esto brindaría una visibilidad más detallada y precisa de las condiciones del entorno acuático, contribuyendo al bienestar de los organismos.

Automatización del Sistema: Aunque el sistema actualmente solo notifica cuando los parámetros se desvían de los valores óptimos, una mejora significativa sería implementar automatización. Esto permitiría que el sistema ajuste los parámetros automáticamente, como la temperatura, la luz o el pH, mediante actuadores, sin intervención manual, garantizando un entorno más estable para los peces y otras especies.

Optimización de la Interfaz de Usuario: La interfaz gráfica de usuario puede ser optimizada para ser más interactiva y fácil de usar. Incluir gráficos en tiempo real, historial de datos y una visualización más avanzada de los parámetros facilitaría la interpretación de los datos por parte del usuario. Además, se podría implementar la capacidad de personalizar alertas y configuraciones según las necesidades del acuario.

9. Conclusión

La implementación de un sistema IoT para el monitoreo y control de acuarios es crucial para garantizar el bienestar y la estabilidad de los ecosistemas acuáticos. El análisis realizado hasta el momento ha sido fundamental para definir y comprender todos los objetivos de este proyecto, los cuales se cumplirán en las siguientes etapas. A través de un análisis detallado de la ineficiencia en la gestión de parámetros críticos de un acuario, se optó por abordar el problema mediante la preparación de un informe que describe los roles de trabajo, objetivos, costos de recursos y el personal necesario para ejecutar el proyecto, asegurando una adecuada organización.

También se definieron los riesgos potenciales que podrían surgir a lo largo del desarrollo y se propusieron acciones remediales para minimizar su impacto y gestionar el tiempo de trabajo de manera más eficiente. Además, se desarrollaron los planteamientos y diagramas generales que traducen la visión y los objetivos del proyecto, incluyendo la realización de los casos de uso junto con sus diagramas de secuencia. Esto permitió identificar las interacciones clave entre el usuario y el sistema, estableciendo flujos de monitoreo y control de los parámetros críticos. Estas interacciones facilitaron el diseño de una interfaz gráfica minimalista e intuitiva que permite al usuario realizar las acciones necesarias, resultando en una etapa crucial para la planificación y desarrollo de la secuencia de interacciones que cumple con los requerimientos establecidos.

En esta última fase, se logró avanzar significativamente en la integración de los módulos del sistema, permitiendo la recepción de datos en tiempo real desde los sensores de temperatura, pH y luz, y la visualización de estos datos en una interfaz gráfica accesible para el usuario. Se completó la conexión del sistema mediante el uso de sockets TCP/IP, asegurando la comunicación continua entre la Raspberry Pi y la interfaz de usuario. Además, se implementaron las funcionalidades de validación de IP, conexión y recepción de datos, lo que permitió verificar el correcto funcionamiento de los módulos de comunicación y la actualización en tiempo real de los parámetros del acuario. Con los resultados de las pruebas, se ha logrado un sistema confiable y funcional, cumpliendo con los requisitos establecidos. Este sistema no solo proporciona una solución eficiente para el monitoreo de acuarios, sino que también asegura una operación estable y continua. Con las funcionalidades completadas y validadas, AquaPi está listo para ser implementado en un entorno real, brindando una herramienta efectiva para los usuarios que buscan mantener condiciones óptimas en sus acuarios.

10. Referencias

[1] AQ-arium. (2023). Guía definitiva acuario para principiantes 2023. AQ-arium.
https://www.aq-arium.com/aqmarine/guia-definitiva-acuario-para-principiantes-2023/

[2] Raspberry Pi Chile. (n.d.). Project Kit electrónica para estudiantes. Raspberry Pi Chile.
https://raspberrypi.cl/producto/project-kit-electronica-para-estudiantes/

[3]Advantage. (n.d.). Arriendo 30 días notebook Core i3 usado. Advantage.
https://www.advantage.cl/catalog/product/view/id/19739/s/arriendo-30-dias-notebook-core-i3-usado-1/category/1160/

[4] Altronics. (n.d.). Raspberry Pi 3 Model B+. Altronics.
https://altronics.cl/raspberry-pi3-model-b-plus

[5] Altronics. (n.d.). Sensor sonda temperatura DS18B20 3mt. Altronics.
https://altronics.cl/sensor-sonda-temperatura-ds18b20-3mt

[6] Afel. (n.d.). Módulo y sonda de detección de pH 0 a 14, sensor de pH. Afel.
https://afel.cl/products/modulo-y-sonda-de-deteccion-de-ph-0-a-14-sensor-de-ph

[7] Altronics. (n.d.). Módulo LDR, sensor de luz. Altronics.
https://altronics.cl/modulo-ldr?search=Sensor%20de%20luz

[8] Altronics. (n.d.). Display LCD 1602 I2C backlight blue. Altronics.
https://altronics.cl/display-lcd-1602-i2c-backlight-blue?search=pantalla%20lcd

[9] Altronics. (n.d.). Adaptador WiFi EDUP 802.11n EP-N8508GS. Altronics.
https://altronics.cl/edup-80211n-ep-n8508gs?search=adaptador%20wifi

[10] Fauna Salud. (n.d.). Bomba sumergible Sobo para acuario. Fauna Salud.
https://faunasalud.cl/product/bomba-sumergible-sobo-para-acuario/

[11] Fauna Salud. (n.d.). Bomba sumergible Sobo para acuario. Fauna Salud.
https://faunasalud.cl/product/bomba-sumergible-sobo-para-acuario/

[12] Talent.com. (n.d.). Salario de jefe de proyecto en Chile. Talent.com.
https://cl.talent.com/salary?job=jefe+de+proyecto

[13] Talent.com. (n.d.). Salario de programador en Chile. Talent.com.
https://cl.talent.com/salary?job=Programador

[14] Talent.com. (n.d.). Salario de diseñador en Chile. Talent.com. https://cl.talent.com/salary?job=diseñador

[15] Talent.com. (n.d.). Salario de control documental en Chile. Talent.com. https://cl.talent.com/salary?job=control+documental

[16] Talent.com. (n.d.). Salario de técnico hardware en Chile. Talent.com. https://cl.talent.com/salary?job=tecnico+hardware

[17] MoisésBM. (2015, marzo 3). *Acuario marino domótico: Automatización del acuario con Raspberry Pi*. MoisésBM WordPress. https://moisesbm.wordpress.com/2015/03/03/acuario-marino-domotico-automatizacion-del-acuario-con-raspberry-pi/