

TRABAJO ESPECIAL DE GRADO

DISEÑO DE UN SENSOR INTELIGENTE PARA APLICACIONES DE MONITOREO DE SALUD ESTRUCTURAL

Presentado ante la ilustre
Universidad Central de Venezuela
por el Br. Jose Alejandro Tovar Briceño
para optar al título de
Ingeniero Electricista.

Caracas, mayo de 2024

TRABAJO ESPECIAL DE GRADO

DISEÑO DE UN SENSOR INTELIGENTE PARA APLICACIONES DE MONITOREO DE SALUD ESTRUCTURAL

TUTOR ACADÉMICO: Prof. José Romero

Presentado ante la ilustre
Universidad Central de Venezuela
por el Br. Jose Alejandro Tovar Briceño
para optar al título de
Ingeniero Electricista.

Caracas, mayo de 2024

A quien desees dedicar este trabajo

RECONOCIMIENTOS Y AGRADECIMIENTOS

Autor del Trabajo de Grado

Título del Trabajo de Grado

Tutor Académico: MSc Jose Romero. Tesis. Caracas, Universidad Central de Venezuela. Facultad de Ingeniería. Escuela de Ingeniería Eléctrica. Mención Electrónica y Control. Año 2024, xvii, 144 pp.

Palabras Claves: Palabras clave.

Resumen.- Escribe acá tu resumen

ÍNDICE GENERAL

RECONOCIMIENTOS Y AGRADECIMIENTOS	III
ÍNDICE GENERAL	VIII
LISTA DE FIGURAS	XII
LISTA DE TABLAS	XV
LISTA DE ACRÓNIMOS	XVI
INTRODUCCIÓN	1
MARCO REFERENCIAL	5
1.1. Planteamiento del problema	5
1.2. Justificación	5
1.3. Objetivos	7
1.3.1. Objetivo general	7
1.3.2. Objetivos específicos	7
1.4. Antecedentes	8
MARCO TEÓRICO	11
2.1. Estructuras civiles	11
2.1.1. Características generales	11
2.1.2. Tipos de estructuras	11
2.1.3. Comportamiento de las estructuras civiles	12

2.1.4.	Daño en estructuras	16
2.1.5.	Principios de la Sismoresistencia	17
2.2.	Salud estructural	19
2.2.1.	Definición	19
2.2.2.	Reseña histórica	20
2.2.3.	Línea de trabajo del Monitoreo de Salud Estructural . . .	22
2.2.4.	Criterios de evaluación	24
2.2.5.	Variables de interés	27
2.3.	Métodos de identificación de daño	30
2.3.1.	Basado en modelo	30
2.3.2.	Basado en respuesta	30
2.4.	Ensayos estructurales	30
2.4.1.	Ensayos invasivos	31
2.4.2.	Ensayos no invasivos	31
2.5.	Microcontroladores (MCU)	32
2.5.1.	Definición y características	32
2.5.2.	Placas de desarrollo	34
2.5.3.	Protocolos de comunicación	35
2.5.4.	Sistemas Operativos en Tiempo Real (RTOS)	37
2.6.	Sensores	40
2.6.1.	Definición	40
2.6.2.	Características de los sensores	41
2.6.3.	Sensores de interés para el Monitoreo de Salud Estructural	41
2.6.4.	Fusión de sensores	46
2.7.	Sensores inteligentes	49

2.7.1. Definición:	49
2.7.2. Características:	50
2.8. Monitoreo	51
2.8.1. Monitoreo cableado	51
2.8.2. Monitoreo inalámbrico	52
2.9. Sistemas de adquisición de datos	53
2.9.1. Definición y esquema general	53
2.9.2. Acondicionamiento de señales	53
2.9.3. DAQ basados en MCU	53
2.10. Sistemas de comunicaciones inalámbricos	54
2.10.1. Características y conceptos básicos	54
2.10.2. Protocolos de comunicación para distancias cortas	55
2.10.3. Protocolos de comunicación de larga distancia	57
2.10.4. Protocolo LoRa	58
2.11. Procesamiento digital de señales:	63
2.11.1. Estudio en el dominio de la frecuencia	64
2.11.2. Transformada de Fourier	65
2.11.3. Transformada Rápida de Fourier	66
2.11.4. Aumentamiento	68
MARCO METODOLÓGICO	70
3.1. Descripción breve del sistema	70
3.2. Selección de componentes	73
3.2.1. Protocolo de comunicaciones	73
3.2.2. Sensores	74
3.2.3. Microcontroladores	76

3.2.4. Diagramas de selección de componentes:	77
3.3. Detalle del diseño	82
3.3.1. Descripción detallada del sistema:	82
3.3.2. Diagrama general funcionamiento del sensor inteligente en conjunto con la estación base	86
3.3.3. Descripción del hardware	87
3.3.4. Descripción del software	88
3.3.5. Diagrama de flujo del sistema	109
3.3.6. Diagrama de funcionamiento del sistema	109
PRUEBAS Y RESULTADOS	110
4.1. Pruebas de comunicaciones	110
4.2. Pruebas para estimación de inclinación	112
4.3. Pruebas de funcionamiento del prototipo	113
CONCLUSIONES	114
RECOMENDACIONES	116
TÍTULO DEL ANEXO	117
TÍTULO DEL ANEXO	118
TÍTULO DEL ANEXO	119
REFERENCIAS	120

LISTA DE FIGURAS

2.1. Modelo de masa concentrada de 1 grado de libertad (Hurtado, 2000).	14
2.2. Respuesta ante vibración libre en sistema Subamortiguado (Hurta- do, 2000).	16
2.3. Esquema de un sistema de SHM (Li, Deng, y Xie, 2015).	23
2.4. Diagrama general del proceso de SHM.	24
2.5. Esquema general del microcontrolador ESP32 (Espressif).	34
2.6. Estados de las tareas en FreeRTOS (FreeRTOS, 2024).	39
2.7. Arquitectura de sistema embebido basado en FreeRTOS (FreeR- TOS, 2024).	40
2.8. Esquema general de funcionamiento de un termopar (Dunn, 2005).	42
2.9. Esquema general de un sensor de temperatura de tecnología MEMS (Dunn, 2005).	43
2.10. Esquema general de un sensor de humedad capacitivo (Dunn, 2005).	44
2.11. Esquema general de un sensor de humedad resistivo (Dunn, 2005).	44
2.12. Esquema general de un sensor de humedad de tecnología MEMS (Alfaifi y Zaman, 2021).	45
2.13. Esquema general de un acelerómetro piezoeléctrico (Dunn, 2005).	45
2.14. Esquema general de un acelerómetro de tecnología MEMS (Dunn, 2005).	46
2.15. Acelerómetro microelectromecánico (Dunn, 2005).	46
2.16. Esquema general del protocolo MQTT (Aloufi, Alhazmi, y cols., 2020).	56

2.17. Esquema comparativo entre distintas tecnologías tomando en cuenta el rango de alcance y el ancho de banda (Khorsandi y Jalalizad, 2023).	58
2.18. Señal chirp utilizada para la transmisión CSS (Aloufi y cols., 2020).	59
2.19. Ancho de banda representado en el espectro (Aloufi y cols., 2020).	60
2.20. Representación de símbolos en una transmisión LoRa (Aloufi y cols., 2020).	61
2.21. Forma de la trama LoRa (Aloufi y cols., 2020).	62
2.22. Descomposición de señal temporal en el espectro de frecuencia (Siemens, 2019).	65
2.23. Método de <i>peak picking</i> (Jin, Jeong, y Sim, 2021).	67
2.24. Método de cálculo de amortiguamiento por ancho de banda local (Andresen, Bäger, y Hamm, 2019).	68
2.25. Funciones de aventanamiento más utilizadas (Francone, Domenicali, y Di Benedetto, 2006).	69
3.1. Diagrama de araña para selección de microcontrolador.	78
3.2. Diagrama de araña para selección de acelerómetro.	79
3.3. Diagrama de araña para selección de sensor de temperatura y humedad.	80
3.4. Diagrama de araña para selección de unidad de medición inercial.	81
3.5. Diagrama de araña para selección del módulo de comunicaciones.	82
3.6. Esquema general del sistema.	86
3.7. Diagrama de bloques del sensor inteligente.	87
3.8. Diagrama de bloques de la estación base.	88
3.9. Ilustración del proceso para programar el ESP32 (Systems, 2016).	89

4.1. Vista en mapa de distancia máxima de pruebas usando módulo	
SX1278.	111

LISTA DE TABLAS

2.1. Ventajas y desventajas de los enfoques de monitoreo.	52
3.1. Comparación entre protocolos de comunicación inalámbrica, (Ogdol, Ogdol, y Samar, 2018) y (Blackman, 2019)	74
3.2. Comparación entre módulos LoRa del fabricante Semtech (Semtech, 2015).	74
3.3. Comparación entre placas de desarrollo basadas en MCU	77
4.1. Resultados de pruebas realizadas con módulo de comunicaciones Ra-02.	112

LISTA DE ACRÓNIMOS

INTRODUCCIÓN

La seguridad de las infraestructuras es un tema de gran importancia en la actualidad, especialmente cuando se trata de estructuras como edificios o puentes.

Los primeros indicios del monitoreo del estado de las infraestructuras data de nuestros comienzos como especie sedentaria. En la antigüedad, los especialistas utilizaban técnicas de inspección visual y auditiva para detectar posibles problemas en las estructuras, como grietas o ruidos inusuales. Con el tiempo, se desarrollaron técnicas más avanzadas para el monitoreo de estructuras, como la utilización de medidores de deformación, inclinación, sensores de vibración, entre otros.

La integración de la instrumentación con el análisis estructural comenzó a desarrollarse en la década de 1960 con el advenimiento de la informática y la disponibilidad de computadoras capaces de realizar cálculos estructurales complejos. En esa época, se comenzaron a utilizar sistemas de adquisición de datos para recopilar información sobre el comportamiento de las estructuras en tiempo real y utilizarla para calibrar y validar los modelos estructurales.

Actualmente, las normas sismo-resistentes apuntan a estructuras que sean capaces de mantener su integridad ante un evento de cierta magnitud. Además, el monitoreo continuo de ciertos indicadores en la estructura permiten determinar un índice de la salud estructural y ajustar el modelo a las condiciones actuales de la misma para evaluar el cumplimiento de la normativa sismorresistente. Para el monitoreo a largo plazo, el resultado de este proceso es información actualizada periódicamente sobre la capacidad de la estructura para desempeñar su función prevista a la luz del inevitable envejecimiento y degradación resultantes de los

entornos operativos.

Según Balageas, Fritzen, y Güemes (2010), el monitoreo de la salud estructural (SHM) tiene por objeto proporcionar, en cada momento de la vida de una estructura, un diagnóstico del estado de los materiales constitutivos, de las diferentes partes, y del conjunto de estas partes que constituyen la estructura en su totalidad. El estado de la estructura debe permanecer en el ámbito especificado en el diseño, aunque este puede verse alterado por el envejecimiento normal debido al uso, por la acción del medio ambiente y por sucesos accidentales. Gracias a la dimensión temporal de la supervisión, que permite tener en cuenta toda la base de datos histórica de la estructura, y con la ayuda del monitoreo del funcionamiento. También puede proporcionar un pronóstico (evolución de los daños, vida residual, entre otros).

Si consideramos solo la primera función, el diagnóstico, podríamos estimar que el monitoreo de la salud estructural es una forma nueva y mejorada de realizar una evaluación no destructiva. Esto es parcialmente cierto, pero SHM es mucho más. Implica la integración de sensores, posiblemente materiales inteligentes, transmisión de datos, potencia computacional y capacidad de procesamiento en el interior de las estructuras. Permite reconsiderar el diseño de la estructura y la gestión completa de la propia estructura y de la estructura considerada como parte de sistemas más amplios.

En este sentido, el monitoreo de las estructuras se ha convertido en una herramienta esencial para garantizar la seguridad de las personas durante la vida útil de la misma, incluyendo la ocurrencia de eventos de cierta magnitud. Además, el monitoreo de las estructuras puede ayudar a mejorar la eficacia de las normas sismorresistentes, ya que permite validar y mejorar los modelos estructurales utilizados en la normativa.

Según Nagayama (2007), dado que las edificaciones suelen ser grandes y complejas, la información de unos pocos sensores es inadecuada para evaluar con precisión el estado estructural. El comportamiento dinámico de estas estructuras es complejo tanto a escala espacial como temporal. Además, los daños y/o el deterioro es intrínsecamente un fenómeno local. Por lo tanto, para comprender el comportamiento dinámico, el movimiento de las estructuras debe ser supervisado por sensores con una frecuencia de muestreo suficiente para captar las características dinámicas más destacadas. Esta información combinada con el registro del comportamiento estático de la estructura permiten tener una visión más amplia del estado actual de la estructura.

El primer paso, además de un mantenimiento adecuado, para garantizar la seguridad de estas estructuras, es contar con sistemas de monitoreo que permitan detectar posibles daños o fallas en su funcionamiento y tomar medidas preventivas. Por tanto, los sistemas de adquisición de datos y monitoreo son herramientas esenciales en la prevención de accidentes y daños.

A su vez, según Nagayama (2007), un dispositivo inteligente, es decir, con capacidad de procesamiento de datos en el caso de los sensores, es una característica esencial que permite incrementar el potencial de los sensores al ser estos inalámbricos. Los sensores inteligentes pueden procesar localmente los datos medidos y transmitir solo la información importante a través de comunicaciones inalámbricas. Cuando estos son configurados como una red, se extienden las capacidades de los mismos.

Los sensores inteligentes, con sus capacidades de cómputo y de comunicación integradas, ofrecen nuevas oportunidades para la SHM. Sin necesidad de cables de alimentación o comunicación, los costes de instalación pueden reducirse drásticamente. Los sensores inteligentes ayudarán a que el monitoreo de las estructuras con un denso conjunto de sensores sea económicamente práctico. Se espera que los

sensores inteligentes instalados en masa sean fuentes de información muy valiosa para la SHM.

En este trabajo de grado se abordará el diseño para una futura implementación de un sistema de adquisición de datos de bajo costo basado dispositivos programables con capacidad de interconexión para el monitoreo y procesamiento de variables como aceleración, inclinación, humedad y temperatura en estructuras críticas, con el objetivo de prevenir daños y accidentes.

CAPÍTULO I

MARCO REFERENCIAL

1.1. Planteamiento del problema

La seguridad en las estructuras es un tema crítico en la ingeniería, especialmente en lo que respecta a estructuras críticas como edificios y puentes. A medida que el tiempo pasa, estas estructuras pueden deteriorarse y presentar fallas que pueden poner en riesgo la vida de las personas que las utilizan. Por lo tanto, es fundamental contar con sistemas de monitoreo que permitan detectar posibles problemas en las estructuras y tomar medidas preventivas.

Sin embargo, la mayoría de los sistemas de monitoreo de estructuras disponibles en el mercado son costosos y no están diseñados específicamente para aplicaciones de salud estructural. Además, muchos de estos sistemas no tienen capacidad para admitir comunicación inalámbrica, lo que limita su aplicación en estructuras de gran tamaño o en áreas de difícil acceso, además de los costos asociados a un sistema de cableado fiable que no perturbe las mediciones.

1.2. Justificación

Un sensor inteligente que conste de un sistema de adquisición de datos basado en un microcontrolador que recolecte y procese la información adquirida en

conjunto con varios sensores dispuestos en un solo dispositivo permite realizar la medición de variables ambientales y mecánicas de interés en aplicaciones de salud estructural a un bajo costo; además, un sistema de este tipo es flexible en cuanto a las variables a medir, puesto que es compatible con distintos tipos de sensores y métodos de comunicación a utilizarse, permitiendo crear soluciones canalizadas a proyectos en específico, logrando disminuir costos y contribuir de una forma más eficaz al proceso de toma de decisiones estructurales.

Existe variedad en cuanto al hardware de bajo costo que puede utilizarse para la implementación de un prototipo del sistema, lo que ofrece una alternativa atractiva a los sistemas de adquisición actuales, además de la capacidad de transmisión que ofrecen las redes de larga distancia disponibles en conjunto con las capacidades de los microcontroladores, permitiendo que la estación base pueda ubicarse lejos de la estructura.

Para lograr medir todas estas variables en tiempo real es conveniente contar con uno o más microcontroladores capaces de gestionar toda esta información, sin dejar de lado la confiabilidad en la adquisición de estos datos y que a su vez sea capaz de emplear las herramientas necesarias para comunicar estos datos de forma inalámbrica.

Sabiendo esto, existe una necesidad clara de desarrollar un sistema de adquisición de datos y monitoreo inalámbrico de bajo costo para aplicaciones de salud estructural que permita reducir la acción humana, minimizando el error humano y los recursos necesarios, aumentando a su vez la confiabilidad y la seguridad; con este desarrollo se busca verificar que el desempeño de la estructura se encuentra entre los rangos establecidos, y en caso contrario notificarlas con el objetivo de garantizar la seguridad de las personas, mediante la realización de un plan de mantenimiento preventivo y correctivo.

1.3. Objetivos

1.3.1. Objetivo general

Diseñar un sensor inteligente para aplicaciones de monitoreo de salud estructural.

1.3.2. Objetivos específicos

1. Documentar las principales características y la importancia del monitoreo para aplicaciones de salud estructural.
2. Documentar los principales métodos de recolección de datos soportados por los sensores necesarios para la medición de las variables de interés.
3. Evaluar y proponer el protocolo de comunicación que permita el envío fiable de los datos recolectados por el sensor.
4. Seleccionar el hardware adecuado tanto sensores como microcontroladores para la implementación futura del sistema.
5. Diseñar el módulo de programa encargado de la recolección y almacenamiento de los datos provenientes de los sensores.
6. Desarrollar el módulo de programa encargado de la comunicación de los datos usando la plataforma escogida.
7. Desarrollar una aplicación para el monitoreo y ajuste del sensor inteligente.
8. Validar el funcionamiento del sistema haciendo uso de un prototipo de pruebas.

1.4. Antecedentes

La implementación de microcontroladores para sistemas de adquisición de datos es un tema que se ha desarrollado en múltiples aplicaciones. Muchas veces, se logran desarrollar soluciones que permiten disminuir costos sin que eso afecte la calidad de las mediciones. Las soluciones que se tomarán como referencia lograron: Desarrollar sistemas de adquisición de datos basados en microcontroladores para la medición de variables físicas. Además, se han logrado crear redes de sensores inteligentes que permiten facilitar el envío de los datos de forma inalámbrica.

El trabajo de Federici (2014) en "*Design of Wireless Sensor Nodes for Structural Health Monitoring applications*" publicado en Procedia Engineering, aborda el diseño de redes de sensores inalámbricos para el monitoreo del estado de estructuras civiles, centrándose específicamente en el diseño de nodos en relación con los requisitos de diferentes clases de aplicaciones de monitoreo estructural. Los problemas de diseño se analizan con referencia específica a una configuración experimental a gran escala (el monitoreo estructural a largo plazo de la Basílica S. Maria di Collemaggio, L'Aquila, Italia). Se destacaron las principales limitaciones que surgieron y se esbozan las estrategias de solución adoptadas, tanto en el caso de la plataforma de detección comercial como de las soluciones totalmente personalizadas. Se revisan las opciones para el diseño y despliegue del sistema de monitoreo, tanto en el caso de la selección de plataformas comerciales como en el caso del desarrollo de plataformas a la medida. El análisis de los registros llevó a importantes consideraciones en la integridad estructural y seguridad, y permitió la identificación de los parámetros modales de la estructura.

En cuanto al desarrollo realizado por Komarizadehasl (2022) en "*Development of Low-Cost Sensors for Structural Health Monitoring Applications*", el ingeniero Komarizadehasl propone cuatro sistemas de monitorización de alta precisión y

bajo coste. En primer lugar, para medir correctamente las respuestas estructurales, se desarrolla el *Cost Hyper-Efficient Arduino Product* (CHEAP). CHEAP es un sistema compuesto por cinco acelerómetros sincronizados de bajo coste conectados a un microcontrolador Arduino que hace el papel de dispositivo de recogida de datos. Para validar su rendimiento, se efectuaron unos experimentos de laboratorio y sus resultados se compararon con los de dos acelerómetros de alta precisión (PCB393A03 y PCB 356B18). Se concluye que CHEAP puede usarse para Monitoreo de Salud Estructural en estructuras convencionales con frecuencias naturales bajas cuando se cuente con presupuestos de monitoreo escasos.

En segundo lugar, se presenta un inclinómetro de bajo coste, un *Low-cost Adaptable Reliable Angle-meter* (LARA), que mide la inclinación mediante la fusión de distintos sensores: cinco giroscopios y cinco acelerómetros. LARA combina un microcontrolador basado en la tecnología del Internet de las Cosas (*NODEMCU*), que permite la transmisión inalámbrica de datos, y un software comercial gratuito para la recogida de datos (*SerialPlot*). Para confirmar la precisión y resolución de este dispositivo, se compararon sus mediciones en condiciones de laboratorio con las teóricas y con las de un inclinómetro comercial (*HI-INC*). Los resultados de laboratorio de una prueba de carga en una viga demuestran la notable precisión de LARA. Se concluye que la precisión de LARA es suficiente para su aplicación en la detección de daños en puentes.

En tercer lugar, también se dilucida el efecto de la combinación de sensores de rango similar para investigar el aumento de la precisión y la mitigación de los ruidos ambientales. Para investigar la teoría de la combinación de sensores, el ingeniero Komarizadehasl, propone un equipo de medición compuesto por 75 sensores para la medición de distancias acoplados a dos microcontroladores de Arduino. Los 75 sensores son 25 HC-SR04 (analógicos), 25 VL53L0X (digitales) y 25 VL53L1X (digitales). Los resultados muestran que promediando la salida de

varios sensores sin calibrar, la precisión en la estimación de distancia aumenta considerablemente.

El último sistema propuesto por Komarizadehasl presenta un novedoso y versátil sistema de adquisición de datos a distancia que permite el registro del tiempo con una resolución de microsegundos para la sincronización posterior de las lecturas de los sensores inalámbricos situados en diversos puntos de una estructura. Esta funcionalidad es lo que permitiría su aplicación a pruebas de carga estáticas, quasi-estáticas o al análisis modal de las estructuras.

Muttillo (2019) en su trabajo "*Structural health continuous monitoring of buildings - A modal parameters identification system*" propone en su tesis el diseño de un Sistema de Monitoreo de la Salud Estructural (*Structural Health Monitoring*) para monitorear y comprobar continuamente el comportamiento estructural a lo largo de la vida útil del edificio. El sistema, compuesto por un datalogger personalizado y dispositivos esclavos, permite el monitoreo continuo de la aceleración de la estructura gracias a su facilidad de instalación y bajo coste. El sistema propuesto se basa principalmente en un microcontrolador que:

- Se comunica con los nodos a través del bus RS485,
- Sincroniza las muestras de adquisición
- Adquiere los datos medidos por los nodos.

El sistema fue probado en una estructura de aluminio en voladizo, a través de tres campañas experimentales diferentes y los datos medidos, recogidos en una memoria interna del *datalogger*, fueron post-procesados a través de *Matlab*. Los resultados permitieron evaluar con éxito los parámetros modales (frecuencias, amortiguamiento y formas modales) de la estructura analizada y su estado de salud.

CAPÍTULO II

MARCO TEÓRICO

En este capítulo se definirán los conceptos o fundamentos de instrumentación estructural, sensores inteligentes y adquisición de datos, necesarios para llevar a cabo esta investigación.

2.1. Estructuras civiles

2.1.1. Características generales

Una estructura se refiere a un sistema de partes o elementos que se interconectan para cumplir una función específica. En el caso de la ingeniería civil, suelen ser miembros que se utilizan para soportar una carga. Algunos ejemplos importantes son los edificios, los puentes y las torres; y en otras ramas de la ingeniería, son importantes las corazas de barcos y aviones, los sistemas mecánicos y las estructuras que soportan las líneas de transmisión eléctrica (Hibbeler y Nolan, 1997).

2.1.2. Tipos de estructuras

Según Hibbeler y Nolan (1997), cada sistema está formado por uno o varios de los cuatro tipos básicos de estructuras:

- Celosías.
- Cables y arcos.
- Armazones.
- Estructuras de superficie.

En general, estos elementos suelen soportar cargas, pueden ser estacionarios y también estar restringidos. Sus diferencias suelen basarse en la cantidad de fuerzas a las que están sujetos estos elementos en un instante dado.

La combinación de estos elementos y los materiales que los componen es lo que se denomina un sistema estructural. Estos sistemas, aunque sean pasados por alto, son utilizados diariamente por industrias y personas, siendo elementos claves en el desarrollo y progreso de la civilización actual.

2.1.3. Comportamiento de las estructuras civiles

La gran mayoría de los sistemas cuentan con una respuesta dinámica y estática. Ambas respuestas permiten conocer el comportamiento completo del sistema en estudio ante distintas entradas o en diferentes situaciones. Al estudiar el comportamiento estructural se encuentra una extensa literatura tanto para el estudio dinámico como para el régimen estático, recopilándose lo siguiente:

- Respuesta estática: En la ingeniería civil toda estructura se diseña para que se encuentre en reposo cuando actúan sobre esta fuerzas externas, es decir, la estructura en conjunto debe cumplir con las condiciones de equilibrio, siendo la fuerza y el momento resultante sobre esta igual a cero en todo momento. Para describir estas condiciones de equilibrio se cuentan con herramientas

matemáticas que proporcionan las condiciones necesarias para su cumplimiento. Estas ecuaciones permiten la resolución estática de la estructura, la cual permite determinar el valor de todas las incógnitas estáticas de interés (Basset Salom, 2014).

Cuando las fuerzas que actúan sobre la estructura pueden calcularse a partir de las ecuaciones de equilibrio, se tiene una estructura en equilibrio y se denomina estructura estáticamente determinada. En caso de tenerse más fuerzas desconocidas que ecuaciones de equilibrio se habla de una estructura estáticamente indeterminada.

- Rigidez: Uno de los parámetros más importantes dentro de la respuesta estática es la rigidez. Esta se define como la propiedad que tiene un elemento estructural de soportar la deformación o deflexión al estar bajo la acción de una fuerza o carga. Una medida de la rigidez viene dada por el Módulo de Young; esta es una constante del material y es independiente de la cantidad de material.
- Respuesta dinámica: La dinámica estructural se encarga de estudiar el efecto que tienen cargas dinámicas sobre el sistema. La respuesta ante estos eventos, como pueden ser sismos, vientos, equipos mecánicos, paso de vehículos o personas, se denomina respuesta dinámica (Hurtado, 2000). Además, la respuesta dinámica permite caracterizar algunos parámetros de gran interés para estudiar su comportamiento conocidos como parámetros modales. Estos parámetros surgen al estudiar las ecuaciones diferenciales que describen el movimiento de la estructura, partiendo de un modelo idealizado simple de masa concentrada como el de la Figura 2.1.

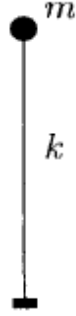


Figura 2.1. Modelo de masa concentrada de 1 grado de libertad (Hurtado, 2000).

La dinámica de este modelo puede describirse utilizando la ecuación diferencial de movimiento:

$$m\ddot{u} + f_R(t) = p(t) \quad (2.1)$$

La ecuación 2.1 se conoce como ecuación de vibración libre sin amortiguamiento. Donde $p(t)$ representa las cargas dinámicas y $f_R(t)$ la fuerza de restitución propia de un material elástico. Esta ecuación es una ecuación diferencial de coeficientes constantes, que consta de una solución homogénea más una solución particular. La solución homogénea será la respuesta de la estructura a la vibración libre, es decir, si la masa de la Figura 2.1 se deja oscilar libremente.

Se sabe que una ecuación de este tipo tendrá una solución como:

$$u = A.\sin\omega t + B.\cos\omega t \quad (2.2)$$

La ecuación 2.2 contiene información relevante para la caracterización dinámica de la estructura. Esta caracterización parte del estudio de los parámetros modales de la misma.

Entre estos parámetros modales se encuentran:

- Frecuencia natural: Toda estructura física tiene asociada una frecuencia de vibración natural. Las máquinas, los puentes, los edificios; todas estas estructuras vibran u oscilan al ser perturbadas o removidas de su estado de reposo inicial. Es una propiedad es intrínseca del sistema y depende de su masa, rigidez y amortiguamiento. Todas tienen al menos una frecuencia natural y es posible que tengan múltiples frecuencias de resonancia (Irvine, 2000).

Se suele calcular la frecuencia natural de resonancia de un sistema libre usando:

$$f = \frac{1}{\sqrt{\frac{k}{m}}} \quad (2.3)$$

- Amortiguamiento: Toda estructura comienza a oscilar una vez es removida de su estado de reposo o equilibrio, sin embargo, ese movimiento no es perpetuo. El amortiguamiento se define como la capacidad de disipación de energía que posee la estructura bajo excitaciones externas. Las soluciones a la ecuación 2.1, al añadir el amortiguamiento de tipo viscoso, arrojan 3 posibles casos:

1. Sistema críticamente amortiguado: El sistema no vibra.
2. Subamortiguado o amortiguado subcrítico: Caso más común por la naturaleza de los materiales utilizados en las estructuras. La respuesta del sistema decae con el tiempo de forma exponencial, como se puede ver en la Figura 2.2.

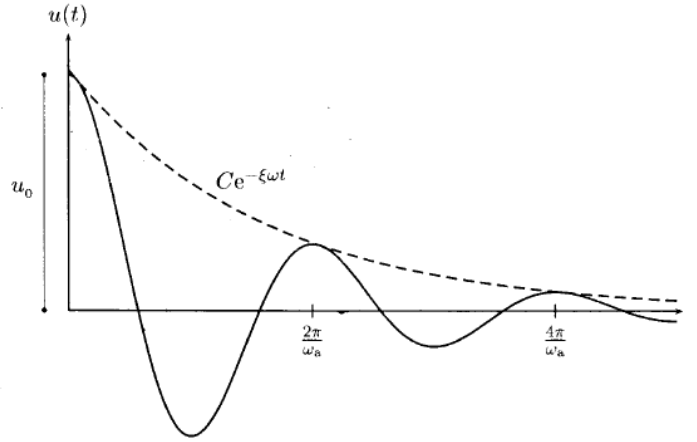


Figura 2.2. Respuesta ante vibración libre en sistema Subamortiguado (Hurtado, 2000).

3. Sobreamortiguado: Nunca se encuentra esta respuesta en sistemas estructurales por los materiales utilizados.

2.1.4. Daño en estructuras

El daño a una estructura civil o mecánica puede definirse como todo cambio en las propiedades materiales o geométricas del material que llegan a afectar de forma adversa la confiabilidad y el desempeño actual o futuro del sistema. Por tanto, el daño es una comparación entre el sistema en cuestión en 2 instantes de tiempo distintos (Farrar y Worden, 2007). Estos efectos adversos pueden ser, en el caso estructural, desplazamientos, estrés indeseado en un elemento o vibraciones estructurales indeseadas (Chen, 2018).

Toda estructura civil, como puentes y edificios, acumulan daño de forma continua a medida que están en servicio y transcurre su vida útil. Este daño puede manifestarse como fracturas, fatiga, socavaciones o desprendimiento del concreto. El daño que no sea detectado puede conducir a una falla estructural que a su vez ocasione pérdidas humanas. Por tanto, es imperativo y necesario detectar el daño

en una estructura tan pronto como sea posible, (Chen, 2018).

Entre algunos de los factores que influyen del deterioro de una estructura se encuentran:

- Proceso de degradación natural de los materiales.
- Corrosión del acero de refuerzo.
- Evento sísmico, incendios o condiciones de guerra.
- Carga por encima del límite de diseño.

Las escalas de tiempo y de extensión del daño son diversas. Por ejemplo, el deterioro por el paso del tiempo bajo ciertas condiciones climáticas es muy lento comparado al daño causado por un evento catastrófico.

2.1.5. Principios de la Sismoresistencia

Una edificación sismorresistente es aquella que está diseñada y construida para soportar las fuerzas causadas por eventos sísmicos. Sin embargo, incluso las edificaciones diseñadas y construidas según las normas sismorresistentes pueden sufrir daños en caso de un terremoto muy fuerte, sin embargo, las normas establecen los requisitos mínimos para proteger la vida de las personas que ocupan la edificación

Algunas de las características de una estructura sismorresistente son:

- Forma regular.
- Bajo peso.
- Mayor rigidez.

- Buena estabilidad.
- Suelo firme y buena cimentación.
- Materiales competentes.
- Capacidad de disipación de energía.
- Fijación de acabados e instalaciones.

En Venezuela las estructuras deben cumplir con la Norma Venezolana COVENIN 1756:2001 (Edificaciones Sismorresistentes).

Se ha observado que al estudiar el comportamiento de las estructuras luego de un evento sísmico, es evidente que cuando se toman en cuenta las normas de diseño sismorresistente dispuestas en la ley y la construcción es debidamente supervisada, los daños estructurales resultan ser considerablemente menores que en las edificaciones en las cuales no se cumplen los requerimientos mínimos indispensables estipulados en la norma, (Blanco, 2012).

Importancia de la instrumentación

La instrumentación estructural permite medir y monitorear las acciones y respuestas estructurales ante distintos eventos. Esto proporciona datos en tiempo real sobre el comportamiento dinámico y estático de la estructura, como deformaciones, aceleraciones y desplazamientos, que son fundamentales para evaluar y verificar si la estructura cumple con los criterios de diseño sismorresistente establecidos en la norma.

La instrumentación estructural ayuda a validar los modelos y suposiciones utilizados en el diseño estructural inicial. Al comparar los datos recopilados por

la instrumentación durante un evento sísmico con las predicciones del modelo, es posible verificar si la estructura se comporta de acuerdo con las expectativas y si cumple con los criterios de seguridad establecidos en la norma.

Además, el monitoreo continuo de la estructura permite conocer el estado actual de la misma, tema que representa la idea principal del Monitoreo de Salud Estructural, permitiendo a los ingenieros evaluar si se sigue cumpliendo con la norma para luego tomar decisiones y actuar en pro de la seguridad de la edificación.

2.2. Salud estructural

2.2.1. Definición

El proceso de implementar una estrategia de identificación de daño para estructuras civiles, mecánicas o aeroespaciales se conoce como Monitoreo de Salud Estructural (SHM por sus siglas en inglés). Esta estrategia requiere medir las condiciones y el ambiente en el que opera la estructura, además de la respuesta de la misma durante un período de tiempo tomando muestras periódicamente espaciadas, (Farrar y Worden, 2007).

La estrategia del SHM requiere de equipos multidisciplinarios de ingeniería, ya que necesita de una red de sensores que midan las variables de interés, el procesamiento y análisis de los datos obtenidos y posteriormente una prognosis del daño para una eventual toma de decisiones. El objetivo del SHM es proveer, en toda la vida útil de la estructura, un diagnóstico del estado de sus materiales constitutivos, de los diferentes elementos que la componen y de la estructura en sí como el conjunto de todas estas partes. Esto para garantizar que la misma se comporte dentro de los parámetros iniciales de diseño, aunque estos cambien por la acción natural del tiempo, el ambiente y accidentes, (Balageas y cols., 2010).

El resultado de este proceso es información actualizada sobre el estado de la estructura y sobre su capacidad actual para seguir desempeñado la función para la cual fue diseñada.

Según Enckell (2006), el SHM se ha convertido en una herramienta muy conocida y utilizada en ingeniería estructural en los últimos años en diferentes países.

2.2.2. Reseña histórica

Las técnicas de detección de daño basadas en vibración tienen sus primeras aplicaciones desde hace cientos de años. En la antigüedad, los constructores golpeaban las estructuras para encontrar espacios vacíos o grietas en elementos de arcilla. La utilidad de estas inspecciones tan simples indicaban que la sofisticación de estos métodos podía proveer información muy valiosa sobre el elemento de interés, sin embargo, esto requiere de instrumentos y herramientas matemáticas que se han desarrollado con el pasar de los años. El auge en el uso de SHM en años recientes es consecuencia de la evolución y miniaturización del hardware computacional actual.

El uso más exitoso del SHM ha sido el monitoreo de la condición de máquinas rotativas, las cuales actualmente han adoptado un enfoque de identificación de daño sin basarse en un modelo de forma casi exclusiva, (Farrar y Worden, 2007).

En los años 70 la industria petrolera consideró el uso de técnicas basadas en vibración para identificar daños en plataformas costa-afuera, este enfoque se diferenció de las máquinas rotativas al estudiar un sistema en donde la ubicación del daño es desconocida y difícil de instrumentar.

En esa misma época, la comunidad aeroespacial y la *National Aeronautics*

Space Agency (NASA), comenzaron a estudiar esta técnica de identificación de daño en los comienzos de la era de lanzamientos espaciales. Este trabajo continúa hoy en día y el *Shuttle Modal Inspection System* (SMIS) se desarrolló para identificar fatiga en distintos componentes de cohetes espaciales reusables, los cuales representan el futuro de esta industria.

Usualmente, los enfoques de estas industrias se basan en comparar modelos analíticos de estructuras sin daño con las mediciones de estructuras con daño, observando principalmente las propiedades modales de las mismas. Se ha observado que cambios en la rigidez en ambos modelos han permitido localizar y cuantificar el daño, (Farrar y Worden, 2007).

Inicialmente, las técnicas no destructivas fueron introducidas en la ingeniería civil a mediados de los años 40, (Abdo, 2014). La necesidad principal surgió en determinar propiedades del concreto fresco *in-situ*. Estas técnicas, que buscaban evaluar la homogeneidad y la resistencia del concreto eran en su mayoría pruebas con martillo y pruebas de *pull-out*. A medida que las estructuras envejecieron, los ingenieros necesitaban idear maneras de medir o estimar las propiedades mecánicas de los elementos que constituyen las estructuras, además de detectar daños que no eran fáciles de observar por la envergadura de las estructuras civiles que se han desarrollado en los últimos 150 años. Es ahí, en los años 70, donde surgen nuevas estrategias no destructivas tales como:

- Emisión acústica.
- Métodos de ultrasonido y radar.
- Termografía.
- Métodos basados en vibración

La comunidad de ingeniería civil ha estudiado la identificación de daño basada en vibración en puentes y edificios desde comienzos de los años 80. Las propiedades modales han sido estudiadas por diferentes autores y son las principales características que se analizan al identificar daño. El auge del SHM es tal, que algunos países asiáticos han implementado regulaciones en donde las compañías constructoras deben verificar la salud estructural de los puentes periódicamente. Estas regulaciones han provocado que la investigación e inversión en esta área siga aumentando de forma considerable, (Chen, 2018).

2.2.3. Línea de trabajo del Monitoreo de Salud Estructural

Los sistemas de SHM consisten de varios elementos que permiten a los ingenieros tener información sobre el estado de una estructura, entre esos elementos se encuentran:

- Sensores.
- Sistemas de adquisición de datos.
- Sistema de transmisión de datos.
- Sistema de procesamiento de datos.
- Sistema de manejo y almacenamiento de datos.
- Equipo de análisis y toma de decisiones.

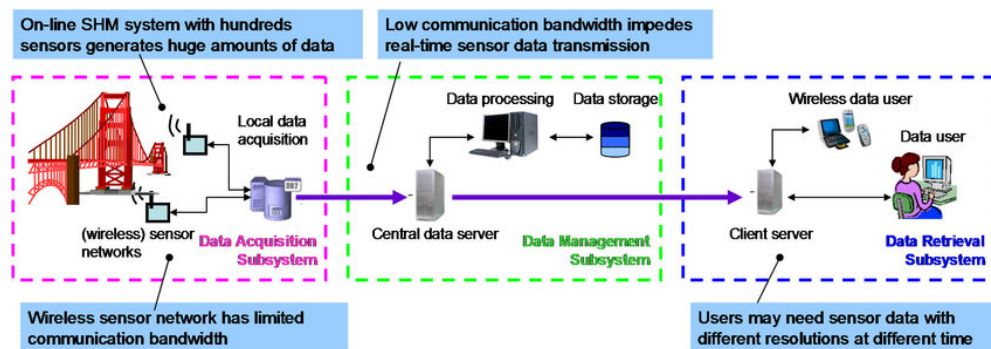


Figura 2.3. Esquema de un sistema de SHM (Li y cols., 2015).

Autores como Rytter (1993) y Farrar y Worden (2007) han esquematizado la estrategia del SHM categorizando el daño en una estructura por niveles de la siguiente forma:

1. Nivel I (detección del daño) ¿Presenta daño el sistema? Es una indicación cualitativa de que puede haber daño presente en la estructura.
2. Nivel II (localización o ubicación del daño) ¿Dónde está presente el daño? Indica la posible localización del mismo.
3. Nivel III (clasificación del daño) ¿Qué tipo de daño está presente? Da información sobre el tipo de daño.
4. Nivel IV (alcance/grado/extensión del daño) ¿Cuál es el alcance del daño? ¿Qué tan grave es? Da un estimado del alcance.
5. Nivel V (prognosis del daño) ¿Cuánta vida útil le queda a la estructura? Da un estimado de la seguridad de la estructura.

En la mayoría de los casos, para alcanzar el nivel final es necesario obtener información sobre los niveles previos. Esto indica que a medida que se sube de nivel se tiene un mayor conocimiento sobre el estado de la estructura.

De acuerdo a Chen (2018), los primeros dos niveles, detección y localización, generalmente pueden alcanzarse usando métodos de detección basados en vibración para obtener mediciones sobre la respuesta dinámica de la estructura.

Por su parte, Chen (2018) describe el proceso de SHM en general como:

1. Observación.
2. Evaluación.
3. Calificación.
4. Gestión.

La estrategia de Monitoreo de Salud Estructural podría resumirse en el siguiente diagrama:

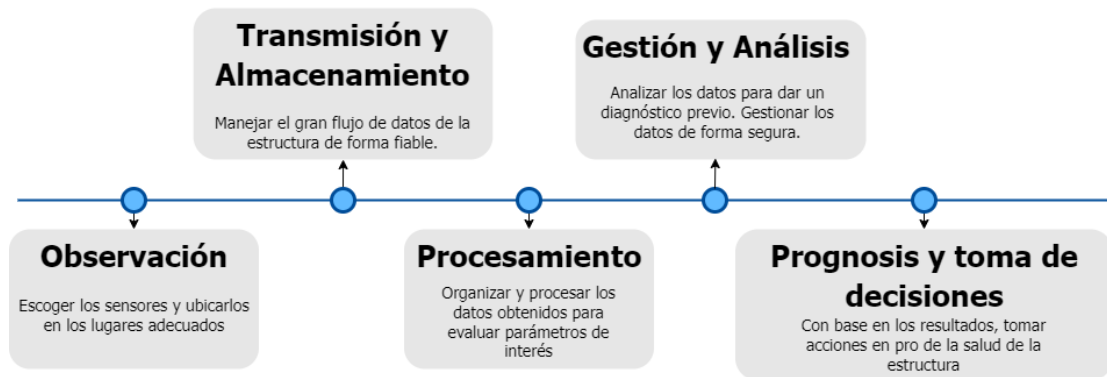


Figura 2.4. Diagrama general del proceso de SHM.

2.2.4. Criterios de evaluación

Como se definió anteriormente, el daño estructural puede tener distintas causas y formas. Lo que se sabe con certeza es que una estructura, una vez entra en

funcionamiento, análogo a los seres humanos al nacer, estará sujeta a envejecimiento natural y a condiciones adversas. Ahora bien, en el caso del SHM, surge la siguiente pregunta ¿Qué se debe medir para poder detectar este daño?.

Numerosos autores concluyen que uno de los indicativos de daño de una estructura viene dado por los parámetros modales definidos anteriormente, frecuencia y amortiguamiento. Esta relación entre los parámetros modales y el daño viene dada por la premisa de que todo daño presente en la estructura se reflejará en un cambio en las propiedades dinámicas de la misma. Worden y Friswell (2009), comprobó la relación entre el cambio progresivo en todas las frecuencias naturales de 15 vigas estudiadas a las cuales se les introdujo un daño relacionado con un cambio del Módulo de Young, el cual, como fue mencionado anteriormente, provee un indicativo de la rigidez de un elemento.

Anteriormente en la ecuación 2.1 se definió un sistema con un grado de libertad (DOF por sus siglas en ingles), sin embargo, en la realidad las estructuras tienen múltiples grados de libertad, por lo que es conveniente modelarlas de esta forma para obtener resultados más precisos. En el caso de los sistemas *n-degrees of freedom* (DOF por sus siglas en ingles), la ecuación de movimiento que describe la dinámica del sistema en vibración libre vendrá dada por:

$$M\ddot{u} + C\dot{u} + Ku = 0 \quad (2.4)$$

Donde M, C y K representan las matrices de masa, amortiguamiento y rigidez de la estructura, respectivamente.

Si se asume un sistema sin amortiguamiento, a fines de estudiar el efecto que tiene sobre la rigidez un cambio en los parámetros modales, de la ecuación 2.4 se obtiene:

$$M\ddot{u} + Ku = 0 \quad (2.5)$$

Si se asume una solución oscilatoria pura, por ser un sistema sin amortiguamiento:

$$u = ve^{j\omega t} \quad (2.6)$$

Al derivar, sustituir y despejar en la ecuación 2.5 se obtiene:

$$(K - \lambda M)\phi = 0 \quad (2.7)$$

Esta ecuación 2.7 representa claramente un problema de autovalores, donde λ representa los autovalores asociados a las frecuencias naturales del sistema y ϕ representa el autovector de desplazamiento.

Si se introduce un pequeño cambio ΔK con perturbaciones similares en los otros parámetros:

$$[(K - \Delta K) - (\lambda - \Delta\lambda)(M - \Delta M)](\phi - \Delta\phi) = 0 \quad (2.8)$$

El daño estructural suele venir asociado a un cambio en la rigidez, más no a cambios en la masa de la estructura, por lo que se asume $\Delta M = 0$, (Hearn y Testa, 1991). A su vez, se tiene que $(K - \lambda M)\phi = 0$. Abdo (2014), Shi, Law, y Zhang (1998) y Hearn y Testa (1991) desarrollan estas ecuaciones obteniendo la siguiente relación:

$$\Delta\lambda = \phi^T \Delta K \phi \quad (2.9)$$

De la ecuación 2.9 se observa que cambios en los autovalores λ que representan las frecuencias naturales, y en los autovectores ϕ (formas modales) están directamente relacionados con cambios en la matriz de rigidez (K) del sistema. De aquí surge el interés en monitorear los parámetros modales como indicadores de daño estructural. Es importante recalcar que estos cambios son indicativos de daño global, más no de la localización del mismo, para lo que se necesitan otras técnicas, (Abdo, 2014).

2.2.5. Variables de interés

Tomando en cuenta la relación entre los parámetros modales y el daño presente en una estructura, es preciso definir las variables de interés para el monitoreo de la salud estructural de una estructura. Si bien existen distintas variables que permiten obtener información valiosa sobre la estructura en estudio, algunas de estas no proporcionan información global del daño, como es el caso de las formas modales y la deflexión local (Rytter, 1993). Sin embargo, estas mediciones proveen indicativos de la ubicación del daño, por lo que pueden constituir parte del sistema de monitoreo en una etapa más avanzada, es decir, una vez el daño fue detectado. A continuación se presentan las más relevantes para el daño global:

- Frecuencias naturales y amortiguamiento: Los parámetros modales de la estructura están ligados de forma directa al estado de la misma. El deterioro en una edificación induce cambios en la rigidez estructural, como se observa claramente en la ecuación 2.9. El daño puede tener efectos distintos en cada modo o cada frecuencia de vibración, por lo que es importante no ubicar

los sensores sobre los nodos modales, ya que experimentos han demostrado la ineficacia en las mediciones. Usualmente, el daño se refleja como una disminución en las frecuencias naturales afectadas, aunque se han observado casos de aumento en las frecuencias de vibración en estructuras de concreto pretensado, (Rytter, 1993).

A su vez, el amortiguamiento varía al introducir daño en la estructura, puesto que su capacidad de disipar energía se ve afectada. Usualmente, los investigadores observan un aumento en el amortiguamiento a medida que el daño aumenta, como se ha demostrado experimentalmente por autores como Hearn y Testa (1991) y Rytter (1993).

- Temperatura y humedad: En los sistemas de monitoreo, la detección del daño estructural puede tomar períodos de tiempo considerables, durante los cuales las características sujetas a temperatura y humedad, sufren cambios que afectan la respuesta estructural.

Es evidente que las condiciones climáticas contribuyen con el deterioro de las edificaciones. A pesar de esta conclusión, relacionar las condiciones climáticas con el daño introducido usando mediciones ambientales es difícil. La medición de estas variables suele tomarse en cuenta para poder cuantificar el cambio que producen estas condiciones en los demás indicadores de daño. Rytter (1993) observó que la humedad y temperatura afectaban las mediciones de amortiguamiento. Por su parte, Abdo (2014), observó que las frecuencias naturales de barras y vigas disminuían a medida que aumentaba la temperatura. A su vez, Sohn (2007) determinó que cuando hay humedad presente en el ambiente, los puentes de hormigón absorben una cantidad considerable de esta, lo que aumenta sus masas y altera sus frecuencias naturales.

- Inclinación y desplazamiento: Una de las variables más comunes en Sistemas de Monitoreo de Salud Estructural es el desplazamiento lineal, que refleja de

forma directa el comportamiento de un elemento o estructura. Esta podría considerarse una variable de tipo estático, ya que a pesar de variar con el tiempo, es decir, depender de un componente dinámico como el viento o la carga vehicular, su interacción con el medio suele ser quasi-estática, causada por la carga estática de la estructura, efectos térmicos y asentamiento.

Los inclinómetros son utilizados en elementos estructurales para medir el desplazamiento lineal de un elemento respecto a una referencia. Suelen ser utilizados para dar una idea de la fijación que presentan los puentes en sus soportes o para monitorear a largo plazo el movimiento de estructuras sobre superficies propensas a deslizamientos.

La deflexión, que puede medirse haciendo uso de inclinómetros los cuales han generado un gran interés en la industria por su sensibilidad y bajo costo, es una variable muy importante a medirse en una estructura. La deflexión puede ser consecuencia de la corrosión de los elementos de soporte, pérdida del pretensado o el crecimiento de grietas en el concreto. Es decir, a medida que la deflexión aumenta se acelera la acumulación de daño, por tanto, medir esta variable puede proveer alertas tempranas sobre posibles cambios estructurales o deterioro (Zhang, Sun, y Sun, 2017).

Autores como Komarizadehasl (2022) destacan la medición de la inclinación en las últimas décadas en el sector de la Ingeniería Estructural, inicialmente con propósitos geotécnicos. Sin embargo, gracias a los avances tecnológicos en los sensores, se comenzó a emplear su uso en otras áreas, sobre todo en el monitoreo de salud estructural.

2.3. Métodos de identificación de daño

2.3.1. Basado en modelo

El método basado en modelo consiste en estructurar un modelo de elemento finito que será utilizado para identificar y localizar el daño en la estructura. Su precisión dependerá de su nivel de correlación con la estructura real, por lo que este método suele emplearse luego de realizar mediciones sobre el sistema y obtener datos experimentales, que permiten ajustar el modelo y disminuir las discrepancias, (García-Fernandez, 2023).

2.3.2. Basado en respuesta

El monitoreo de salud estructural basado en datos hace uso de datos reales sobre la estructura obtenidos a través de mediciones experimentales en sitio. Diferentes arreglos de sensores se utilizan para medir los parámetros de interés para el SHM (Abdo, 2014).

En este método, se comparan los datos obtenidos con los datos históricos de la estructura sin daño. Esta información previa puede ser suministrada por un software de simulación de la estructura basado en sus características iniciales de construcción. Autores como Fritzen (2005) estudiaron el uso de la información modal mediante estudios de vibración ambiental y forzada.

2.4. Ensayos estructurales

Los ensayos estructurales se utilizan para evaluar la integridad y el comportamiento de estructuras. Estos ensayos pueden clasificarse en invasivos y no

invasivos, según la forma en que interactúan con la estructura en estudio. A continuación, se describen brevemente ambos tipos de ensayos:

2.4.1. Ensayos invasivos

- Extracción y análisis de núcleos de concreto: Estos ensayos implican la extracción de muestras de material de la estructura para someterlas a pruebas en laboratorio. Por ejemplo, se pueden realizar pruebas de compresión, tracción o flexión en muestras de concreto, acero u otros materiales utilizados en la construcción. A su vez, los núcleos se someten a distintas condiciones de temperatura y humedad para evaluar su comportamiento.
- Pruebas de carga: Consisten en aplicar cargas controladas a la estructura y medir su respuesta. Esto puede incluir la aplicación de cargas estáticas o dinámicas para evaluar la capacidad de carga, la rigidez y la respuesta estructural.

2.4.2. Ensayos no invasivos

Los ensayos no invasivos se introdujeron en la ingeniería civil durante los años 40 (Abdo, 2014). Desde entonces, muchas de estas pruebas han sido estandarizadas, siendo las normas COVENIN 2221:84 y COVENIN 318-84 algunas de las normativas para este tipo de ensayos en Venezuela.

Con el envejecimiento progresivo de las estructuras, la industria de la construcción ideó nuevas técnicas durante los años 70, permitiendo evaluar de forma más rigurosa las propiedades mecánicas de los materiales y detectar anomalías o daños sin perjudicar la estructura. Algunos de estos métodos se describen a continuación:

- Inspección visual: Estos ensayos se basan en la inspección visual de la estructura para detectar signos de daño, deformaciones o fallas.
- Radiografía: Se utilizan rayos X o rayos gamma para inspeccionar estructuras en busca de defectos internos, como grietas, inclusiones de materiales o corrosión.
- Ultrasonido: Estos ensayos se basan en la inspección visual de la estructura para detectar signos de daño, deformaciones o fallas.
- Análisis de vibración estructural: Los ensayos de vibración permiten describir y evaluar el comportamiento dinámico de la estructura en estudio. Se hace uso de acelerómetros para obtener registros del movimiento estructural ante distintos tipos de vibración y posteriormente se ejecuta el estudio de estos datos en el dominio de la frecuencia, prestando especial interés a las frecuencias de resonancia presentes en el espectro, tanto propias del sistema (modos de vibración), como ajenas al mismo que puedan afectar su funcionamiento.

2.5. Microcontroladores (MCU)

2.5.1. Definición y características

Un microcontrolador es un dispositivo diseñado para controlar tareas específicas en sistemas electrónicos. Usualmente se fabrican utilizando técnicas de miniaturización como VLSI (*Very Large Scale Integration*). Varios tipos de microcontroladores están disponibles en el mercado con diferentes características, clasificándose generalmente por su tamaño en bits (2, 4, 8, 64 y 128 bits). Los microcontroladores encapsulan en un solo chip un microprocesador, memoria, periféricos de entrada/salida y otros componentes necesarios para controlar y

ejecutar tareas específicas. Esencialmente, es un pequeño ordenador en un solo circuito integrado.

Los microcontroladores se utilizan en una amplia variedad de aplicaciones donde se requiere control y procesamiento de datos en tiempo real. Pueden encontrarse en dispositivos electrónicos como electrodomésticos, automóviles, equipos médicos, sistemas de seguridad, robots, entre otros.

Entre los elementos que componen un microcontrolador se encuentran:

- Unidad Central de Procesamiento.
- Memoria RAM y ROM.
- Periféricos.
- Convertidores ADC y DAC.
- Temporizadores.
- Interfaces de comunicación.
- Real Time Clock.

En la figura 2.5 se observa el diagrama de bloques de las partes que constituyen un microcontrolador del fabricante Espressif.

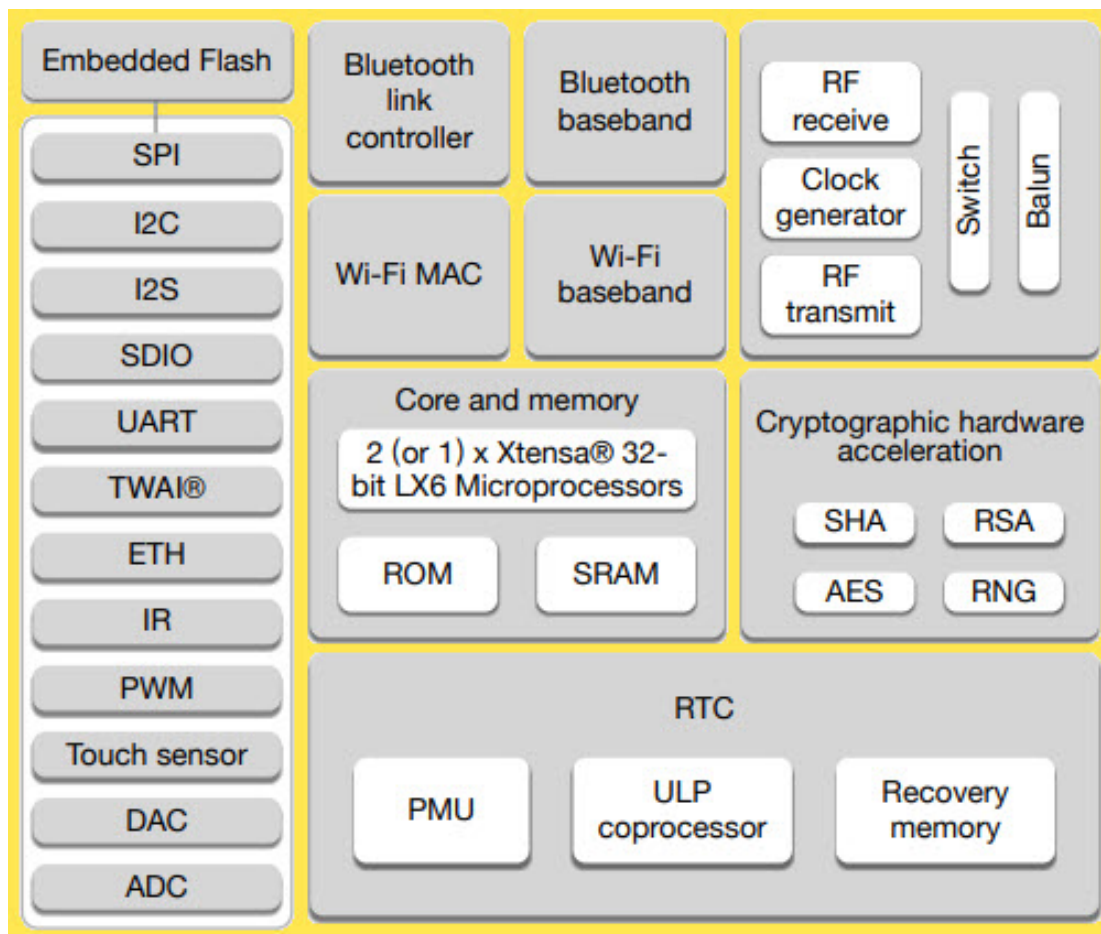


Figura 2.5. Esquema general del microcontrolador ESP32 (Espressif).

2.5.2. Placas de desarrollo

Actualmente existen numerosas formas de hacer uso de las bondades que ofrecen los microcontroladores, especialmente al utilizar las distintas placas de desarrollo que se han diseñado para facilitar su programación e interfaz, ofreciendo la capacidad de comunicarse con un computador de forma sencilla a través del puerto USB, brindando distintas funcionalidades a los periféricos y facilitando la alimentación al microcontrolador. Estas tarjetas han logrado disminuir el costo del uso de los microcontroladores, especialmente para la creación de prototipos que

permiten evaluar la factibilidad de un proyecto antes de diseñar la placa de circuito impreso (PCB por sus siglas en inglés) con un microcontrolador pre-programado.

Las placas de desarrollo se pueden subdividir en:

- *System-on-Chip* (SoC).
- Basadas en microcontrolador (MCU-based).
- *Single-Board-Computer* (SBC).

Algunas de los fabricantes de tarjetas de desarrollo más reconocidos que cuentan con comunidades y soporte técnico son:

- STMicroelectronics.
- Espressif.
- Texas Instruments.
- Nordic Semiconductors.
- NXP
- Arduino.
- Raspberry Pi.

2.5.3. Protocolos de comunicación

Los protocolos de comunicación fueron desarrollados y estandarizados para que sensores, drivers y diferentes periféricos y dispositivos de entrada y salida

pudieran comunicarse entre sí sin problemas, evitando la tediosa tarea de buscar dispositivos compatibles dependiendo de la aplicación y el fabricante a utilizar.

Cada protocolo tiene sus características y aplicaciones, siendo diseñados para propósitos distintos. El desarrollador o diseñador puede escoger uno o más dependiendo de las especificaciones del producto a diseñar. A continuación se mencionan los más utilizados por los microcontroladores:

- UART (*Universal Asynchronous Receiver-Transmitter*): Uno de los protocolos más simples y comúnmente utilizado para comunicación entre dos dispositivos. Como su nombre lo indica, no utiliza una señal de reloj para sincronizar al dispositivo transmisor y receptor. Es por esto que, a diferencia de la mayoría de los protocolos, solo utiliza 2 cables para lograr la comunicación.

UART requiere de bits de inicio y parada para detectar data entrante, además de requerir que la velocidad de comunicación sea configurada al mismo valor en bits por segundo (bps) en el receptor y transmisor. Admite comunicaciones Simplex, Half-Duplex y Full-Duplex.

- I2C (*Inter-Integrated Communication*): El protocolo I2C fue desarrollado por Philips Semiconductors en 1982. Se utiliza principalmente para comunicación entre periféricos, sensores y circuitos integrados en distancias cortas. Requiere 2 cables de datos (SDA y SCL) para el envío de la data serial. Este protocolo utiliza un bus de reloj para sincronizarse y verificar que la data se envíe y reciba correctamente.
- SPI (*Serial Peripheral Interface*): Desarrollado por Motorola en los años 80. Es un protocolo de un solo maestro, varios esclavos. Requiere una señal de reloj para la sincronización entre los dispositivos esclavos y el maestro. Siendo una comunicación Full-Duplex, puede enviar y recibir datos de forma

simultánea. Requiere 4 cables para la comunicación serial.

- USB (*Universal Serial Bus*): El protocolo USB es el más popular dentro de los protocolos seriales, con un uso extendido en el mundo computacional y electrónico. No requiere señal de clock siendo un protocolo serial asíncrono. Suele ser el estándar para conectar periféricos.

2.5.4. Sistemas Operativos en Tiempo Real (RTOS)

Un sistema operativo en tiempo real (RTOS por sus siglas en inglés), es un sistema operativo diseñado para manejar tareas que requieran ser monitoreadas de forma precisa y eficiente, usualmente estas tareas son críticas en el tiempo, lo que lo diferencia de un sistema operativo de propósito general, el cual se especializa en ejecutar varias tareas al mismo tiempo. El RTOS se concentra en ejecutar las tareas en tiempo real.

El primer RTOS fue un programa desarrollado en la Universidad de Cambridge en los años 60. Este sistema permitía que varios procesos se llevaran a cabo bajo restricciones de tiempo precisas. Desde entonces, numerosos avances en la industria han hecho crecer la demanda por sistemas operativos en tiempo real, siendo utilizados en aplicaciones espaciales, médicas, militares y entre otras.

Entre las características principales de un RTOS se encuentran:

- Planificador (Scheduler): Administra el tiempo de CPU de cada tarea y determina qué tarea debe ejecutar en qué momento. Es el encargo de que el sistema sea determinístico.
- Despachador (Dispatcher): Provee o cede el control del CPU a una tarea. También se encarga de guardar el contexto de la tarea que se ejecuta para

seguir su ejecución la próxima vez que esta se utilice.

- Multiprocesamiento simétrico: El RTOS es el sistema operativo ideal para una arquitectura de multiprocesamiento simétrico (SMP por sus siglas en inglés), en la cual cada núcleo de un procesador tiene su propio sistema operativo y tiene la capacidad de acceder a la lista de tareas disponibles. Todos los núcleos deben tener la misma arquitectura, lo que lo diferencia del multiprocesamiento asimétrico.

FreeRTOS

FreeRTOS es una clase de RTOS diseñado para ser ejecutado en microcontroladores. A pesar de ser un kernel muy liviano, su uso se ha extendido más allá de los sistemas embebidos. Es distribuido de forma libre bajo la licencia MIT de software libre.

En aplicaciones basadas en microcontroladores rara vez se requiere una implementación completa de un RTOS, por las restricciones de memoria de los mismos. Es por esto que FreeRTOS implementó un sistema enfocado en la sincronización, funcionalidades de comunicación entre tareas y planificación de tareas. Este RTOS incluye además distintas librerías que expanden sus funciones y mejoran la experiencia de uso.

Los principales elementos de un programa basado en FreeRTOS contienen:

- Colas (*Queues*): Las colas son el método principal de comunicación entre tareas. Pueden enviar datos de una tarea a otra o entre interrupciones y tareas. Suelen utilizarse como buffers FIFO (First-In-First-Out).
- Tareas (*Tasks*): Las tareas son las funciones que el sistema operativo en

tiempo real ejecutará una vez el scheduler inicie el ciclo de programa. En ellas se contienen las distintas acciones que se quieren implementar en el microcontrolador. No tiene valores de retorno y se implementan como un loop infinito. En la figura 2.6 se observan los distintos estados en los que puede encontrarse una tarea de FreeRTOS. Estas distintas posibilidades son las que permiten controlar el orden de ejecución de un programa de forma sincronizada. La ejecución de las tareas está regida por su prioridad y estas deben ser creadas antes de poderse ejecutar.

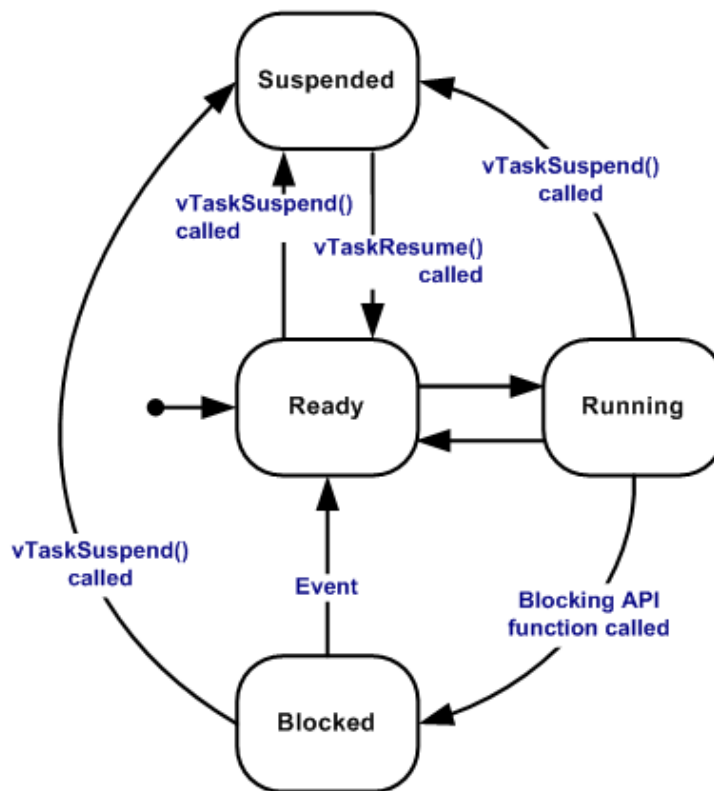


Figura 2.6. Estados de las tareas en FreeRTOS (FreeRTOS, 2024).

- Semáforos binarios (*Binary Semaphores*): La idea general de los semáforos binarios es permitir que las tareas accedan a recursos compartidos y ejecutar tareas complejas de sincronización. Suelen usarse en conjunto con los mutex (que viene del término *mutual exclusion*), lo cual evita que se corrompan

sectores de memoria cuando 2 tareas (que pueden estar ejecutándose en paralelo) quieran acceder a un espacio de memoria al mismo tiempo.

La arquitectura de un sistema basado en FreeRTOS se muestra en la figura 2.7:

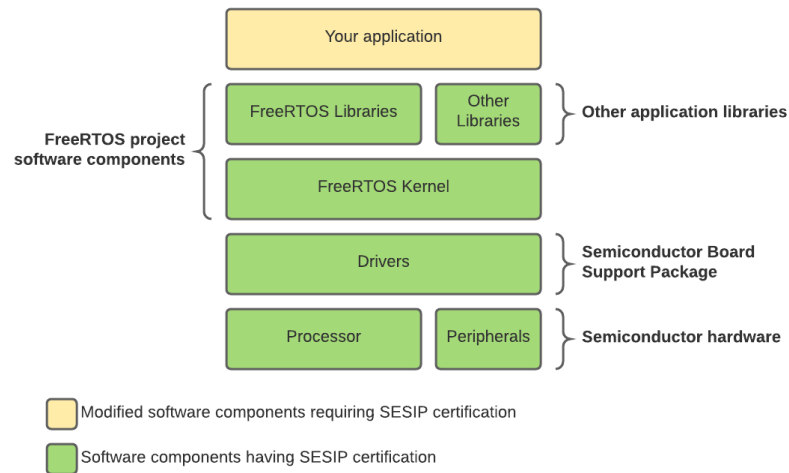


Figura 2.7. Arquitectura de sistema embebido basado en FreeRTOS (FreeRTOS, 2024).

2.6. Sensores

2.6.1. Definición

Los sensores son dispositivos capaces de detectar un parámetro físico y responder a la entrada de este fenómeno físico con una señal, usualmente eléctrica, que puede ser medida y transmitida. Usualmente se clasifican según la variable a medir, por el tipo de señal que emiten (digital o analógica, en el caso de señales eléctricas) y por su interacción con la variable a medir, siendo los activos los que generan una señal para obtener la medición y los pasivos los que responden a los cambios del medio físico. Son la base principal de todo sistema de instrumentación

al representar el vínculo entre los variables físicas que se desean medir y el ser humano, cuyas capacidades sensoriales no están diseñadas para medir ciertas variables y mucho menos tomar valores y registros de las mismas.

2.6.2. Características de los sensores

- Sensibilidad: Razón entre el incremento en la señal de salida del sensor y la variación en la variable medida.
- Resolución: Representa la variación mínima de la variable a medir que puede ser detectada por el sensor.
- Rango: Representa el conjunto de valores comprendidos entre el límite superior e inferior de la capacidad del instrumento de medición. Este espectro suele representarse mediante los valores extremos.
- Temperatura de operación: Rango de temperaturas dentro del cual el comportamiento del sensor es el esperado dentro de un límite de error.
- Frecuencia de respuesta o ancho de banda: Rango de valores en el espectro de frecuencia dentro de los cuales los sensores son capaces de obtener mediciones de la variable de interés. Esta característica es particularmente importante en ensayos de vibración y ensayos dinámicos.

2.6.3. Sensores de interés para el Monitoreo de Salud Estructural

En el campo de la monitorización de salud estructural (SHM), existen varios tipos de sensores que se utilizan comúnmente. Algunos de los sensores más utilizados en SHM son los siguientes:

- Deformación: Estos sensores miden la deformación o la tensión en la estructura. Pueden incluir galgas extensiométricas, sensores de fibra óptica o sensores de cable extensométrico.
- Temperatura: Estos sensores miden la temperatura ambiente o la temperatura de la estructura. Son útiles para evaluar el efecto del calor en el comportamiento estructural y detectar anomalías relacionadas con cambios térmicos. Se suelen utilizar los siguientes tipos:
 - Termopares: consisten en dos cables de diferente material metálico unidos en un extremo. La diferencia de temperatura entre los extremos genera una pequeña corriente eléctrica que se puede medir y relacionar con la temperatura.

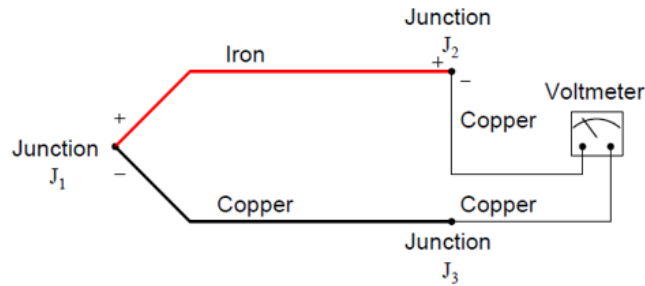


Figura 2.8. Esquema general de funcionamiento de un termopar (Dunn, 2005).

- Termistores: Utilizan materiales semiconductores cuya resistencia eléctrica cambia con la temperatura.
- Microelectromecánicos (MEMS): Suelen utilizar el efecto termoeléctrico o la variación de las propiedades eléctricas de los materiales con la temperatura para medir y cuantificar los cambios térmicos.

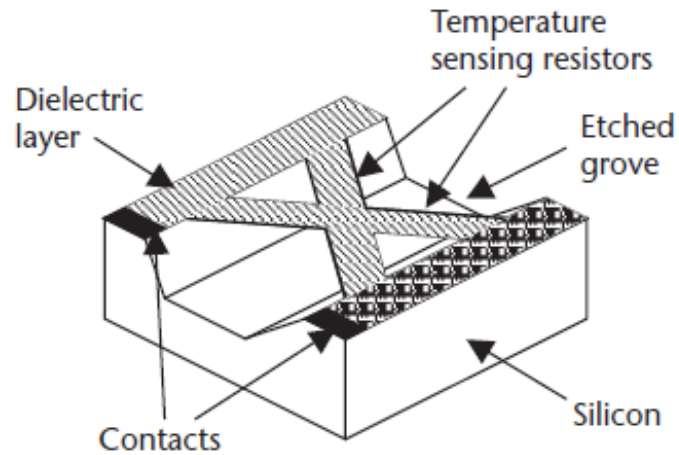


Figura 2.9. Esquema general de un sensor de temperatura de tecnología MEMS (Dunn, 2005).

- Humedad: Los sensores de humedad se utilizan para medir la humedad relativa o la presencia de agua en la estructura. Son particularmente útiles para evaluar el grado de corrosión en estructuras metálicas.
- Capacitivos: utilizan la capacidad dieléctrica de un material sensible a la humedad para medir el contenido de agua en el aire o en el suelo. El cambio en la capacidad dieléctrica se traduce en una señal de salida proporcional a la humedad relativa.

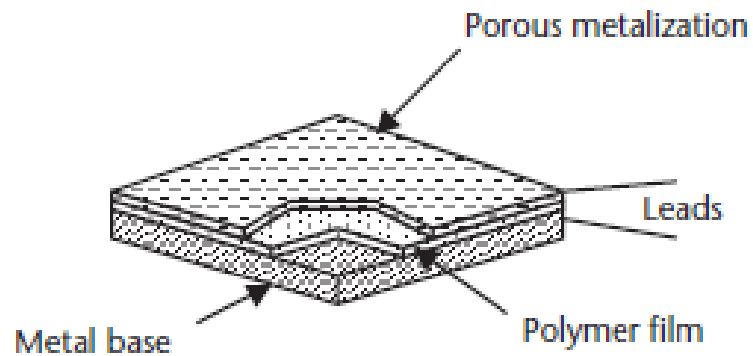


Figura 2.10. Esquema general de un sensor de humedad capacitivo (Dunn, 2005).

- Resistivos: Utilizan un material sensible a la humedad, como una película polimérica o un material cerámico, cuya resistencia eléctrica varía con la humedad.

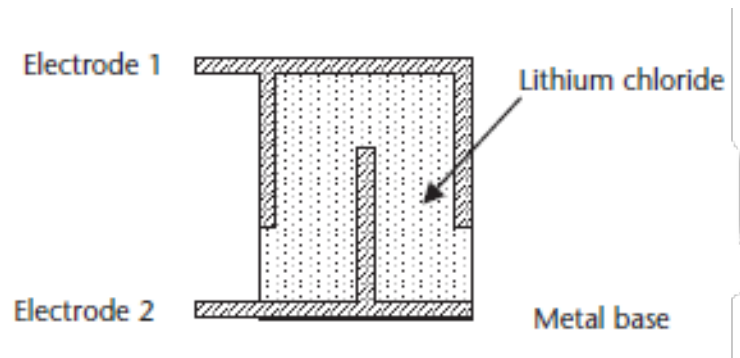


Figura 2.11. Esquema general de un sensor de humedad resistivo (Dunn, 2005).

- Microelectromecánicos (MEMS): suelen utilizar una capa de material sensible a la humedad, como un polímero higroscópico, que experimenta cambios en sus propiedades físicas, eléctricas o capacitivas en respuesta a la humedad del ambiente.

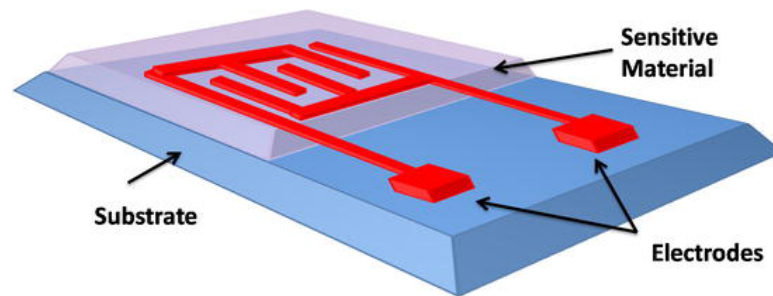


Figura 2.12. Esquema general de un sensor de humedad de tecnología MEMS (Alfaifi y Zaman, 2021).

- Acelerómetros:
 - Piezoeléctrico: Se basan en el principio piezoeléctrico, donde un cristal piezoeléctrico genera una carga eléctrica proporcional a la aceleración aplicada. Son sensibles, ofrecen una amplia respuesta en frecuencia y son adecuados para medir tanto bajas como altas frecuencias de vibración.

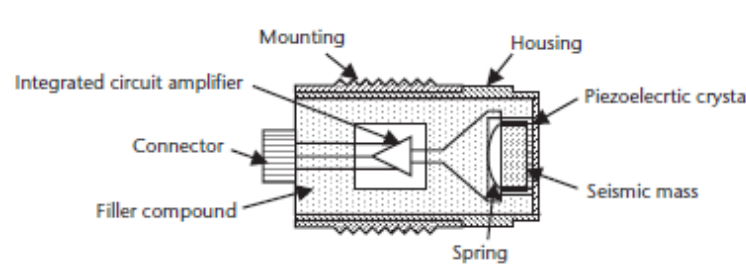


Figura 2.13. Esquema general de un acelerómetro piezoeléctrico (Dunn, 2005).

- Capacitancia Diferencial: utilizan un conjunto de placas móviles y fijas separadas por una pequeña distancia. La aceleración causa un cambio en la distancia entre las placas, lo que a su vez modifica la capacitancia del dispositivo.
- Microelectromecánicos (MEMS): Se basan en el desplazamiento de una masa suspendida mediante resortes micromecánicos. Son pequeños, económicos y se utilizan ampliamente en aplicaciones de monitoreo

estructural debido a su tamaño compacto y bajo consumo de energía.

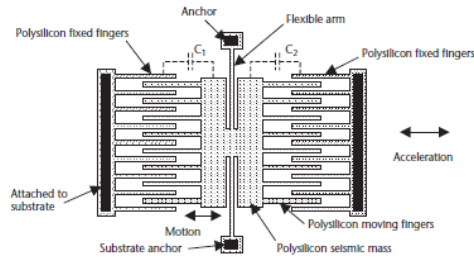


Figura 2.14. Esquema general de un acelerómetro de tecnología MEMS (Dunn, 2005).

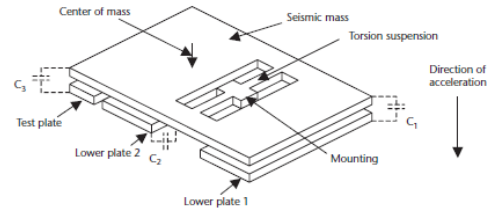


Figura 2.15. Acelerómetro microelectromecánico (Dunn, 2005).

2.6.4. Fusión de sensores

La fusión de sensores consiste en combinar la información de múltiples sensores para obtener una medida más precisa y confiable de la variable de interés (Lamkin-Kennard y Popovic, 2019). La fusión de sensores puede mejorar la precisión, la confiabilidad y la robustez de las mediciones, ayudando también a reducir la incertidumbre y el ruido presente en los datos. La fusión de sensores se puede realizar de varias maneras, las cuales se mencionan a continuación:

- Sensores redundantes: Todos los sensores proporcionan la misma información sobre la variable de interés. La redundancia de sensores puede mejorar la confiabilidad y la precisión de las mediciones, ya que si un sensor falla, los otros sensores pueden seguir proporcionando datos.
- Sensores complementarios: Los sensores proporcionan información independiente o complementaria sobre la variable de interés. Al combinar la información de sensores complementarios, se puede obtener una medida más precisa y completa de la variable de interés.

- **Sensores coordinados:** Los sensores recopilan información sobre la variable de interés de forma secuencial.

La fusión de sensores permite obtener información a partir de varios sensores que no está disponible a través de un sensor en específico, siendo necesario recopilar distintas variables físicas para reconstruir la variable de interés. La fusión de sensores es una técnica comúnmente utilizada en aplicaciones de monitoreo estructural, robótica, navegación y sistemas de control, entre otros.

Estimación de ángulos

Una de las variables físicas que suele obtenerse a partir de la fusión de información de varios sensores es la inclinación. Existen distintos métodos, cada uno con sus ventajas y deficiencias, para obtener una estimación de la inclinación de un sistema a partir de mediciones de acelerómetros, giróscopos y magnetómetros. Algunos de los métodos más comunes son:

- **Integración de la medición del giróscopo:** Se utiliza la medición de la velocidad angular del giróscopo para estimar la orientación del sistema. Sin embargo, la integración de la velocidad angular puede acumular errores y derivar en una estimación incorrecta de la orientación.
- **Estimación por trigonometría:** Se utilizan las mediciones de los acelerómetros y magnetómetros para estimar la orientación del sistema a partir de cálculos trigonométricos. Este método puede ser sensible a errores en las mediciones y a la presencia de campos magnéticos externos.
- **Filtro de Kalman:** El filtro de Kalman es un algoritmo de estimación que combina mediciones de sensores con un modelo dinámico del sistema para

obtener una estimación precisa de la orientación. El filtro de Kalman es particularmente útil para fusionar mediciones de acelerómetros, giróscopos y magnetómetros y estimar la orientación en tiempo real.

- Filtro de Madgwick: El filtro de Madgwick es un algoritmo de fusión de sensores que utiliza mediciones de giroscopios, acelerómetros y magnetómetros para estimar la orientación del sistema. El filtro de Madgwick es una versión simplificada del filtro de Kalman y es ampliamente utilizado en aplicaciones de realidad virtual, aumentada y sistemas de navegación inercial.

Este método, desarrollado por Madgwick y cols. (2010), hace uso de cuaterniones para la representación espacial. Estos son una extensión de los números complejos y se utilizan para representar rotaciones en el espacio tridimensional.

Las representaciones tridimensionales traen problemas como el bloqueo de cardán, que consiste en la pérdida de un grado de libertad en la representación de los ángulos de Euler.

Un cuaternión consta de cuatro componentes:

1. Parte escalar (w).
2. Tres partes vectoriales (x, y, z).

Se representan de la siguiente forma:

$$q = w + xi + yj + zk \quad (2.10)$$

El filtro emplea esta representación para evitar los problemas asociados con la representación tridimensional de los ángulos de Euler, permitiendo que los datos de aceleración y magnetómetro se utilicen en un algoritmo de descenso de gradiente derivado de forma analítica y optimizado para

calcular la dirección del error de medición del giroscopio como una derivada de cuaternión.

El algoritmo utiliza una combinación de la información proporcionada por los giroscopios (para estimar la velocidad angular) y los acelerómetros y magnetómetros (para estimar la gravedad y el vector magnético terrestre) para calcular la orientación.

2.7. Sensores inteligentes

2.7.1. Definición:

Si bien los sensores por sí mismos han permitido un rápido y sostenido desarrollo en todos los ámbitos de la instrumentación en distintas áreas de la industria, el advenimiento de la era digital y la miniaturización de los componentes ha impulsado cambios en la manera en la que se mide hoy en día. Con el poder de cómputo actual, los sensores aumentan sus capacidades cuando son parte de un sistema más grande capaz de procesar la información que estos proveen.

Un sensor inteligente es un sistema en donde uno o varios sensores y una interfaz electrónica dedicada trabajan en conjunto. La tecnología empleada para llevar a cabo estos sistemas ha avanzado rápidamente en años recientes, lográndose obtener sistemas de un solo chip para medir temperatura o campo magnético (Nagayama, 2007).

El término sensor inteligente o “*smart sensor*” surgió a mediados de los años 80. La “inteligencia” de estos dispositivos viene dada por un microcontrolador (MCU), un procesador de señales digitales (DSP) y circuitos de aplicación específica (Frank, 2002). Posteriormente, en el año 1998, el *IEEE* en su estándar *Networked smart transducer interface standard 1451* definió un “*smart sensor*”

como un transductor que provee funciones más allá de las necesarias para generar y representar una cantidad medida.

2.7.2. Características:

Un sensor inteligente suele tener 5 características esenciales, las cuales se resumen a continuación:

- **Microprocesador integrado:** El microprocesador es lo que diferencia un sensor común de un sensor inteligente, siendo este el encargado de hacer los cálculos necesarios, guardar los datos de forma local, enviar resultados y coordinar las tareas a ejecutarse.
- **Capacidades de medición:** La principal característica de un sensor es su capacidad de convertir una variable física en algún tipo de señal que pueda ser posteriormente procesada, en este caso por el microprocesador integrado, siendo el sensor el vínculo entre el procesador y los fenómenos físicos de interés. Un sensor inteligente, también llamado nodo, puede constar de varios sensores que miden distintas variables.
- **Comunicación inalámbrica:** En gran medida, se busca que los sensores inteligentes sean capaces de reemplazar los actuales sistemas de medición cableados, permitiendo esto la expansión del sistema de medición a un bajo costo, además de las capacidades de transmisión que ofrecen alternativas como laser, infrarrojo, o incluso ondas de radio y telefonía.
- **Alimentación autónoma:** Con frecuencia los sensores inteligentes son alimentados de forma autónoma como baterías, en particular en aplicaciones de difícil acceso, como es el caso de algunas estructuras civiles. Si bien esto representa un desafío, la logística y los requerimientos asociados a conectar

todo un sistema mallado de sensores a la red eléctrica atenta en contra de la facilidad de instalación y mantenimiento de un sistema inteligente.

- Bajo costo: Una de las características más importantes que se busca obtener al hacer uso de sistemas basados en sensores inteligentes es reducir los distintos costos generados por un sistema basado en sensores comunes. Tecnologías como MEMS (*Micro electro-mechanical systems*) han logrado reducir de forma considerable el tamaño y el costo por su producción en masa basada en semiconductores. Esta característica permite que redes densas de sensores sean implementadas en estructuras civiles.

2.8. Monitoreo

El monitoreo se refiere a la observación continua de un proceso, medición o medición para obtener información sobre su estatus y comportamiento. Estos datos pueden ser procesados y analizados posteriormente para detectar cambios, anomalías, fallas y tomar decisiones de control. Se puede subdividir en dos enfoques:

2.8.1. Monitoreo cableado

Involucra el uso de cables que comunican los dispositivos de adquisición o medición y el sistema central de procesamiento o almacenamiento. La información viaja a través de estos cables que pueden ser coaxiales, fibra óptica, de par trenzado, entre otros.

2.8.2. Monitoreo inalámbrico

Involucra el uso de redes de comunicaciones inalámbricas para el envío de los datos adquiridos por los dispositivos de medición. Los sensores deben contar con un transmisor de datos, enviando los datos en forma de ondas electromagnéticas hasta el receptor, el cual debe encargarse de recibir y decodificar estas señales en la estación central. Algunas de las tecnologías más utilizadas para el monitoreo inalámbrico son WiFi, Bluetooth, Zigbee, LoRa y la red celular.

Tabla 2.1. Ventajas y desventajas de los enfoques de monitoreo.

Monitoreo	Ventajas	Desventajas
Cableado	<ul style="list-style-type: none">■ Confiabilidad.■ Seguridad.■ Gran ancho de banda.	<ul style="list-style-type: none">■ Movilidad limitada■ Instalación compleja■ Costo elevado.
Inalámbrico	<ul style="list-style-type: none">■ Flexibilidad y movilidad.■ Fácil instalación■ Escalabilidad y bajo costo.	<ul style="list-style-type: none">■ Complejidad elevada.■ Posibilidad de interferencia.■ Ancho de banda limitado.

2.9. Sistemas de adquisición de datos

2.9.1. Definición y esquema general

Un sistema de adquisición de datos o DAQ (textitData acquisition systems) consiste en un sistema que incluye sensores, un computador o procesador y software de adquisición de datos integrado. Es capaz de obtener, almacenar, visualizar y procesar los datos de interés. Fueron diseñados con el fin de obtener información sobre algún fenómeno físico aislado o parte de un proceso o sistema más grande, siendo la interfaz entre las variables físicas y el monitoreo del sistema en cuestión.

2.9.2. Acondicionamiento de señales

Las señales obtenidas por el DAQ presentan ruido y pueden verse atenuadas por el canal de comunicación que comunica los sensores con el dispositivo de almacenamiento y procesamiento, es por esto que es preciso filtrar y amplificar la señal previo a su entrada al sistema de adquisición, o en su defecto, previo a su almacenamiento y procesamiento.

2.9.3. DAQ basados en MCU

Dadas las capacidades de procesamiento y comunicación de los microcontroladores en la actualidad, es común ver sistemas de adquisición de datos basados en microcontroladores, haciendo uso de sus periféricos y leyendo el voltaje de los sensores usando el ADC integrado en los mismos, o a su vez, interpretando las señales digitales provenientes de sensores que utilizan protocolos de comunicación serial.

2.10. Sistemas de comunicaciones inalámbricos

2.10.1. Características y conceptos básicos

Un sistema de comunicaciones inalámbrico es un sistema de transmisión de información que utiliza ondas electromagnéticas para transmitir datos sin la necesidad de cables físicos. En lugar de utilizar conexiones alámbricas, se emplean tecnologías de radio, microondas, infrarrojos u otras frecuencias electromagnéticas para la transmisión de datos de manera inalámbrica. Los elementos principales de un sistema de comunicaciones se describen a continuación:

- **Transmisor:** Se refiere al dispositivo o sistema que se encarga de convertir una señal o información en una forma adecuada para su transmisión a través de un medio de comunicación. El transmisor realiza procesos de modulación y amplificación de la señal para adaptarla a las características del canal de transmisión del sistema.
- **Canal de transmisión:** Es el medio físico o inalámbrico a través del cual se envía la señal o información desde el transmisor al receptor. En el caso inalámbrico, este canal es el aire. Puede introducir distorsiones, atenuación, retardos o interferencias en la señal transmitida.
- **Receptor:** Dentro de un canal de comunicaciones, es el dispositivo o sistema que recibe la señal transmitida a través del canal de transmisión y la convierte nuevamente en una forma adecuada para su interpretación o uso. El receptor realiza procesos de demodulación, filtrado y amplificación de la señal recibida para recuperar la información original.
- **Interferencias:** Las interferencias se refieren a señales o perturbaciones no deseadas que afectan la señal transmitida en el canal de comunicación.

Estas interferencias pueden ser causadas por fuentes externas, como señales electromagnéticas de otros dispositivos o fenómenos naturales, o por interferencias internas generadas por componentes electrónicos o ruido en el propio sistema de comunicación. Las interferencias pueden degradar la calidad de la señal, causar errores en la transmisión o incluso impedir la correcta recepción de la información.

- **Ruido:** Puede definirse como una señal o perturbación no deseada que se superpone a la señal útil durante la transmisión o recepción de datos. El ruido puede ser generado internamente en los componentes electrónicos, introducido por fuentes externas o ser el resultado de las características inherentes del canal de transmisión.

2.10.2. Protocolos de comunicación para distancias cortas

- **WiFi:** Es el protocolo de comunicación inalámbrica más popular. Consiste en una tecnología inalámbrica que hace uso del estandar 802.11 del IEEE a través de frecuencias como 2.4 GHz y 5 GHz. Suele tener un rango entre 20 y 40 metros y se caracteriza por velocidades de transmisión muy altas, de hasta 600 Mbps.
- **Bluetooth:** Bluetooth, uno de los medios de comunicación más utilizados en la actualidad, es una tecnología utilizada para el intercambio de datos a distancias cortas (con un rango entre 50 y 100 metros) que hace uso de frecuencias entre 2.4 a 2.485 GHz. Tiene una velocidad máxima de 1 Mbps y es ampliamente utilizado para transferencia de datos entre dispositivos móviles.
- **Zigbee:** Zigbee es un estándar de comunicación inalámbrica de corto alcance y bajo consumo de energía diseñado para aplicaciones de Internet de las

cosas (IoT). Utiliza la tecnología de radio de baja potencia y bajo consumo para permitir la comunicación inalámbrica entre dispositivos Zigbee, como sensores, actuadores y controladores. Zigbee opera en las bandas de frecuencia de 2.4 GHz, 900 MHz u 868 MHz y se basa en una topología de malla, lo que significa que los dispositivos pueden comunicarse directamente entre sí o a través de otros dispositivos Zigbee cercanos.

- MQTT (Message Queuing Telemetry Transport): MQTT es un protocolo ligero basado en el modelo “publish/subscribe”, diseñado por ingenieros de la IBM y Arcom Systems específicamente para aplicaciones del internet de las cosas (Internet of Things), requiriendo poco ancho de banda y siendo fiable en redes inestables. Consiste en un broker, encargado de gestionar los datos, al cual se conectan los clientes, los cuales pueden publicar (enviar datos), o suscribirse a un tópico del broker (acceder a los datos que se publicaron a ese tópico desde algún cliente).

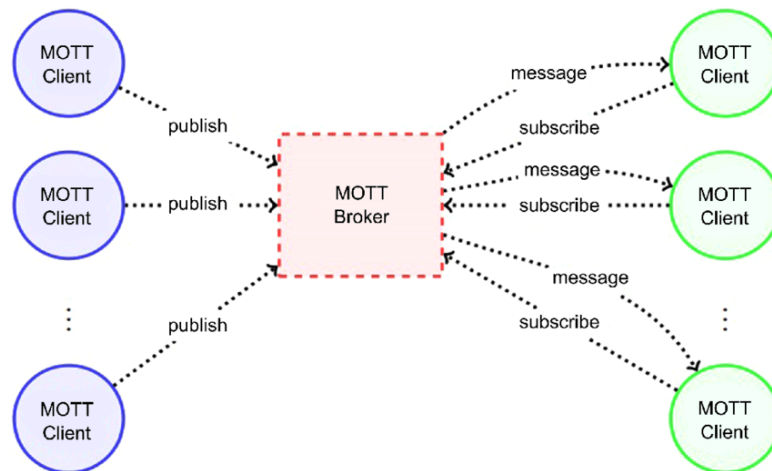


Figura 2.16. Esquema general del protocolo MQTT (Aloufi y cols., 2020).

2.10.3. Protocolos de comunicación de larga distancia

- NB-IoT: NB-IoT (Narrowband Internet of Things) es un estándar de comunicación de larga distancia y bajo consumo de energía para dispositivos IoT. Utiliza la infraestructura de redes celulares existentes y opera en bandas de frecuencia estrechas para proporcionar conectividad de bajo ancho de banda y larga duración de la batería.
- Sigfox: Es una tecnología de comunicación de larga distancia y bajo consumo de energía diseñada para el IoT. Utiliza una red de bajo consumo de energía y bajo costo para transmitir pequeñas cantidades de datos de forma eficiente. Sigfox es utilizado en aplicaciones de seguimiento de activos, monitoreo ambiental y aplicaciones industriales.
- LoRa: LoRa (Long Range) es una tecnología de comunicación inalámbrica de largo alcance y baja potencia diseñada para aplicaciones de IoT y M2M (*machine-to-machine*). Se basa en el estándar LoRaWAN (*LoRa Wide Area Network*) y utiliza una modulación de espectro ensanchado para lograr una mayor cobertura y capacidad de penetración en comparación con otras tecnologías inalámbricas. LoRa opera en las bandas de frecuencia libre de licencia y puede proporcionar alcances de varios kilómetros en áreas urbanas y aún mayores en áreas rurales. Debido a su bajo consumo de energía, es adecuado para dispositivos de batería de larga duración y aplicaciones de bajo ancho de banda.

Dependiendo de la aplicación se debe escoger la tecnología más conveniente. Esto dependerá de la distancia entre los dispositivos que recolectan datos y la estación base, la naturaleza de los datos a enviar y las restricciones de tiempo impuestas por el sistema en estudio. A continuación se presenta un gráfico comparativo entre las distintas tecnologías:

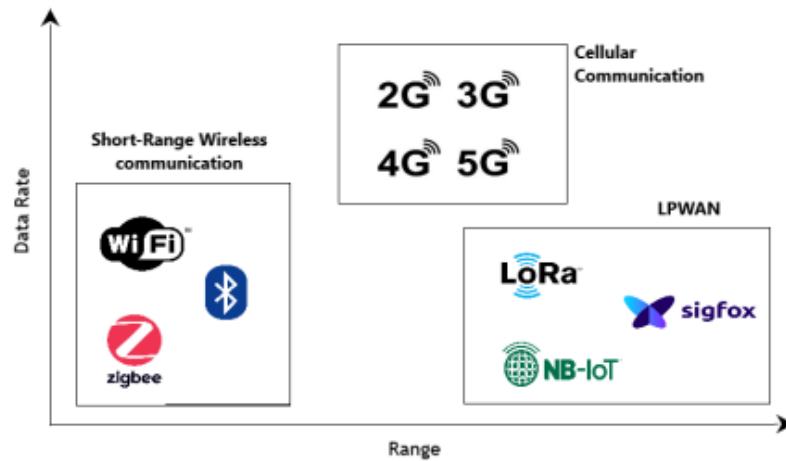


Figura 2.17. Esquema comparativo entre distintas tecnologías tomando en cuenta el rango de alcance y el ancho de banda (Khorsandi y Jalalizad, 2023).

2.10.4. Protocolo LoRa

Esta tecnología fue desarrollada en Francia en el año 2012 por la empresa francesa Semtech. Es el tipo de modulación utilizado entre dos dispositivos LoRa o entre un *end-device* y un *gateway*.

El récord mundial para una transmisión LoRa es de 832 km, logrado por Thomas Telkamp en la banda EU868 utilizando 25 mW / 14 dBm (Montagny, 2021).

Suelen confundirse los términos LoRa y LoRaWAN (Long Range Wide Area Network). LoRa se refiere al tipo de modulación mientras que LoRaWAN es una extensión del protocolo LoRa que da la capacidad de conectar los dispositivos a un servidor para servir de interfaz con el usuario final. Puede definirse LoRaWAN como una arquitectura de red que consta de dispositivos finales, gateways y servidores.

- Funcionamiento: LoRa utiliza una técnica de modulación de espectro ensanchado llamada modulación de espectro ensanchado de baja potencia (CSS, por sus siglas en inglés). Esta técnica permite que las señales de radio se propaguen a distancias más largas y atraviesen obstáculos, lo que proporciona una mayor cobertura en comparación con otras tecnologías inalámbricas.

Para transmitir los datos hace uso de “Chirps” (Compressed High Intensity Radar Pulse por sus siglas en inglés), de donde viene su nombre CSS. Esta señal, que puede observarse en la figura 2.18. Un chirp es una señal que varía en frecuencia linealmente con el tiempo.

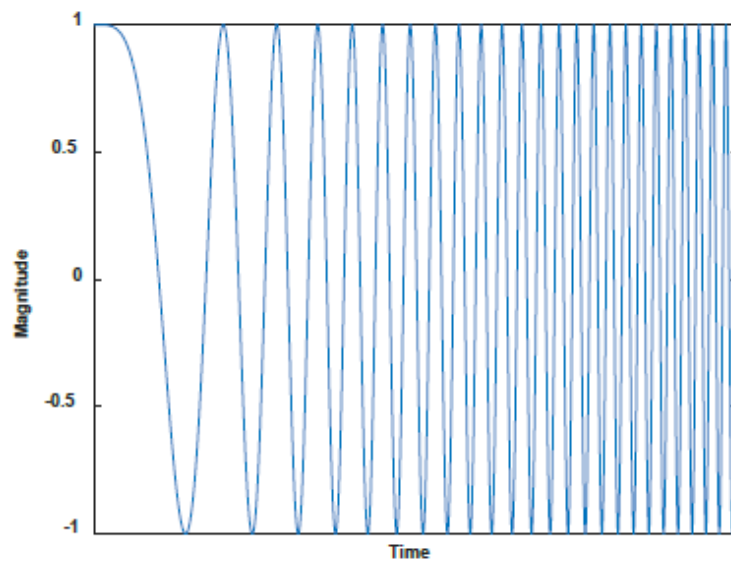


Figura 2.18. Señal chirp utilizada para la transmisión CSS (Aloufi y cols., 2020).

- Ancho de banda: Uno de los parámetros más importantes dentro de un canal de comunicaciones. Puede definirse como la cantidad de espectro de frecuencia ocupado por una señal de transmisión. Es un parámetro importante que determina la capacidad de transmisión de datos y la eficiencia espectral de la comunicación. En LoRa, el ancho de banda se puede configurar en diferentes opciones, como 125 kHz, 250 kHz y 500 kHz. La elección del ancho de banda

afecta directamente la velocidad de transmisión de datos y el alcance de la comunicación.

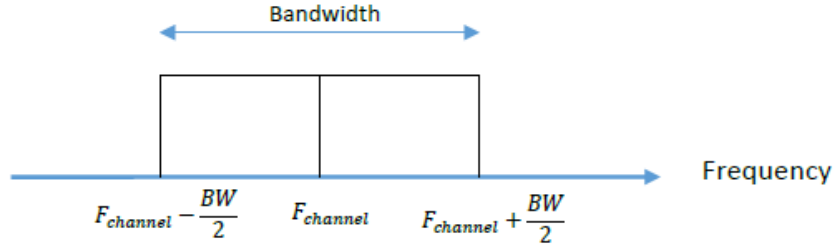


Figura 2.19. Ancho de banda representado en el espectro (Aloufi y cols., 2020).

- Factor de propagación: El factor de propagación (Spreading Factor) se refiere a un parámetro que determina la velocidad de transmisión de datos y la robustez de la comunicación.

El spreading factor se representa como un número entero que varía típicamente de 7 a 12 en la tecnología LoRa. Un spreading factor más bajo indica una velocidad de transmisión de datos más alta, pero con menor alcance y capacidad de penetración de la señal. Por otro lado, un factor de propagación más alto proporciona un mayor alcance de comunicación, pero con una velocidad de transmisión de datos más baja.

En LoRa cada símbolo representa un número de bits transmitidos. Por tanto, se define la siguiente regla:

$$\text{Número de bits transmitidos en un símbolo} = \text{Factor de propagación} \quad (2.11)$$

Para un SF10, un símbolo o *Chirp* representará 10 bits. Esto se observa claramente en la figura 2.20:

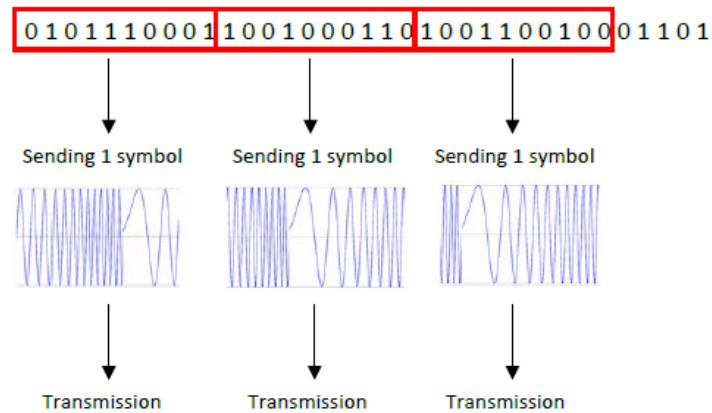


Figura 2.20. Representación de símbolos en una transmisión LoRa (Aloufi y cols., 2020).

- **Potencia:** Los dispositivos LoRa generalmente permiten ajustar la potencia de transmisión de la señal. Esto proporciona flexibilidad para adaptarse a diferentes escenarios y requisitos de alcance. Por ejemplo, en áreas densamente pobladas o con obstáculos, se puede aumentar la potencia de transmisión para superar la atenuación de la señal y mejorar la cobertura. En Venezuela la potencia máxima de transmisión está regulada por CONATEL, siendo en la banda libre de telemetría alrededor de los 470 MHz de 250 mW (24 dBm) (Comisión Nacional de Telecomunicaciones, 2010).
- **Rango:** El rango de transmisión en LoRa puede variar desde unos pocos cientos de metros hasta varios kilómetros, dependiendo de los factores mencionados anteriormente. Utilizando técnicas como factor de propagación alto, potencia de transmisión adecuada y condiciones de propagación favorables, es posible lograr comunicaciones en rangos de varios kilómetros en entornos abiertos y sin obstrucciones.
- **Tiempo de transmisión:** El tiempo de transmisión de cada símbolo depende del factor de propagación. Es decir, a mayor SF, el tiempo de transmisión será mayor manteniendo el ancho de banda igual. Tomando esto en cuenta

se tiene la siguiente ecuación:

$$T_{symbol} = \frac{2^{SF}}{\text{Ancho de banda}} \quad (2.12)$$

- Forma de la trama: Las tramas de LoRa, como la que se muestra en la figura 2.21, siguen una estructura predefinida que consta de 4 partes:

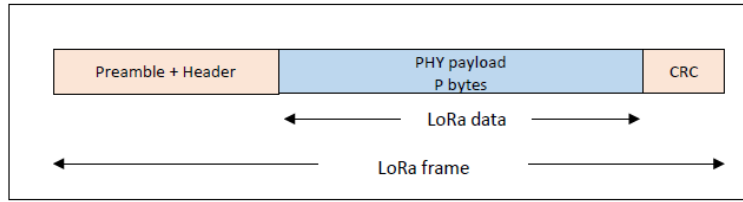


Figura 2.21. Forma de la trama LoRa (Aloufi y cols., 2020).

1. Preámbulo (Preamble): Secuencia de bits que se utiliza para sincronizar el receptor con la transmisión y permitir la detección y sincronización de la trama.
 2. Encabezado (Header): Se puede utilizar un encabezado adicional para incluir información adicional sobre la trama, como la dirección de origen o destino, los identificadores de servicio, etc. Puede ser generada de forma automática o programada.
 3. Carga útil (Payload): Campo principal de la trama que contiene los datos que se van a transmitir.
 4. CRC (*Cyclic Redundancy Checking*): Campo que contiene un código de detección de errores para verificar la integridad de la trama recibida. El receptor realiza un cálculo CRC sobre los datos recibidos y compara el resultado con el valor de CRC incluido en la trama. Si el valor calculado y el valor de CRC coinciden, se asume que la trama se recibió sin errores
- Tasa de codificación (CR): En LoRa, se utiliza la codificación de corrección de errores de tipo “*forward error correction*” (FEC), que permite detectar

y corregir errores en la recepción de los datos. El coding rate especifica la cantidad de bits redundantes que se agregan a los datos para mejorar la confiabilidad de la comunicación. En el caso de un $CR = 4/8$, 8 bits son transmitidos mientras que solo 4 bits contienen información útil.

- Consumo de energía: El protocolo se caracteriza por tener un bajo consumo de energía, permitiendo ser empleado en dispositivos a baterías para envío de datos periódicos. Esto se traduce en velocidades bajas de transmisión y cargas útiles (payloads) menores a los encontrados en otras tecnologías, sin embargo, la relación entre las distancias largas a cubrir y los algoritmos de detección permiten que para ciertas aplicaciones este protocolo sea el ideal, razón por la cual se ha convertido en uno de los más utilizados en años recientes.

2.11. Procesamiento digital de señales:

El procesamiento digital de señales (DSP por sus siglas en inglés) es un proceso que involucra la manipulación de señales de distinta naturaleza en un procesador o computador. La necesidad de este procesamiento surge por la necesidad de mejorar, cambiar o mostrar los datos obtenidos de señales reales en distintas formas mediante métodos matemáticos (Proakis y Manolaki, 1999).

En gran cantidad de aplicaciones, los procesadores analógicos están siendo reemplazados por chips de DSP, esto debido a la disminución en el costo además de la capacidad de procesamiento que sigue en aumento. Si bien se puede hacer procesamiento digital con cualquier procesador o microcontrolador, estos chips están diseñados para realizar cálculos matemáticos y llevar a cabo algoritmos de procesamiento de forma eficiente.

En el contexto de SHM, el DSP es necesario una vez los datos fueron adquiridos en campo, siendo esta herramienta la que permite realizar un estudio exhaustivo de las señales adquiridas por los sensores en el dominio de la frecuencia, siendo este estudio uno de los más importantes y utilizados en esta área.

2.11.1. Estudio en el dominio de la frecuencia

El estudio de señales en el dominio de la frecuencia se remonta al siglo XIX y está estrechamente relacionado con los trabajos del matemático francés Jean-Baptiste Joseph Fourier. Desde entonces, el impacto que ha tenido en la ingeniería y las ciencias es notable. En nuestro día a día, muchos de los sistemas de uso diario hacen uso, de alguna forma u otra, del concepto inicial planteado por Fourier.

En el SHM, el análisis de frecuencia también desempeña un papel importante. En el monitoreo de salud estructural se utilizan técnicas de análisis de frecuencia, como la transformada de Fourier, para identificar patrones y características de frecuencia en las señales registradas por los sensores. Esto permite detectar cambios en las propiedades dinámicas de la estructura, como frecuencias naturales, modos de vibración, y cambios en la rigidez o la masa.

Como se planteó anteriormente, al estudiar el dominio de la frecuencia en el SHM, se pueden identificar patrones anómalos, compararlos con datos de referencia y tomar medidas preventivas o correctivas para garantizar la integridad y la seguridad de la estructura.

Los análisis de vibración suelen analizarse utilizando la Transformada Rápida de Fourier (FFT por sus siglas en inglés), un algoritmo computacional que ha permitido hacer uso de la Transformada de Fourier en todo tipo de aplicaciones. Para entender la FFT, es necesario definir el desarrollo matemático de la Transformada

de Fourier:

2.11.2. Transformada de Fourier

Es una herramienta matemática utilizada para descomponer una función en sus componentes de frecuencia. Permite analizar una señal en el dominio de la frecuencia y determinar las diferentes frecuencias que la componen. Se ha convertido en una de las principales herramientas para resolver muchos desafíos de la comunidad científica, siendo su aplicación más conocida en el análisis de sistemas lineales invariantes en el tiempo. Se define matemáticamente como sigue:

$$F(k) = \int_{-\infty}^{\infty} f(x)e^{-2\pi i k x} dx \quad (2.13)$$

En la figura 2.22 se observa como una señal compuesta por distintas sinusoidales que varían en amplitud y frecuencia, son representadas en el dominio de la frecuencia como componentes aislados.

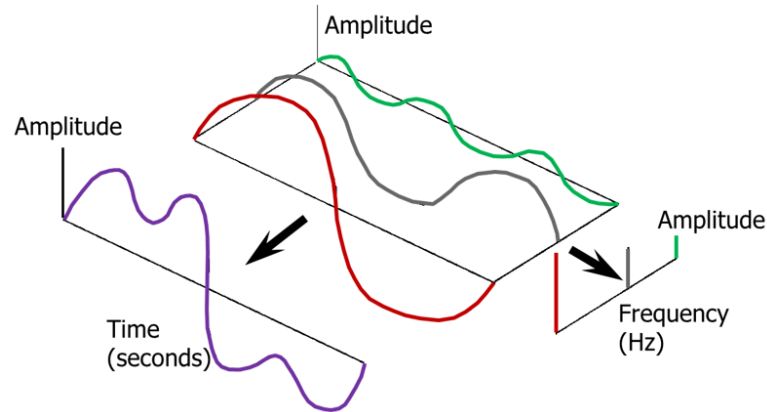


Figura 2.22. Descomposición de señal temporal en el espectro de frecuencia (Siemens, 2019).

2.11.3. Transformada Rápida de Fourier

Tal vez uno de los algoritmos más importantes de todos los tiempos. Descrita por algunos de sus autores principales, la transformada rápida de Fourier (FFT por sus siglas en inglés) es una herramienta computacional que facilita el análisis de señales en el espectro de la frecuencia mediante el uso de computadores digitales. Ejecuta la transformada discreta de Fourier de una forma más eficiente

La transformada de Fourier exige un cálculo que puede ser computacionalmente costoso, especialmente cuando se trata de muchos puntos. Es ahí donde la FFT explota las simetrías matemáticas para lograr disminuir el cálculo de una complejidad $O(N^2)$, una función prácticamente lineal en “n”, a mayor “n” el valor no crece tan rápidamente que la expresión cuadrática original. Siendo N es el número de datos, a una complejidad de $O(N \log_N)$. Se describe matemáticamente utilizando la siguiente expresión:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j2\pi kn/N} \quad (2.14)$$

Donde:

- $X[k]$ = Muestra “k” de la transformada discreta de Fourier (DFT).
- $x[n]$ = Muestra “n” de la señal de entrada.
- N = Cantidad de muestras.
- j = Unidad imaginaria
- e = base del logaritmo natural.

Una vez se lleva a cabo la FFT, se debe analizar el espectro en frecuencia para así obtener los parámetros de interés, entre algunas de estas técnicas se encuentran:

- Método de *Peak Picking*: Este método es utilizado para determinar la ubicación de los picos sobresalientes de la representación gráfica de una unidad física. En el contexto del dominio de la frecuencia, estos picos se corresponden con la respuesta en frecuencia del sistema, asociándose con las frecuencias de vibración del sistema en estudio. El peak picking asume que todo pico en frecuencia se corresponde con uno de los modos de vibración natural del sistema, en el caso del espectro de la figura 2.23, serían los picos encerrados en círculo en el espectro. Esta premisa implica que al estar los picos muy cercanos entre sí, se dificulte el proceso de evaluación usando esta técnica.

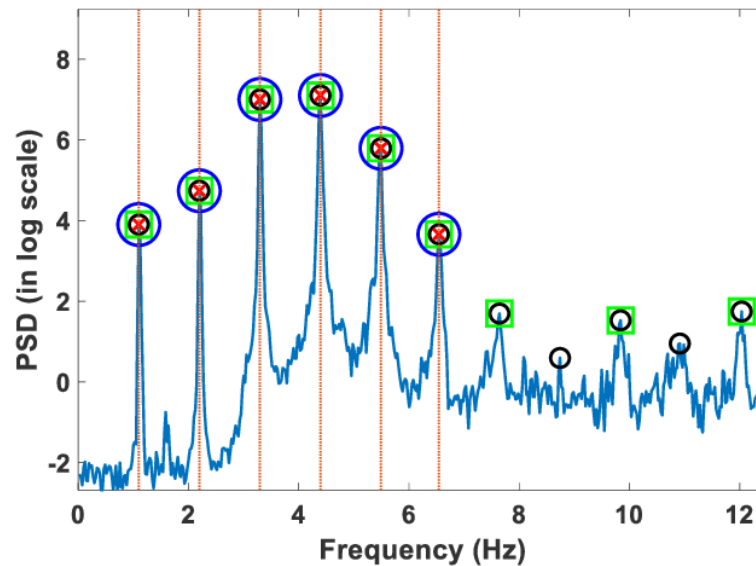


Figura 2.23. Método de *peak picking* (Jin y cols., 2021).

- Estimación del amortiguamiento por ancho de banda local: Una vez identificadas las frecuencias naturales del sistema, es de interés conocer el factor de amortiguamiento que corresponde a cada uno. Para esto, se suele aplicar

el método de ancho de banda local, también conocido como método de la potencia mitad. Consiste en estudiar el espectro de potencia y ubicar los valores en los cuales la potencia de la señal se reduce a la mitad, en el caso de la figura 2.24, estas frecuencias serían f_{max} y f_{min} , siendo la diferencia entre ambos Δf y la frecuencia central f_o . Una vez obtenidas estas frecuencias se calcula el amortiguamiento de la siguiente forma:

$$\% \zeta = \frac{\Delta f}{2f_o} \quad (2.15)$$

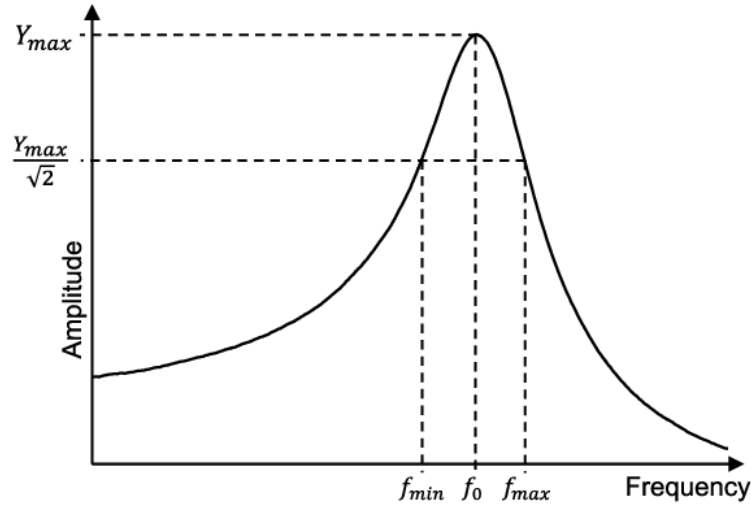


Figura 2.24. Método de cálculo de amortiguamiento por ancho de banda local (Andresen y cols., 2019).

2.11.4. Aventanamiento

Las señales provenientes de los sensores representan una ventana finita de tiempo, es decir, contienen una cantidad finita de datos. Es bien conocido que la FFT asume señales periódicas en el tiempo, eso se traduce en que las ecuaciones asumen que los datos se repiten una vez se termina el registro hasta el infinito. Esto genera discontinuidades notables en los bordes del registro que suelen verse como

componentes frecuenciales que aparecen alrededor de las frecuencias naturales reales del sistema en estudio, fenómeno conocido como filtración de espectro o *spectral leakage*, estas señales de alta frecuencia aparecen como “alias” en las componentes de baja frecuencia, comportamiento que es indeseado para términos de análisis.

Para evitar este fenómeno se utiliza el método de aventanamiento, en el cual se multiplica la ventana de tiempo por una función que suaviza los ejes y disminuye su contribución en el espectro. Estas funciones hacen que los bordes del registro se “encuentren” como si se tratase de una señal periódica, resultando en un espectro mucho más limpio y en picos más claros.

Algunas de las funciones de aventanamiento más utilizadas:

- Hanning.
- Hamming.
- Blackman.

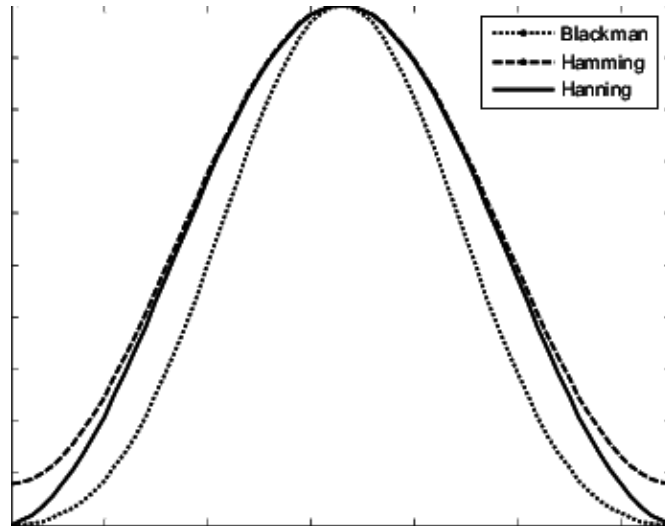


Figura 2.25. Funciones de aventanamiento más utilizadas (Francone y cols., 2006).

CAPÍTULO III

MARCO METODOLÓGICO

En este capítulo se describirá el diseño del sensor inteligente, describiendo detalles de hardware y software del mismo.

3.1. Descripción breve del sistema

El sistema diseñado integra un conjunto de sensores y módulos que permiten medir variables dinámicas y cuasi-estáticas de un sistema estructural, enviarlas a larga distancia y posteriormente procesarlas y almacenarlas.

Una vez el sistema es encendido, calibra de forma automática todos los sensores, con especial énfasis en eliminar el offset de los 3 ejes del acelerómetro y también calibrar el magnetómetro y giróscopo del acelerómetro de 9 grados de libertad. El sistema ejecutará las tareas de calibración cada 2 registros de datos enviados con éxito.

Luego de la calibración y ajuste de la hora y fecha, el sistema comienza a medir de forma continua la aceleración triaxial, inclinación, temperatura, humedad y estima la inclinación con sensores electrónicos de bajo consumo para posteriormente enviar los datos de forma inalámbrica a la estación base, que puede estar ubicada a más de 150 metros de distancia. Allí son recibidos, decodificados y preprocesados para luego ser subidos vía inalámbrica a un computador que sirve de

servidor en donde se almacenaran los datos, siendo controlado el sistema desde esta misma estación base, pudiendo enviar comandos de adquisición de datos de forma remota. A su vez, se desarrolló una interfaz gráfica que permite visualizar los datos obtenidos, realizar peticiones de datos a distancia, observar sus características principales, acceder al histórico de datos recogidos por el sensor en una fecha específica, descargar los datos y ejecutar post-procesamiento a los mismos para evañuar las variables de interés para el monitoreo de la salud estructural.

Puesto que la estación base está conectada a internet, el microcontrolador ubicado en la estación adquiere la fecha y hora actualizada utilizando un servidor del protocolo NTP (*Network Time Protocol*) y sincroniza su RTC (*Real Time Clock*) interno con estos valores. Una vez obtenida la fecha y hora, espera el comando de inicio del sensor inteligente que envía de forma automática una vez se enciende y calibra sus sensores, la cual le indica a la estación base que debe enviar la fecha y hora actual. De esta forma se sincronizan los relojes internos de ambos y permite al sensor inteligente tener la fecha y hora a la cual tomó todo registro, enviando esta información como parte del registro de datos. La fecha y hora se envía bajo el formato *UNIX Epoch*, el cual indica la cantidad de segundos transcurridos desde el 1 de Enero de 1970.

En la estación base, el sistema se conectará a internet a través de la red WiFi, enviará los datos recibidos al computador en la estación base, siendo esta herramienta la encargada de pre-procesar los datos recibidos vía inalámbrica y convertirlos a un formato adecuado para poder ser almacenados en el computador y posteriormente procesados usando librerías de análisis numérico. La interfaz de usuario podrá estar disponible para todos los usuarios de la red local, siempre que tengan los servicios necesarios instalados en local.

El comportamiento de los sistemas estructurales a estudiar condiciona los rangos e intervalos de medición de los sensores, con un límite superior cercano

a los 3 g en un movimiento telúrico considerable. Por su parte, la temperatura, humedad e inclinación suelen considerarse variables cuasi-estáticas, permitiendo que el sistema mida estas variables con intervalos lo suficientemente largos para poder monitorear cambios considerables en las mismas y posteriormente correlacionarlos a las condiciones estructurales. La frecuencia de muestreo de aceleración es deseable que est

El sistema en su funcionamiento normal está ejecutando las siguientes tareas principales:

- Adquisición continua (envío programado a ciertas horas del día o ante eventos importantes): El sensor inteligente mide constantemente las variables de interés y envía periódicamente, a ciertas horas del día programadas con antelación, un registro de datos a la estación base. Al estar midiendo de forma continua, está atento para generar una interrupción o alerta ante algún valor de aceleración o inclinación que esté por encima de algún límite escogido con anterioridad que dependerá en gran medida de la estructura a monitorear y su naturaleza, aunque se escogió por default el valor de 2 m/s como generador de alerta, basado en la *Escala de Mercalli* (Survey, 2011), la cual indica que este valor de aceleración (que corresponde a 0.20 g) equivale a un sismo fuerte con daño moderado. El sistema envía de forma automática un registro de datos del acontecimiento importante que generó la alerta.
- Escuchando petición de trama de datos inmediata (Envía trama ante request/query de estación base): Al recibir desde la estación base el comando de adquisición de datos, ejecutado desde la interfaz de control por el operador, el sistema comienza a tomar un registro de datos de forma inmediata, siempre y cuando el mismo no haya detectado previamente un evento importante que superara los límites establecidos, y lo envía a la estación base permitiendo que el usuario obtenga información del sistema en el instante en el cual se

realiza la petición de los datos. Una vez es enviado y recibido con éxito la trama de datos, el sistema regresa a su estado anterior, tomando datos de forma continua y esperando alertas, comandos u horas programadas.

3.2. Selección de componentes

Una vez obtenida una base teórica sobre estos temas, se procedió a escoger el hardware y los protocolos de comunicación más adecuados para llevar a cabo el objetivo de diseñar un sensor inteligente para aplicaciones de monitoreo de salud estructural.

3.2.1. Protocolo de comunicaciones

Para el protocolo de comunicación se buscaron protocolos capaces de manejar los datos recogidos por los sensores de forma eficaz y confiable. En este caso se refiere al protocolo utilizado para enviar los datos desde el sensor inteligente hasta la estación base. Sin embargo, se utilizaron distintos protocolos para la comunicación de los sensores con el microcontrolador y a su vez para comunicar la estación base con el receptor de datos.

Para el envío de datos a la estación base, preferiblemente el protocolo debía ser capaz de funcionar en rangos de distancia amplios, permitiendo que el sensor inteligente esté ubicado lejos de la estación base en donde serán monitoreadas las variables de interés. En primer lugar se escogieron algunos protocolos de forma preliminar, para luego estudiar a fondo sus características. Estos protocolos y sus características se resumen en la tabla 3.1:

Tabla 3.1. Comparación entre protocolos de comunicación inalámbrica, (Ogdol y cols., 2018) y (Blackman, 2019)

Protocolo	Frecuencia	Rango	Velocidad	Consumo de energía
Zigbee	784 MHz/2.4 GHz	100 m - 300 m	250 kbps-500 kbps	Bajo
Sigfox	868 MHz/915 MHz	3km - 10km	100 bps	Bajo
NB-IoT	LTE	1km - 10km	200 kbps	Bajo
WiFi	2.4 GHz/5.8 GHz	100m	54Mbps/1.3Gbps	Alto
Bluetooth	2.4 GHz	10m - 100m	720 kbps	Bajo
LoRa	430 MHz/433 MHz /868 MHz/915 MHz	15 km-30 km	0.3kbs hasta 50 kbps	Bajo

Con base en esta información, se escogió el protocolo LoRa como el más adecuado para el sensor inteligente, debido a su amplio rango y bajo consumo de energía. En general, la vasta mayoría de los módulos están basados en los chips fabricados por Semtech (los precursores del protocolo LoRa) SX126X y SX127X, por tanto se compararon ambas tecnologías:

Tabla 3.2. Comparación entre módulos LoRa del fabricante Semtech (Semtech, 2015).

Módulo	Modem	Amplificador	Corriente RX	Sensibilidad	Velocidad (bit rate)
SX1261/62/68	LoRa y FSK	+15 dBm - +22 dBm	4.6 mA	-148 dBm	62.5 kbps - 300 kbps
S1272/73	LoRa	+14 dBm	10 mA	-137 dBm	300 kbps
S1276/77/78/79	LoRa	+14 dBm	9.9 mA	-148 dBm	300 kbps

3.2.2. Sensores

Para la selección de los sensores a utilizarse, es preciso definir las necesidades de un sistema de adquisición para sistemas estructurales, siendo su comportamiento el que define las características de los instrumentos de medición.

Aceleración

En el caso de la medición de aceleración, el sensor inteligente debe contar con un sensor con las siguientes características:

- Bajo nivel de ruido.
- Compensación de temperatura.
- Ancho de banda dentro del rango deseado en sistemas estructurales.
- Buena resolución.
- Suficientes grados de libertad.
- Compatibilidad con microcontroladores disponibles en el mercado.
- Bajo consumo

Temperatura y humedad

Para la medición de temperatura y humedad, el sensor inteligente debe contar con un sensor con las siguientes características:

- Rango de trabajo dentro de las condiciones en las que se encuentre la estructura.
- Buena resolución y sensibilidad.
- Compatibilidad con microcontroladores.
- Bajo consumo.

Inclinación

Para la medición de inclinación, la cual, como se explica en la sección 2.6.4 el sensor inteligente debe contar con un sensor que cuente con las siguientes características:

- Acelerómetro, giróscopo incorporado, convirtiéndose en una IMU (*Inertial Measurement Unit*) o en el mejor de los casos, incluir un magnetómetro, para convertirse en un MARG (*Magnetic, Angular Rate, and Gravity*)
- Bajo nivel de ruido.
- Buena resolución y sensibilidad.
- Compatibilidad con microcontroladores.
- Bajo consumo.

3.2.3. Microcontroladores

En el caso de los microcontroladores, se buscaron dispositivos capaces de obtener los datos provenientes de los sensores, procesarlos, almacenarlos temporalmente y posteriormente hacer uso del módulo de comunicaciones para su envío, usando este mismo módulo para recibir mensajes o comandos. También se tomaron en cuenta las capacidades de conexión inalámbrica de cada microcontrolador, su documentación y soporte por parte de los fabricantes, y por último su compatibilidad con los distintos frameworks, librerías y entornos de programación disponibles para los sensores y módulos, los cuales disminuyen el tiempo necesario para poner en marcha el funcionamiento del sistema. Se estudiaron las características de distintas placas de desarrollo, para posteriormente escoger el más adecuado. A

continuación, en la tabla 3.3, se presentan las placas de desarrollo consideradas de forma preliminar y sus características principales:

Tabla 3.3. Comparación entre placas de desarrollo basadas en MCU

Placa	Procesador	Velocidad de reloj	RAM (kB)	ROM (kB)	GPIO	Conectividad
Teensy 4.0	ARM M7	600 MHz	1024	2048	40	-
Raspberry Pi Pico W	Dual ARM-M0	133 MHz	264	2048	26	WiFi
STM32 Discovery	ARM M4	168 MHz	192	1024	82	-
STM32 Nucleo	ARM M0 - ARM M4	84 MHz - 180 MHz	96 - 128	512	50	-
Espressif ESP32	Dual Xtensa LX6	240 MHz	520	4096	34	WiFi/BT(BLE)
STM32 Blackpill	ARM M4	100 MHz	128	512	37	-

3.2.4. Diagramas de selección de componentes:

Selección del microcontrolador

Siendo el componente central del sensor inteligente y el encargado de darle al sensor sus capacidades de interconexión y preprocesamiento, el microcontrolador cumple el rol más importante dentro del sistema. Para la elección de la placa de desarrollo a usar se tomaron en cuenta las placas estudiadas y mostradas en la tabla 3.3 que mostraron potencial, obteniendo los resultados observados en la figura 3.1

Diagrama de araña para selección de tarjeta de desarrollo basada en MCU

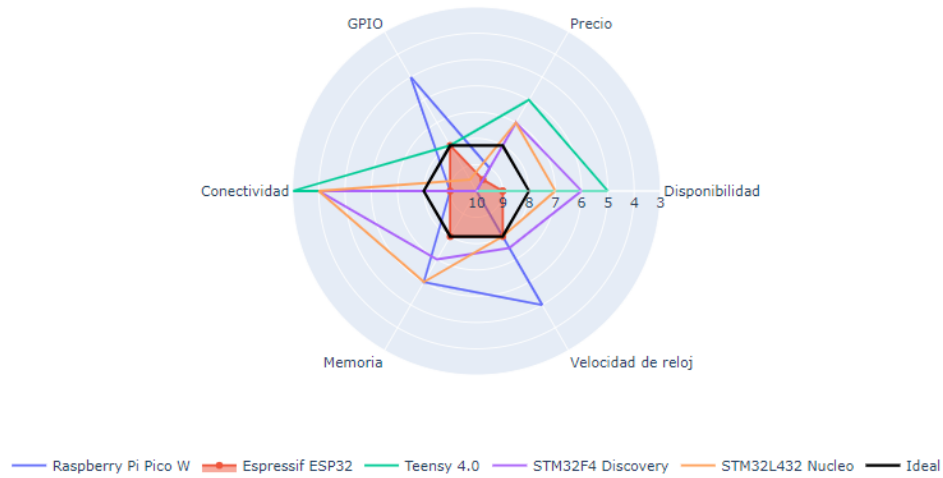


Figura 3.1. Diagrama de araña para selección de microcontrolador.

Selección del sensor de aceleración

Con base en la información recopilada de distintos módulos de acelerómetros, se observa en la figura 3.2 que el MPU6050 de Invensense se ajusta a las necesidades del acelerómetro necesario para tomar los registros de vibración. Otro módulo del mismo fabricante, el MPU9250 también presenta un buen desempeño. Sin embargo, el MPU6050 tiene un mejor precio y ofrece funcionalidades similares, por lo cual fue el escogido para el prototipo de pruebas.

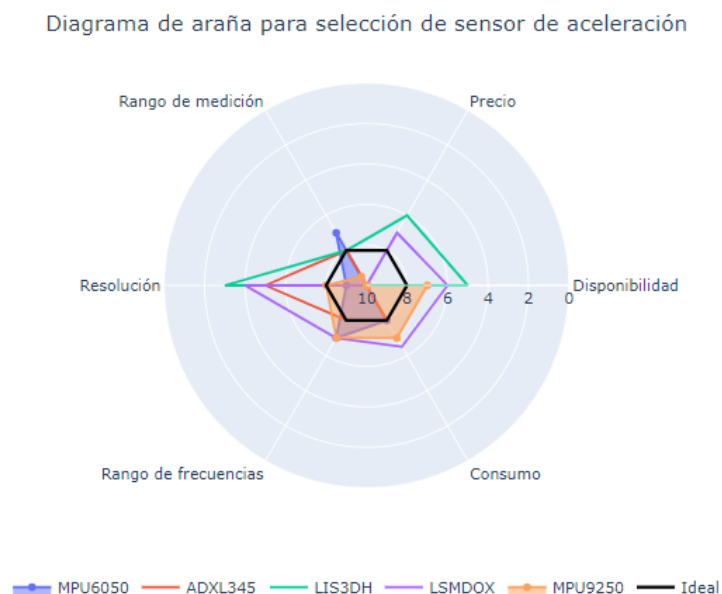


Figura 3.2. Diagrama de araña para selección de acelerómetro.

Selección del sensor de temperatura y humedad

Se observa en la figura 3.3 que la mayoría de los sensores de temperatura y humedad, los cuales suelen estar integrados en un mismo módulo, no distan mucho en desempeño entre sí, sin embargo, entre ellos destaca el BME280 del reconocido fabricante Bosch, el cual cuenta con buena resolución además de una excelente documentación y librerías para distintos microcontroladores. El módulo SHT31 muestra potencial por su resolución, en este caso se descarta por la poca disponibilidad del módulo, pero es una buena opción para futuras implementaciones. Es por esta razón que se escogió el BME280 para llevar a cabo las mediciones de las variables ambientales en el prototipo de pruebas.

Diagrama de araña para selección de sensor de temperatura y humedad

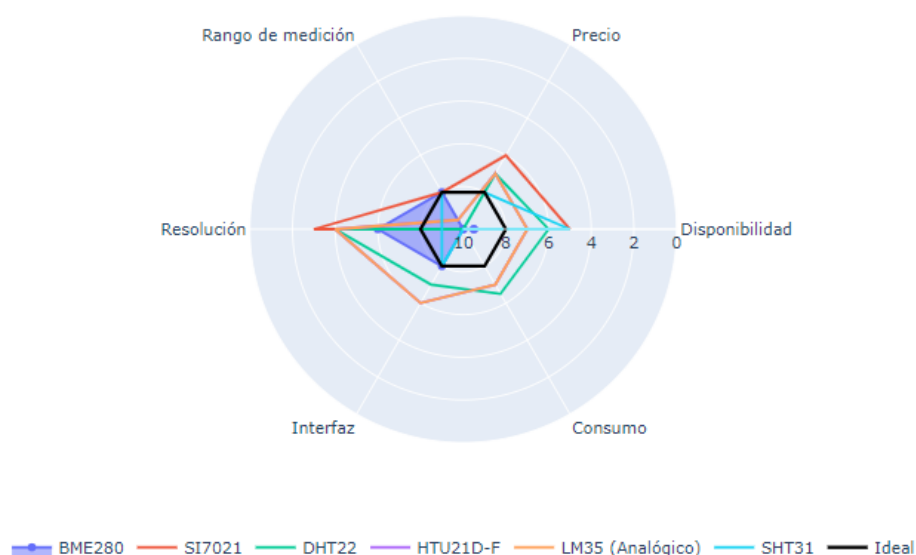


Figura 3.3. Diagrama de araña para selección de sensor de temperatura y humedad.

Selección del acelerómetro para estimación de ángulos

Utilizando el mismo análisis que se llevó a cabo en la figura 3.2, se modifica para fines de estimación de ángulo tomando en cuenta las premisas de la sección 2.6.4 para la fusión de sensores. Por tanto, con base en la figura 3.4 se elige el módulo MPU9250, el cual cumple con la función de ser una MARG (*Magnetic, Angular Rate, and Gravity*) de 9 grados de libertad, siendo ideal para la estimación de ángulos en el prototipo.

Diagrama de araña para selección de unidad de medición inercial

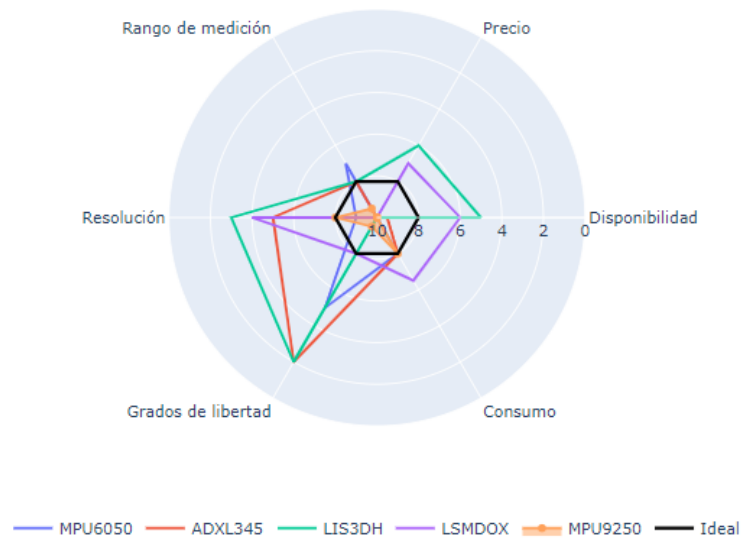


Figura 3.4. Diagrama de araña para selección de unidad de medición inercial.

Selección del módulo de comunicaciones

Se ubicaron módulos de comunicaciones LoRa que fueran compatibles con el microcontrolador escogido, con documentación disponible y cuyas características se ajustaran a las necesidades del proyecto a llevar a cabo. Si bien el protocolo es el que condiciona las características de la gran mayoría de los módulos de comunicación del protocolo en cuestión, se buscó un módulo con facilidad de conexión e intercomunicación con el microcontrolador. Se observa en la figura 3.5 que la mayoría de los módulos tienen un desempeño similar, esto se debe a que están basados en distintas versiones de los módulos estudiados en la tabla 3.2. Sin embargo, el condicionante es la disponibilidad y precio de los mismos, siendo el RA-02 de Ai-Thinker el seleccionado en este caso.

Diagrama de araña para selección de transmisores de radiofrecuencia

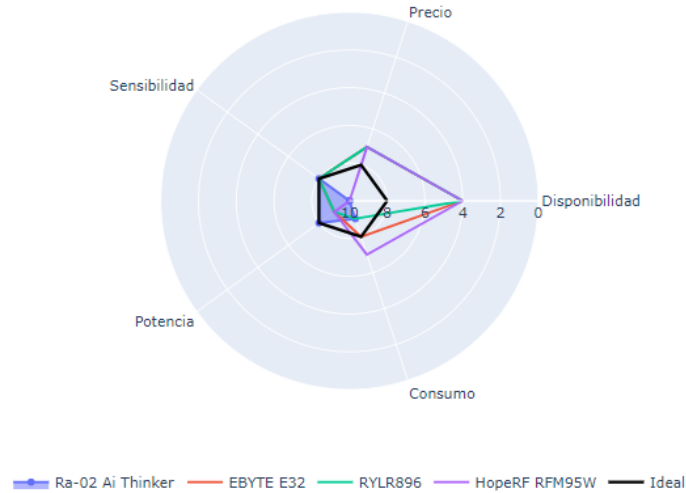


Figura 3.5. Diagrama de araña para selección del módulo de comunicaciones.

3.3. Detalle del diseño

A continuación se describe en detalle tanto el hardware como el software utilizado para llevar a cabo el sensor inteligente, además de las herramientas utilizadas para verificar su funcionamiento.

3.3.1. Descripción detallada del sistema:

Complementando lo descrito en el apartado 3.1, el prototipo del sistema se encarga de tomar registros de vibración estructural utilizando el acelerómetro MPU6050 mientras se miden variables ambientales, mediante el sensor de temperatura y humedad BM280, y se estima la inclinación mediante el MARG MPU9250. Estos datos son recopilados y preprocesados por la tarjeta de desarrollo ESP32 DOIT DEVKIT, basada en el ESP32 de Espressif. Al encender el sensor inteligente, el cual será alimentado mediante el puerto USB-C de la tarjeta de

desarrollo o mediante una fuente de 5 V a los pines de alimentación del ESP32, este comienza la calibración de los sensores con tareas que serán descritas con más detalle en apartados siguientes. Una vez culmina la calibración, el sensor inteligente inicializa el módulo de comunicaciones LoRa RA-02 de Ai-Thinker, el cual una vez inicializado, envía un mensaje a la estación base solicitando la fecha y hora actual, con lo cual la estación base recibe, decodifica y responde al mensaje con los datos solicitados, lográndose la sincronización del reloj RTC interno del ESP32.

Luego de realizadas las tareas de calibración, inicialización y sincronización satisfactoriamente, el sensor inteligente comienza a tomar datos de todos los sensores de forma continua, manteniéndose alerta a cualquier evento que supere los valores límite en donde está ubicado el sensor. Como se describió en el apartado 3.1, el sensor inteligente está configurado para enviar un registro ante los siguientes eventos:

- El valor de aceleración medido por el sensor en algún eje supera el valor límite establecido.
- El reloj RTC interno indica que la hora actual se corresponde con una de las horas de envío programadas previamente.
- El sensor inteligente recibe un comando desde la estación base el cual indica que se comience a tomar un registro inmediato.

Una vez se toma el registro, este es almacenado temporalmente en buffers que se son parte de estructuras de datos para posteriormente ser enviado utilizando el módulo de comunicaciones RA-02 de Ai-Thinker mediante paquetes sucesivos. Al terminar de enviar todos los paquetes, el sensor inteligente vuelve a su estado normal de toma de datos continua.

El sensor inteligente cuenta con 3 leds (*Light Emitting Diodes*) indicativos:

- LED Amarillo encendido: Calibración.
- LED Verde encendido: Toma de un registro de datos por evento.
- LED Rojo intermitente: Toma de datos continua activa en espera de eventos.

La tarjeta de desarrollo utilizada cuenta con un LED azul integrado que se utiliza para notificar sobre la recepción o envío de algún paquete LoRa por el módulo de comunicaciones.

En la estación base se cuenta con otro módulo ESP32 DOIT DEVKIT, el cual se conecta a la red WiFi, dentro de la cual está ubicado un computador que cuenta con los servicios de NodeRED y Mosquitto, herramientas que se describirán en detalles más adelante. Este computador sirve de base de datos y broker para los datos a recibirse vía MQTT desde el ESP32 de la estación base, y además, proporciona una interfaz de monitoreo y control, basada en Python, la cual permite acceder a los registros y enviar comandos de adquisición de datos al sensor inteligente vía LoRa.

Al estar conectado a WiFi, el ESP32 sincroniza su reloj RTC interno con el de un servidor del protocolo NTP. El ESP32 es el encargado de recibir los datos provenientes del sensor inteligente mediante un módulo de comunicaciones LoRa Ra-02 similar al del sensor inteligente. Los datos se almacenan temporalmente en buffers de datos para posteriormente convertirlos a formato JSON (*JavaScript Object Notation*), la cual permite enviar los datos al broker MQTT en la red local. Una vez los datos son recibidos por el broker, estos son parseados utilizando la herramienta NodeRED, la cual genera un formato tal que puede almacenarse en local usando el formato CSV (*Comma separated values*). A su vez, NodeRED genera una notificación que envía un mensaje MQTT a la interfaz gráfica indicando

que un nuevo registro está disponible. Los registros se guardan según su fecha y hora de llegada al broker en una carpeta específica en local.

La interfaz gráfica de usuario o GUI (*Graphic User Interface*), basada en Python, se encarga de proporcionar una aplicación sencilla en donde pueden verse datos importantes de los registros, tales como:

- Valores de inclinación de la estructura durante el registro escogido.
- Valores de temperatura y humedad del registro.
- Gráfico de aceleración vs. tiempo tomado por el sensor inteligente.
- Espectro en frecuencia utilizando la FFT del registro de aceleración.
- Valores máximos de picos en espectro en frecuencia (método de peak picking automatizado).
- Gráfico de la Densidad Espectral de Potencia (PSD por sus siglas en inglés).
- Gráfico de barras de los valores de inclinación para verificar si el dispositivo está a nivel.

Además de la posibilidad de visualizar y monitorear el estado de la estructura mediante la lectura de sus registros, la GUI permite:

- Enviar un comando de adquisición de datos al sensor inteligente.
- Seleccionar el archivo a estudiar.
- Guardar el archivo seleccionado en otra ubicación, como puede ser un dispositivo extraíble o en una carpeta creada para un estudio en específico, en formato CSV.

- Indicación de registro nuevo disponible.
- Indicación de sensor inteligente a nivel, y de no estarlo, indica el eje en el que se debe ajustar.
- Fecha y hora del registro escogido para control de eventos.
- Botón de actualización de gráfica.
- Menú para escoger entre 3 funciones de aventanamiento distintas para ser aplicadas a los datos a procesarse.

3.3.2. Diagrama general funcionamiento del sensor inteligente en conjunto con la estación base

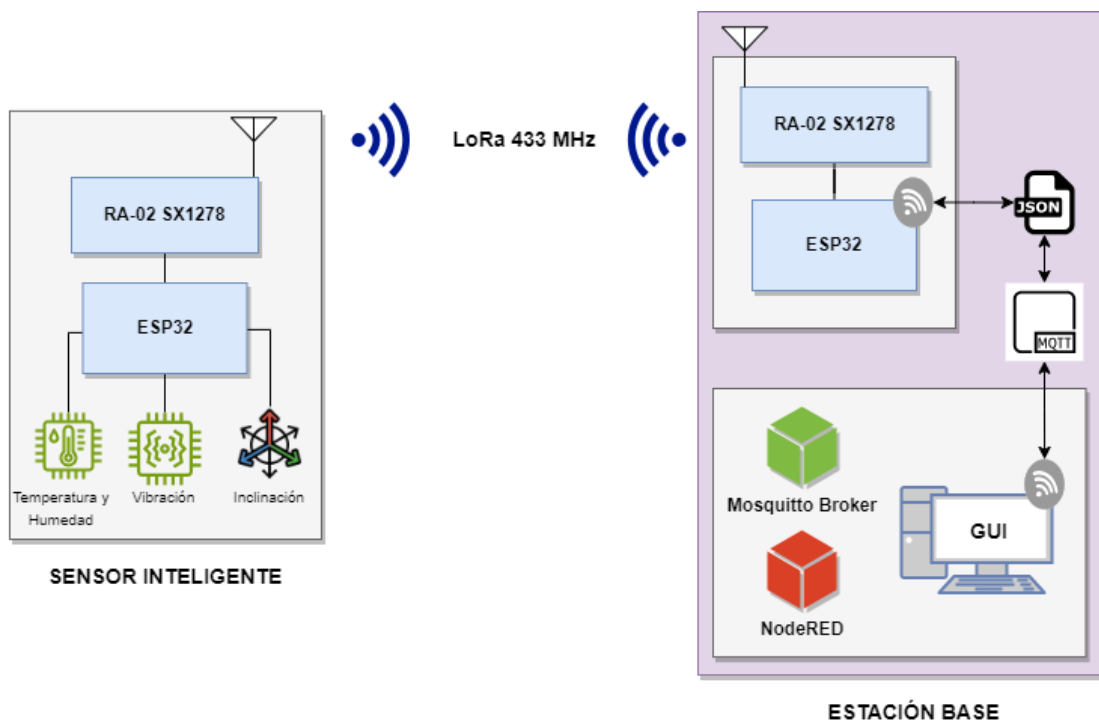


Figura 3.6. Esquema general del sistema.

3.3.3. Descripción del hardware

A continuación se describen las conexiones y el hardware utilizado tanto para el sensor inteligente como para la estación base.

Basados en el funcionamiento del sistema explicado en el apartado 3.3.1 y 3.1, y tomando en cuenta los componentes escogidos en el apartado 3.2, se procede a realizar los siguientes diagramas de bloques:

Sensor inteligente:

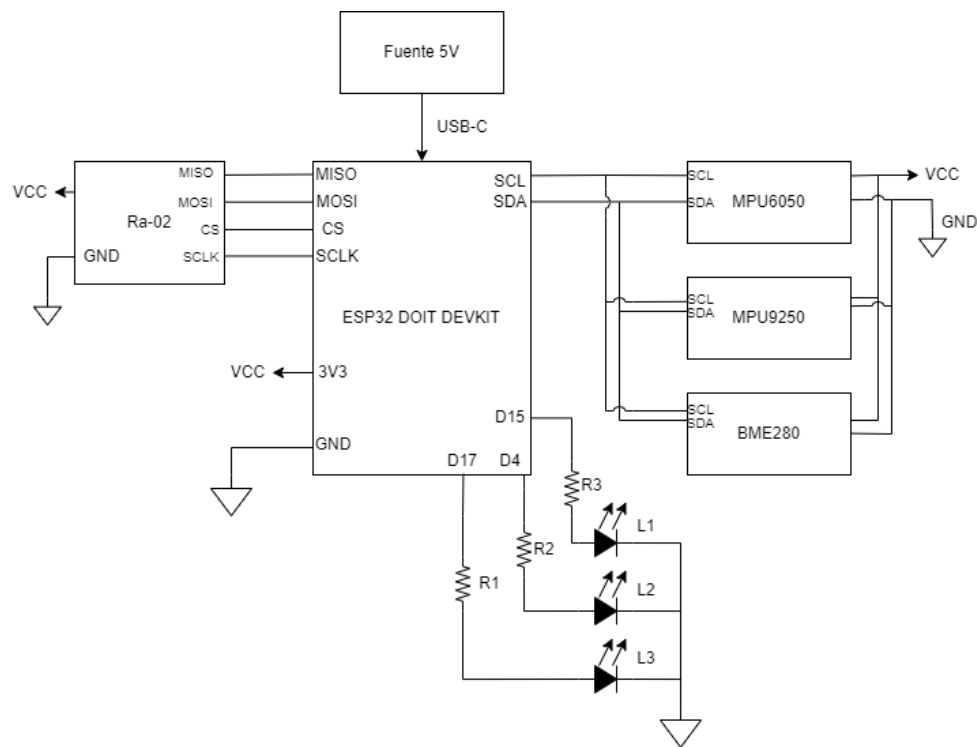


Figura 3.7. Diagrama de bloques del sensor inteligente.

Estación base:

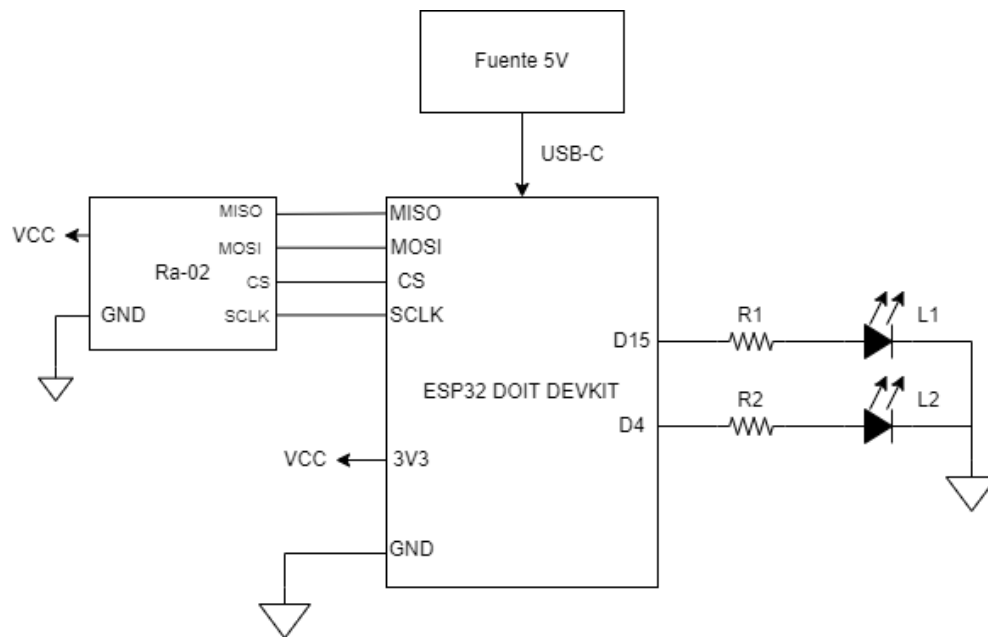


Figura 3.8. Diagrama de bloques de la estación base.

3.3.4. Descripción del software

A continuación se describen las distintas tareas programadas tanto en el sensor inteligente como en la estación base para cumplir con el funcionamiento descrito en el apartado 3.3.1.

Programación del ESP32:

La documentación del fabricante del ESP32 *Espressif Systems* especifica los requerimientos de hardware y software para poder programar el ESP32. En la figura 3.9, se observan las herramientas requeridas para subir una aplicación al microcontrolador.

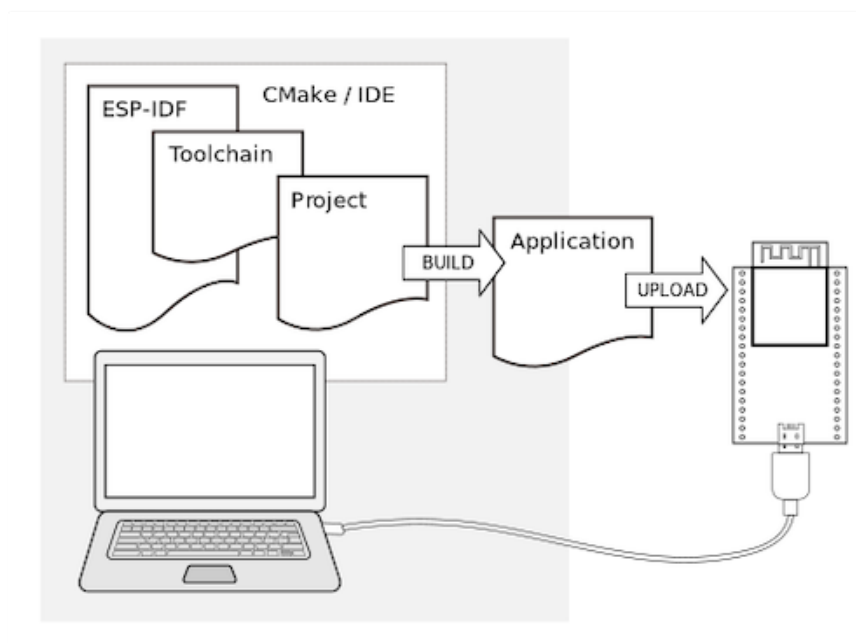


Figura 3.9. Ilustración del proceso para programar el ESP32 (Systems, 2016).

PlatformIO surge como una necesidad de los desarrolladores de sistemas embebidos al tener que instalar herramientas específicas para cada fabricante de hardware, además de la tediosa tarea de la instalación de los toolchains y frameworks de trabajo para cada placa de desarrollo. El IDE (*Integrated Development Environment*) de PlatformIO es una herramienta para sistemas embebidos que permite trabajar con múltiples plataformas, arquitecturas y frameworks en el mismo lugar, permitiendo programar distintos microcontroladores, acceder a librerías, e incluso depurar el código por línea dentro del mismo IDE. PlatformIO está disponible como una extensión del conocido IDE Visual Studio Code, desarrollado por Windows. Visual Studio Code es un editor de código ligero que permite crear aplicaciones en Windows y Linux.

Para el desarrollo del programa se utilizó PlatformIO v3.3.3 y Visual Studio Code en el sistema operativo Windows 10. El microcontrolador utilizado fue el ESP32 DOIT DevKit1, usando el Arduino Core del mismo. El Arduino Core

permite programar el microcontrolador usando el lenguaje C++, además de las facilidades para la implementación de librerías por la gran comunidad que desarrolla proyectos usando esta herramienta.

Librerías

Para llevar a cabo las distintas tareas del proyecto se hizo uso de una serie de librerías escritas en C++ para facilitar la comunicación con los sensores y módulos. A continuación se mencionan las librerías utilizadas y su función dentro del proyecto:

- AdafruitBME280: Desarrollada por Adafruit, implementa rutinas para facilitar la comunicación con sensores Bosch BME280 de temperatura, humedad y presión, permitiendo acceder a los registros donde se almacenan las mediciones a través de la clase AdafruitBME280.
- AdafruitMPU6050: También desarrollada por Adafruit Industries, facilita la comunicación con sensores MPU6050, especialmente con placas de componentes o “*breakout boards*”, las cuales contienen el sensor y el hardware necesario para facilitar su comunicación y conexión a un microcontrolador a través de pines de salida. Implementa funciones de adquisición al leer los registros del sensor MPU6050. Para mejorar las capacidades de esta librería, se añadieron funciones de “*Self test*” y de lectura de bytes tomadas de la implementación de Kris Winer de la librería MPU6050. La implementación de la clase AdafruitMPU6050 permite acceder a esta función de forma ordenada y sencilla.
- MPU9250: Desarrollada por Hideaki Tai. Se escogió por implementar algoritmos de estimación de inclinación basados en el filtro de Madgwick y Mahony. Se puede acceder a estas funciones a través de la clase MPU9250.

- RadioLib: Desarrollada por Jan Gromes, representa una de las librerías más utilizadas para comunicaciones inalámbricas en proyectos de software libre, soportando 13 módulos de comunicaciones distintos y 8 protocolos de comunicaciones distintos. Se escogió esta librería por su facilidad para controlar los módulos de Semtech de las series SX126X y SX127X, permitiendo configurar el módulo, sus parámetros más importantes, y posteriormente implementar funciones propias basadas en interrupciones usando las funciones de la librería.
- ESP32Time y time: Desarrollada por Felix Biego, ESP32Time permite actualizar la estructura que contiene la fecha y hora actual del RTC interno del ESP32. A su vez, la librería “time”, estándar de C, permite acceder a estos valores de fecha y hora.
- PubSubClient: Desarrollada por Nick O’Leary, permite implementar la comunicación como cliente MQTT de forma sencilla en microcontroladores. Facilita la suscripción y publicación de datos a tópicos ubicados en un broker.
- ArduinoJson: Desarrollada por Benoit Blanchon, se complementa con la librería PubSubClient, permitiendo generar cadenas JSON para facilitar el envío de datos mediante MQTT.

Programación de la aplicación de monitoreo y control

Es conveniente contar con plataformas de monitoreo y control que sean de fácil acceso a los operadores del sistema para poder acceder a la información de la estructura de forma rápida y sencilla.

Es común conseguir plataformas o aplicaciones propias de un fabricante. Estas suelen ser limitadas y, al ser propiedad del fabricante, tienen poco espacio para personalizarlas según la aplicación o caso de uso lo requiera.

Se buscaba implementar una aplicación sencilla que permitiera a los operadores acceder a los datos, almacenarlos en una ubicación deseada, que provea un formato de datos replicable y compatible con distintos programas, que permitiera acceder a los valores claves para el monitoreo de la salud estructural y que diera la posibilidad de modificarse según la aplicación lo requiera. Para esto, se diseñó una GUI utilizando Python en conjunto con la librería “CustomTkinter”.

El programa, alojado en el computador y corriendo al estar el intérprete de Python instalado en el mismo, es capaz de enviar y recibir información vía MQTT usando la librería “paho-mqtt”. Se implementó una interfaz gráfica de usuario utilizando “customTkinter” la cual provee distintos botones, widgets, menús y gráficas que permiten monitorear y controlar el sensor inteligente a distancia.

A continuación se mencionan las librerías de Python 3.11.3 utilizadas en el programa para procesar y mostrar los gráficos una vez se accede a los archivos guardados en el computador, además del uso que se le da en la GUI:

- CustomTkinter 5.2.2: Provee las herramientas para la creación de la GUI. Crea la ventana y subventanas, permite crear los distintos botones y menús, además de adjuntar gráficas y texto.
- Numpy 1.24.3: Permite ejecutar cálculos matemáticos y provee las funciones necesarias para llevar a cabo la FFT de los datos obtenidos del archivo leído. Permite manejar los arreglos de datos como matrices.
- Matplotlib 3.7.1: Es la responsable de graficar los datos de aceleración, inclinación, densidad espectral de potencia y el espectro en frecuencia a partir de arreglos de datos.
- Pandas 2.2.1: Permite acceder a los datos guardados como archivo CSV (“comma-separated-values”) y guardarlos para su posterior procesamiento.

- Scipy 1.11.1: Permite acceder a distintos tipos de filtro para preprocesar los datos antes de ejecutar la FFT.
- Paho-mqtt 2.0.0: Provee las herramientas para crear un cliente MQTT capaz de suscribirse y publicar información a tópicos. Esta herramienta es la que permite recibir notificaciones sobre paquetes nuevos y enviar comandos al microcontrolador de la estación base para su posterior envío vía LoRa al sensor inteligente.

Inicialización y setup

Antes de comenzar con las rutinas de toma de datos, el sensor inteligente debe inicializar el hardware y llevar a cabo tareas de calibración y eliminación del offset, además en la creación de tareas y colas. En primer lugar, las colas, encargadas de enviar información entre tareas como se describió en el apartado 2.5.4, que fueron implementadas en el sensor inteligente son:

- data_temphumQueue
- aclQueue
- bufferQueue
- incQueue
- tramaLoRaQueue
- temphumarrayQueue
- incarrayQueue

Estas colas permiten que se envíe la información adquirida por una tarea, como puede ser obtener los valores leídos por el sensor mediante el objeto correspondiente a su clase, a otra tarea encargada de almacenarla en un buffer temporal para su posterior envío.

Una vez creadas las colas, se inicializa el hardware del sensor inteligente con las siguientes funciones:

- `bme.begin(0x76)`: Inicializa el sensor BME280 usando el objeto `bme`. Dentro de esta rutina de la librería se definen los parámetros como modo de operación, cantidad de sobremuestreo en datos, duración del tiempo de standby.
- `setup_acl_MPU6050()`: Esta función inicializa el sensor MPU6050, configura su tiempo de muestreo a 200 Hz, enciende el led de calibración amarillo, lleva a cabo la función de self test y verifica que el valor se encuentre dentro de los valores nominales de fabricación. Posteriormente, se elimina el offset en todos los ejes al tomar 1000 mediciones y tomar el promedio de estas aceleraciones. Se inicializan los rangos del acelerómetro y se desactiva el filtro pasa-altos ya que, según la documentación del MPU6050 de Invensense, esto introduce retardo en las mediciones.
- `setup_mpu9250()`: Esta función, análoga a las funciones anteriores, inicializa el MPU9250 usando el objeto creado perteneciente a la clase definida en la librería MPU9250. Se ejecuta una rutina de eliminación de offset similar a la del MPU6050 en donde se toman mediciones por un intervalo de tiempo, se promedian, y posteriormente se guardan en los registros de “bias” del dispositivo.
- `setup_lora_radiolib()`: La función de inicialización del módulo LoRa Ra-02 se encarga de configurar la frecuencia de trabajo, el factor de propagación y el ancho de banda. Luego de esto, envía un mensaje de inicialización en

donde le hace saber al dispositivo de la estación base que el sensor inteligente ya se encuentra operativo y con sus sensores listos para las mediciones. Luego, se activa la interrupción (ISR por sus siglas en inglés), y se configura la función de callback a ejecutar en caso de que la señal cambie de 0 V a un valor positivo en el pin 2 del microcontrolador. Esta ISR es la que permite manejar el caso en el que el módulo reciba una señal desde estación base. Una vez configurada la interrupción, se configura el módulo tal que escuche a mensajes presentes en el espectro.

- `Wire.setClock(400000)`: Ajusta la frecuencia de las comunicaciones I2C del microcontrolador a 400 kHz.

En cuanto a la estación base, la inicialización consta de las siguientes funciones de configuración:

- `setup_wifi()`: Se encarga de conectarse a una red WiFi cuyas credenciales se encuentran definidas en un archivo `wifi_credentials.h`. Si no logra conectarse, seguirá intentando hasta lograr la conexión a la red especificada.
- `setup_mqtt()`: Si la conexión a la red WiFi es exitosa, el microcontrolador de la estación base se conecta al broker ubicado en la dirección IP especificada y asignada al computador donde está alojado el broker MQTT. Una vez ahí, se suscribe a un tópico al cual envía mensajes periódicos para que no se pierda la conexión. Finalmente, configura el tamaño del buffer del cliente a 40000 bytes.
- `update_timepacket()`: Esta función tiene como objetivo actualizar el reloj RTC interno del ESP32 usando el protocolo NTP, con el cual el microcontrolador solicita a un servidor la fecha y hora actual, el servidor responde y esta información es almacenada en el ESP32.

Una vez el hardware está configurado, se procedió a crear las tareas que se ejecutarán por el sensor inteligente. Para esto, se hace uso del comando `xTaskCreatePinnedToCore` de la siguiente forma:

```
1 BaseType_t xTaskCreatePinnedToCore(TaskFunction_t pvTaskCode,
    const char * const pcName, const uint32_t usStackDepth, void
    * const pvParameters, UBaseType_t uxPriority, TaskHandle_t *
    const pvCreatedTask, const BaseType_t xCoreID);
```

Código 3.1. Creación de tareas en FreeRTOS

A todas las tareas se les asigna una prioridad, un handle, el núcleo donde se ejecutarán y el espacio de memoria que pueden ocupar dependiendo de la que necesite una vez se ejecute.

Estructuras de datos del sensor inteligente

Se crearon distintas estructuras de datos para manejar las mediciones tomadas por los sensores y además permitir la comunicación entre tareas mediante colas, manteniendo los datos ordenados para posteriormente enviarlos a larga distancia hacia y desde la estación base. Las estructuras más relevantes fueron:

- **BufferACL:** Estructura que almacena los datos provenientes del sensor MPU6050 para enviarlos por paquetes utilizando el módulo de comunicaciones. Consta de 3 arreglos de 1024 flotantes cada uno.
- **BufferTempHumedad:** Almacena los valores de temperatura y humedad obtenidos por el sensor BME280. Estos valores son promediados previos al almacenamiento y además pasan por un filtro de primer orden que le asigna un mayor peso a la medición anterior para evitar cambios bruscos.

- BufferInclinacion: Almacena los valores de inclinación estimados por el sensor MPU9250 luego de implementar el Filtro de Madgwick.
- Packet: Corresponde a la estructura principal del sistema, ya que es la encargada de almacenar los distintos paquetes que surgen producto de la subdivisión del arreglo de flotantes de aceleraciones. Este paquete consta de los bytes de encabezado y el payload (los datos de aceleración adquiridos).
- Packet2: Sigue la estructura de Packet con la diferencia de tener un payload de tamaño 1. Este paquete se corresponde con los mensajes de comandos desde y hacia estación base, los cuales constan de 1 solo byte.
- TimePacket: Se utiliza para actualizar el reloj RTC interno del ESP32 del sensor inteligente con la fecha y hora obtenida usando el protocolo NTP en la estación base.
- StringPacket: Es la encargada de almacenar el string de inicialización enviado por el sensor inteligente una vez termina su inicialización y notifica a la estación base que está listo para actualizar su RTC y tomar datos.
- THIPacket: Siguiendo la estructura de los paquetes anteriores, esta estructura almacena los valores promedio de inclinación, temperatura y humedad que estaban almacenados en los buffers BufferInclinacion y BufferTempHumedad. Además, se incluye en esta estructura el timestamp correspondiente al momento en el cual se comienza a tomar el registro de datos de aceleración en formato “UNIX Epoch”, que corresponde al número de segundos transcurridos desde el 1 de enero de 1970, siendo este formato más sencillo de enviar y luego reconvertir en la estación base al formato de fecha y hora más adecuado.

Programas del sensor inteligente

A continuación se describe el funcionamiento de las tareas y funciones para la adquisición y preprocesamiento de los datos:

- Tarea de adquisición de datos de aceleración: Esta tarea es una de las más importantes del sensor inteligente. Se encarga de adquirir las mediciones de aceleración en los intervalos de tiempo especificados por el macro `F_SAMPLING`. En esta tarea se verifica si los valores sobrepasan el límite de aceleración en alguno de los 3 ejes, también verifica si es hora de adquirir un registro según el reloj RTC interno del ESP32 y por último se asegura de que la bandera `flag_acl`, la cual corresponde a una petición de datos vía LoRa desde la estación base, no se haya activado. En cualquiera de estos casos, la tarea envía los últimos 3 valores de aceleración registrados a una cola, la cual es recibida por una función que se encarga de almacenar estos valores en un buffer sucesivo hasta que se completan según el tamaño de registro fijado.
- Tarea de creación de buffer: Esta rutina es de gran importancia para el control y la sincronización de los eventos del sensor inteligente. Esto se debe a que es la encargada de evaluar si el buffer de datos ya fue llenado en su totalidad por los datos provenientes de la función de adquisición. De ser así, desactiva las tareas de adquisición temporalmente y activa la rutina de envío de datos por LoRa, la cual se encargará de reiniciar las tareas de adquisición. Además, reinicia todas las banderas límite para que el sensor inteligente esté alerta a nuevos eventos una vez se envíen todos los paquetes.
- Evaluación de valores límite: La evaluación de los valores límite consiste en comparar los últimos valores de aceleración escogidos y levantar una bandera en caso de que alguno de los ejes sobrepase el límite de $2[\frac{m}{s^2}]$.

- Verificación de hora actual: Esta función se encarga de verificar si la hora actual del microcontrolador, a la cual se puede acceder a través del objeto `rtc` de la librería `ESP32Time`, se corresponde con alguna de las horas previamente programadas en el sistema. De ser así, levanta una bandera para que el sensor inteligente comience a tomar un registro de datos.
- Tarea de adquisición de datos de temperatura y humedad: Esta tarea se encarga de adquirir los valores de temperatura y humedad del sensor `BME280` y enviarlos mediante una cola a la función encargada de ejecutar los promedios y aplicar el filtro de primer orden a los datos para posteriormente llenar el buffer temporal de temperatura y humedad de forma sucesiva.
- Tarea de estimación de datos de inclinación: Análogo a las tareas de adquisición de aceleración, temperatura y humedad, esta tarea es la encargada de adquirir los valores de estimación de inclinación por parte del sensor `MPU9250` para posteriormente, si se está tomando un registro de datos, enviar estos valores, usando una cola, a una función encargada de almacenar los valores promedio en un buffer temporal.

A continuación se describen las tareas encargadas de la comunicación de los datos vía LoRa en el sensor inteligente:

- Rutina de interrupción: La rutina de interrupción o ISR es la encargada de reactivar la tarea de recepción de datos cuando en el pin asignado para la ISR se detecta un flanco ascendente. Estas rutinas se ejecutan en RAM por lo que requieren el macro `ICACHE_RAM_ATTR` específico para ESP32. Es decir, esta rutina siempre se ejecuta independientemente de la tarea que se esté ejecutando en el microcontrolador. Es importante acotar que se programó el sistema para desactivar esta interrupción mientras se está tomando un registro de datos. Esto impide que la recepción de un mensaje interrumpa

la toma de los datos que ya está ejecutándose. A continuación, en el código 3.2, se observa la ISR implementada:

```
1      void ICACHE_RAM_ATTR setFlag(void){  
2          //Activo tarea de recepcion de datos  
3          vTaskResume(xHandle_receive_task);  
4          receivedFlag = true;  
5      }  
6
```

Código 3.2. ISR de recepción de datos en sensor inteligente

La bandera de transmisión “receivedFlag” debe ser de tipo *volatile*, para que pueda ser modificada dentro de una ISR.

La interrupción se genera gracias a la configuración del módulo Ra-02 SX1278 de Ai-Thinker, el cual, al recibir un mensaje que cumpla con las características de encabezado de un mensaje LoRa, activa el pin DIO0 del módulo. Esta salida digital se conecta al pin configurado para detectar el flanco de subida y se genera la interrupción.

- Tarea de recepción de datos: Esta tarea es clave al ser la que decodifica la naturaleza del mensaje recibido desde la estación base. Dependiendo de la longitud del mensaje recibido, se identifica si el mensaje corresponde a un comando de adquisición o a una actualización de RTC. Si la longitud no se corresponde con ninguno de los 2 casos, la data está corrupta o no es para este receptor. Una vez se recibe el paquete con éxito, se verifica el payload y se levanta bandera de adquisición inmediata o se ejecuta función de actualización de RTC, según sea el caso.
- Tarea de envío de mensaje de inicialización: Es la encargada de enviar un string de inicialización a la estación base para notificar que el sensor inteligente ya culminó su período de inicialización y ajuste y está listo para una actualización de RTC.

- Función de actualización de RTC: Toma como entrada el arreglo de bytes recibido que contiene la información de la fecha y hora actual. La información se convierte a una estructura de datos de tipo TimePacket y se sincroniza el reloj interno del ESP32.
- Tarea de envío de paquetes de datos de aceleración: Una de las tareas de mayor importancia en todo el sistema al ser la encargada de gestionar el envío de los paquetes sucesivos de byte arrays a la estación base con la información almacenada en los buffers de aceleración correspondientes.
- Función para generar arreglo de datos a enviar: Función para generar el arreglo de subpaquetes (chunks) que contienen 128 bytes cada uno para su posterior envío, mediante una cola, a la tarea de envío de datos LoRa en forma de byte array.
- Tarea de envío de datos de temperatura, humedad e inclinación: Una vez se envían todos los datos de aceleración, siguiendo la misma estructura de envío de datos de aceleración, se envían los contenidos del buffer promedio.

Programas de la estación base:

- Rutina de interrupción: Análoga a la ISR implementada en el sensor inteligente, se encarga de ejecutar en RAM la activación de la tarea encargada de la recepción de los datos independientemente del código que se esté ejecutando.
- Tarea de envío de fecha y hora para actualización de RTC: Esta tarea es la encargada de enviar una estructura de tiempo, dentro de una estructura de tipo TimePacket, al sensor inteligente con el fin de sincronizar su reloj RTC interno. Para el envío del paquete, se llena la estructura de tiempo y luego se convierte a un arreglo de bytes usando la línea de código presente en

3.3, esto consiste en "castear" de un tipo de datos a otro. En este caso, es un cast de reinterpretación, permitiendo que esta conversión sea compatible en distintas arquitecturas. La notación es originaria del lenguaje C y permite efectuar operaciones de bajo nivel con el puntero que contiene el byte array.

```
1      byte *byteArrTime = (byte *)&time_packet;  
2
```

Código 3.3. Conversión de datos a tipo byte array usando casting

Una vez los datos están en el tipo byte array, el cual es aceptado por el módulo Ra-02 Ai-Thinker, los datos se envían utilizando el comando "startTransmit" de la librería RadioLib. Este comando recibe como entradas el arreglo de datos de toda la estructura a enviar, la cual contiene el encabezado y el payload, y el tamaño del arreglo de datos final. Este comando devuelve un código de respuesta indicativo del estado del envío, notificando sobre si el envío fue exitoso o no.

Una vez el arreglo de bytes es enviado con éxito, se debe introducir un retardo de 500 ms antes de configurar el módulo para que escuche comando nuevamente con el comando "startReceive", como se observa en la figura 3.4. La necesidad de este retardo surgió luego de observar que el módulo no recibía comandos si se configuraba para escuchar comandos inmediatamente después. Esto se debe a la necesidad del módulo de unos cuantos milisegundos para enviar el paquete exitosamente. Luego de enviar el paquete, se debe reiniciar la ISR para comenzar a escuchar paquetes nuevamente.

```
1      int state2 = radio.startReceive();  
2      attachInterrupt(digitalPinToInterrupt(2), setFlag,  
        RISING); // Reinicia ISR  
3
```

Código 3.4. Configuración del módulo LoRa para escuchar paquetes

- Rutina para recepción de mensajes por MQTT: Para poder recibir mensajes vía MQTT, primero debe definirse la función callback a llamarse cuando se recibe un mensaje vía MQTT desde el tópico al cual está suscrito el microcontrolador.

La función de callback está implementada como se observa en el código 3.5:

```

1      void messageReceived(char* topic, byte* payload,
2      unsigned int length) {
3
4          //Convierte payload a string
5          String mensaje;
6          for (int i = 0; i < length; i++) {
7              mensaje += (char)payload[i];
8          }
9
10         //Chequear si el mensaje recibido es "ON"
11         if (mensaje == "ON") {
12             //Llama a la ISR, petición de datos al Smart
13             Sensor
14
15             if(!transmitFlag){
16                 ISR_MQTT_Request(); //ISR
17             }
18         }
19     }

```

Código 3.5. Función de callback para recepción de mensajes MQTT

- Tarea de envío de comandos de adquisición al sensor inteligente:

Una vez se recibe vía MQTT el comando, ejecutado desde la GUI por el operador, de enviar un comando de adquisición de datos al sensor inteligente, se ejecuta la ISR, que se implementó como se observa en el código 3.6 que activa la envío de datos usando el módulo LoRa:

```

1  //Rutina de ISR por software en caso de recibir una
   petición por MQTT
2  void IRAM_ATTR ISR_MQTT_Request()
3  {
4      transmitFlag = true;
5      vTaskResume(xHandle_send_task);
6  }
7

```

Código 3.6. ISR de activación de envío de comando por LoRa

La tarea de envío de datos por LoRa, se encarga de enviar en el payload el byte 0x01, este payload es decodificado e identificado por el sensor inteligente y lo interpreta como un comando, activando la toma de datos.

Los comandos utilizados son análogos a los utilizados en el envío de datos para la actualización del RTC, definida anteriormente. Podrían numerarse de la siguiente forma:

1. Desactivar interrupción.
2. Llenar la estructura de datos a enviar.
3. Convertir la estructura de datos al tipo de datos byte array para su envío.
4. Enviar el paquete LoRa usando el comando “startTransmit” de la librería RadioLib.
5. Introducir retardo de 500 ms para que el módulo pueda enviar con éxito el paquete.
6. Reiniciar bandera de transmisión.
7. Configurar el módulo a modo “*listening*”.
8. Reactivar interrupción.

- Tarea de recepción de datos provenientes del sensor inteligente: Corresponde a la tarea principal de la estación base. Se encarga de recibir y almacenar los datos recibidos vía LoRa para su posterior subida al broker MQTT.

Para lograr esto, la tarea lleva a cabo los siguientes pasos:

1. Reinicia bandera de ISR.
2. Crea unión `packetUnion` para que contenga las distintas estructuras que pueden utilizarse.
3. Lee los datos recibidos y obtiene el tamaño del paquete.
4. Dependiendo del tamaño del paquete se activan distintas banderas que identifican el mensaje como datos de aceleración, datos de temperatura y humedad o un mensaje de inicialización del sensor inteligente.
 - Caso 0 (Datos de aceleración): Se guardan los arreglos de datos en buffers sucesivos, los cuales se identifican dependiendo de la identificación presente en el encabezado del mensaje. El número de paquetes está fijado por el número de bytes que se envían en cada payload. Una vez se reciben todos los paquetes, se envían los buffers en una cola para su posterior envío vía MQTT.
 - Caso 1 (Mensaje de inicialización de sensor inteligente): Al recibir un mensaje de inicialización del sensor inteligente, esto significa que el sensor culminó su proceso de ajuste e inicialización y está listo para sincronizar su RTC. En este caso, se identifica el `messageID` del mensaje y si cumple con las características de la inicialización del RTC, se envía el mensaje correspondiente.
 - Caso 2 (Datos de temperatura y humedad): Análogo al caso 0, si se identifica el mensaje como uno que corresponde a datos de temperatura y humedad a partir de se almacenan los datos de temperatura humedad e inclinación en una estructura dedicada.

Esta se envía mediante una cola a la tarea encargada de subir los datos al broker MQTT.

5. En caso de tratarse de datos corruptos cuya CRC no corresponda con la esperada por el receptor, se descartan.

- Tarea de envío de datos de temperatura, humedad e inclinación al broker vía MQTT:

Esta tarea tiene como objetivo subir los datos recibidos mediante una cola, enviada por el RTOS luego de recibir los datos de temperatura, humedad e inclinación, al broker ubicado en el computador cuya dirección IP es conocida. Para lograrlo se llevaron a cabo los siguientes pasos:

1. Suspende temporalmente la tarea `keepalive_task`, encargada de enviar mensajes cortos de forma periódica al broker y mantener activa la conexión mientras no se están enviando datos.
2. Crea estructura de datos de tipo `THIPacket`.
3. Recibe la cola con los datos.
4. Ejecuta la función `sendTHI`, la cual recibe como entrada el tópico y el dato a enviar. Esta función convierte los datos a tipo string y luego los publica en el tópico de interés.
5. Activa la tarea para envío de datos de aceleración
6. Se suspende a sí misma.

- Tarea de envío de arreglos de datos de aceleración al broker vía MQTT:

Similar a la tarea anterior, se encarga de subir los datos de aceleración recibidos mediante una cola al broker MQTT. Se llevaron a cabo los siguientes pasos para subir exitosamente los daots:

1. Recibe la cola con los arreglos de aceleración y los almacena en una estructura de datos.

2. Ejecuta la función “sendAxis”, similar a “sendTHI”, la cual recibe como entrada el tópico, el nombre del eje que está siendo enviado, el arreglo de datos almacenado en la estructura y el tamaño del arreglo. Esta función crea un “nestedArray”, es decir, crea nuevo array JSON anidado. El arreglo se llena con los datos de forma sucesiva utilizando un bucle. Luego, serializa el documento JSON a una cadena de texto. Finalmente, la representación en texto del documento JSON es publicada al tópico de interés.
3. Reinicia la ejecución de la tarea `keepalive_task`.
4. Se suspende a sí misma hasta que sea reanudada en el próximo envío.

Aplicación de monitoreo y control:

A continuación se describen las funciones del sistema de monitoreo y control:

- **Conexión MQTT:** La aplicación se conecta al broker MQTT ubicado en la dirección IP del computador. Para lograrlo, se ejecuta la función *connect* de la librería *paho-mqtt*, la cual recibe como entradas la dirección IP del broker, el puerto de conexión y el nombre del cliente. Una vez conectado, se suscribe y publica a los tópicos de interés.
- **Petición de datos a sensor inteligente:** La aplicación permite al operador solicitar datos al sensor inteligente mediante el envío de un mensaje vía MQTT al tópico encargado de gestionar el envío de comandos al sensor inteligente. Para lograrlo, se ejecuta la función *publish* de la librería *paho-mqtt*, la cual recibe como entradas el tópico de interés y el mensaje a enviar.
- **Gráfica de aceleración en tiempo:** Para las gráficas se hizo uso de la librería *matplotlib*, la cual proporciona las herramientas necesarias para mostrar

gráficamente series de datos que pueden estar almacenados en arreglos. En el caso de la aceleración, se lee el registro escogido por el operador en formato “csv” y se almacenan las columnas correspondientes en arreglos de la librería “NumPy”. El eje x, que representa el tiempo, se construye a partir de la frecuencia de muestreo y el número de muestras. Luego, se grafican los datos en función del tiempo.

- Gráfica de espectro en frecuencia: Ejecutando la FFT sobre los datos de aceleración almacenados en arreglos de NumPy, se obtiene el espectro en frecuencia del registro escogido. El eje x debe truncarse para graficar solo las frecuencias positivas. Se incluyen en la gráfica indicadores que apuntan a los valores máximos por eje.
- Valores de temperatura, humedad e inclinación: En el archivo escogido por el operador se encuentran los valores de las variables ambientales y la inclinación en columnas separadas. Estos datos se almacenan y se agregan a la ventana.
- Gráfica de barras para inclinación: Los valores de inclinación utilizados para ser mostrados en la ventana, se usan para construir un gráfico de barras que permite al operador saber si el sensor inteligente se encuentra a nivel en un rango dado. Además, en caso de no estar dentro del rango deseado, se notifica sobre el ángulo que debe ajustarse para que el sensor inteligente esté a nivel.
- Gráfica de Densidad Espectral de Potencia: Similar al proceso para obtener la gráfica del espectro en frecuencia, se ejecuta la densidad espectral de potencia elevando al cuadrado los arreglos que contienen los espectros, graficando posteriormente estos valores contra el número de datos obtenidos en frecuencia.

3.3.5. Diagrama de flujo del sistema

3.3.6. Diagrama de funcionamiento del sistema

CAPÍTULO IV

PRUEBAS Y RESULTADOS

En el siguiente capítulo se presentan las pruebas y los resultados obtenidos a partir de la metodología descrita en el capítulo anterior. Se presentan las pruebas preliminares de comunicaciones para determinar las características óptimas del canal, las pruebas para la estimación de la inclinación que permitieron escoger el método de estimación apropiado y finalmente las pruebas de funcionamiento del prototipo.

4.1. Pruebas de comunicaciones

Una vez escogido el hardware propuesto en la sección 3.2, se llevaron a cabo una serie de pruebas para comprobar el funcionamiento de los módulos de comunicaciones y para evaluar las características más adecuadas para el canal de comunicaciones.

Como se definió en el apartado 2.10.4, el protocolo LoRa implementado en el módulo Ra-02 de Ai-Thinker requiere fijar el valor de los siguientes parámetros:

- Factor de propagación.
- Ancho de banda.
- Potencia.

- Tasa de codificación.

La prueba consistió en el envío de un paquete de bytes a una distancia de aproximadamente 115 metros, sin línea de vista y con obstáculos de acero y concreto, como se observa en el mapa de la figura 4.1.

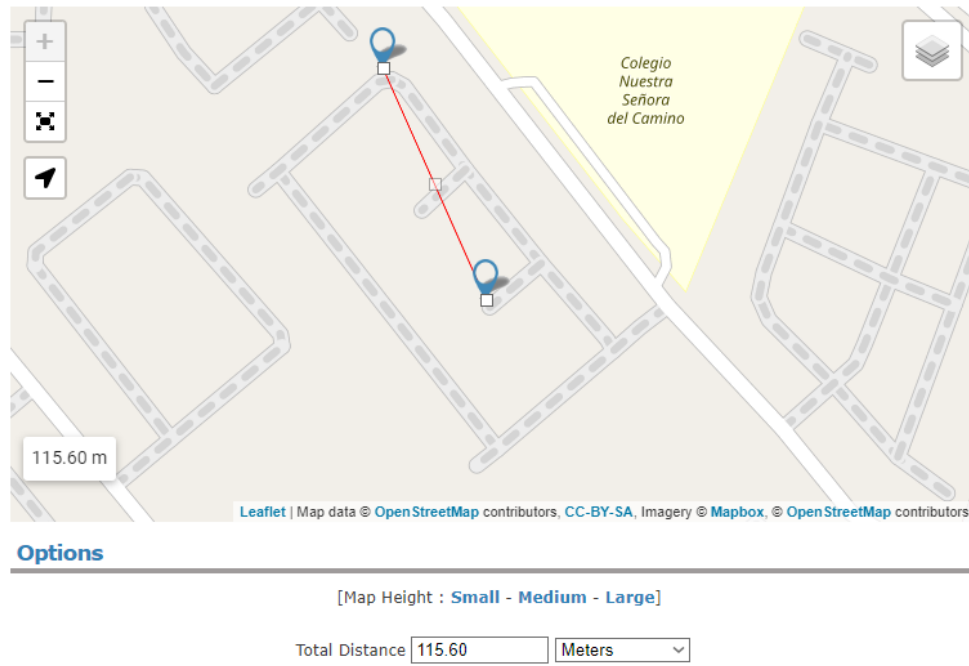


Figura 4.1. Vista en mapa de distancia máxima de pruebas usando módulo SX1278.

Se modificaron los parámetros del canal y se midió la tasa de paquetes con errores o paquetes corruptos en el receptor. Se realizaron pruebas con los siguientes parámetros fijos:

- Frecuencia de operación: 433 MHz.
- Tasa de codificación:
- Potencia: 10 dBm.

- Longitud del preámbulo: 8 bytes.
- Tamaño de la carga útil (payload): 128 bytes.

Estos parámetros son recomendados por el fabricante del módulo y se mantuvieron constantes durante las pruebas. Estos se confirmaron tras pruebas preliminares en las cuales se observó que para *payloads* mayores a 128 bytes el porcentaje de paquetes cuyo CRC era erróneo (data corrupta) aumentaba considerablemente, siendo 128 bytes un valor que disminuía esta proporción. Los parámetros que se modificaron fueron el factor de propagación, el ancho de banda y el período de envío entre paquetes. Los resultados de las pruebas se presentan en la tabla 4.1.

Tabla 4.1. Resultados de pruebas realizadas con módulo de comunicaciones Ra-02.

Configuración	F. de Propagación	Ancho de banda	Período	Tasa de paquetes perdidos
1	9	125 kHz	500 ms	105/500
2	7	250 kHz	150 ms	1/500
3	8	125 kHz	150 ms	300/500
4	8	250 kHz	500 ms	2/500
5	8	250 kHz	200 ms	Error de CRC
6	7	250 kHz	250 ms	2/500
7	7	250 kHz	200 ms	2/500
8	7	250 kHz	150 ms	2/500
9	7	250 kHz	100 ms	Error de CRC

Basados en estos resultados, se escogió la configuración 8 para las pruebas de comunicaciones, ya que es la que presenta la menor tasa de paquetes corruptos a la velocidad más alta de envío de paquetes. Esta configuración se utilizó para las pruebas de estimación de inclinación y para las pruebas de funcionamiento del prototipo.

4.2. Pruebas para estimación de inclinación

Para escoger el método de estimación de ángulos, se compararon los siguientes:

- Cálculo trigonométrico a partir de mediciones de acelerómetro.
- Filtro de Kalman.
- Filtro de Madgwick.

4.3. Pruebas de funcionamiento del prototipo

CAPÍTULO V

CONCLUSIONES

La revisión bibliográfica mostró los avances realizados por distintos autores en cuanto a la implementación de sistemas de monitoreo de variables estructurales basados en microcontroladores y haciendo uso de canales de comunicación inalámbrica, lo cual permitió concentrar y canalizar el trabajo de investigación de forma efectiva.

Se logró plantear el diseño de un sensor inteligente para aplicaciones de monitoreo de salud estructural con éxito, identificando las variables de interés para este tipo de sistemas e incluso implementando un prototipo de pruebas funcional capaz de llevar a cabo ensayos similares los que se llevan a cabo en el Instituto de Materiales y Modelos Estructurales.

Se diseñaron con éxito los distintos programas necesarios para obtener registros de vibración y mediciones de variables cuasi-estáticas haciendo uso de un microcontrolador. El ESP32 en conjunto con FreeRTOS, el cual permitió controlar de forma efectiva y ordenada las distintas tareas, mostraron ser herramientas capaces de manejar el preprocesamiento y adquisición de los datos de forma exitosa.

Se implementó con éxito una interfaz de monitoreo y control capaz de enviar comandos de control y a la vez visualizar los registros tomados por el sensor inteligente, aplicando herramientas de procesamiento numérico con bajo costo

computacional dentro de la misma aplicación, con capacidades de personalización dependiendo del cliente o proyecto y con la posibilidad de guardar los registros en un formato compatible con la gran mayoría de programas de procesamiento.

Al haber tomado en cuenta la fecha y hora de adquisición de los datos, y por las características del protocolo LoRa y el control que permite tener sobre los dispositivos esclavos asignándoles números de identificación única, se sentaron las bases a la posibilidad de escalar el número de sensores inteligentes de bajo costo en la estructura, aumentando la resolución espacial y permitiendo tener acceso a las gráficas del comportamiento modal de la estructura.

A pesar de las limitaciones de memoria de sistemas basados en microcontroladores, el sensor inteligente cumple sus funciones de monitorear las variables estructurales de interés; permitiendo que el operador tenga acceso a un histórico de datos de la estructura, disminuyendo además los costos y la complejidad de sistemas cableados con fines similares.

En este tipo de aplicaciones es esencial garantizar la integridad de los datos. A pesar de ser LoRa un protocolo lento en comparación a otros, la seguridad que ofrece al tener muy bajo porcentaje de pérdida de paquetes y el largo alcance, que permite ubicar la estación base en una ubicación segura alejada de la estructura lo posicionan como un protocolo prometedor para aplicaciones de monitoreo de salud estructural.

Los sistemas de adquisición de datos basados en microcontroladores representan una opción confiable y de bajo costo para implementar soluciones de monitoreo estructural periódico que sustituyen a los sistemas cableados.

CAPÍTULO VI

RECOMENDACIONES

Apéndice I

TÍTULO DEL ANEXO

Apéndice II

TÍTULO DEL ANEXO

Apéndice III

TÍTULO DEL ANEXO

REFERENCIAS

- Abdo, M. (2014). *Structural health monitoring, history, applications and future. a review book*. Open Science Publishers.
- Alfaifi, A., y Zaman, A. (2021). Mems humidity sensors. *Humidity Sensors-Types and Applications*.
- Aloufi, K. S., Alhazmi, O. H., y cols. (2020). A hybrid iot security model of mqtt and uma. *Communications and Network*, 12(04), 155.
- Andresen, S., Bäger, A., y Hamm, C. (2019, 10). Eigenfrequency maximisation by using irregular lattice structures. *Journal of Sound and Vibration*, 465, 115027. doi: 10.1016/j.jsv.2019.115027
- Balageas, D., Fritzen, C.-P., y Güemes, A. (2010). *Structural health monitoring* (Vol. 90). John Wiley & Sons.
- Basset Salom, L. (2014). Análisis estático de estructuras planas. *Universitat Politècnica de València*. oai:riunet.upv.es:10251/38538.
- Blackman, J. (2019). *Lpwa matchup / lorawan vs. sigfox vs. nb-iot vs. lte-m: technical draw (round 5)*. Descargado de <https://www.rcrwireless.com/20190829/carriers/lpwa-matchup-round-five> (Consultado: 02/05/2024)
- Blanco, M. (2012). Criterios fundamentales para el diseño sismorresistente. *Revista de la Facultad de Ingeniería Universidad Central de Venezuela*, 27(3), 071–084.
- Chen, H.-P. (2018). *Structural health monitoring of large civil engineering structures* (Vol. 1). John Wiley & Sons Ltd.
- Comisión Nacional de Telecomunicaciones. (2010). *REFORMA DE LAS CONDICIONES PARA LA CALIFICACIÓN DE LOS EQUIPOS DE USO*

- LIBRE*. Norma. Descargado de http://www.conatel.gob.ve/files/consulta/2011/2_PA_EUL_CP_PROYECTO_WEB.pdf
- Dunn, W. (2005). *Introduction to instrumentation, sensors, and process control*. Artech.
- Enckell, M. (2006). *Structural health monitoring using modern sensor technology: long-term monitoring of the new årsta railway bridge* (Tesis Doctoral no publicada). KTH.
- Farrar, C. R., y Worden, K. (2007). An introduction to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851), 303–315.
- Federici, R., Fabio y Alesii. (2014). Design of wireless sensor nodes for structural health monitoring applications. *Procedia Engineering*, 87, 1298–1301.
- Francone, M., Domenicali, D., y Di Benedetto, M.-G. (2006, 12). Time-varying interference spectral analysis for cognitive uwb networks. En (p. 3205 - 3210). doi: 10.1109/IECON.2006.348076
- Frank, R. (2002). Understanding smart sensors 2nd edn. *Measurement Science and Technology*, 13(9), 1501–1502.
- FreeRTOS. (2024). *Freertos documentation*. Descargado de https://www.freertos.org/Documentation/RTOS_book.html (Consultado: 18/01/2024)
- Fritzen, C. P. (2005). Vibration-based structural health monitoring—concepts and applications. *Key Engineering Materials*, 293, 3–20.
- García-Fernandez, M., Natalia y Aenlle. (2023). A review on fatigue monitoring of structures. *International Journal of Structural Integrity*, 14(2), 133–165.
- Hearn, G., y Testa, R. B. (1991). Modal analysis for damage detection in structures. *Journal of structural engineering*, 117(10), 3042–3063.
- Hibbeler, R. C., y Nolan, G. (1997). *Structural analysis*. Prentice Hall Upper Saddle River, New Jersey New Jersey.
- Hurtado, J. E. (2000). Introducción a la dinámica de estructuras. *Universidad*

Nacional de Colombia SEDE Manizales.

- Irvine, T. (2000). An introduction to frequency response functions. *Rapport, College of Engineering and Computer Science, 2000.*
- Jin, S.-S., Jeong, S., y Sim. (2021, 06). Fully automated peak-picking method for an autonomous stay-cable monitoring system in cable-stayed bridges. *Automation in Construction, 126*, 103628. doi: 10.1016/j.autcon.2021.103628
- Khorsandi, K., y Jalalizad, S. (2023). *Performance evaluation of lora networks for air-to-ground communications.*
- Komarizadehasl, S. (2022). Development of low-cost sensors for structural health monitoring applications. *Sensors*. Descargado de <http://hdl.handle.net/2117/379464>
- Lamkin-Kennard, K. A., y Popovic, M. B. (2019). Sensors: Natural and synthetic sensors. *Biomechatronics*, 81–107.
- Li, J., Deng, J., y Xie, W. (2015, 04). Damage detection with streamlined structural health monitoring data. *Sensors, 15*, 8832–8851. doi: 10.3390/s150408832
- Madgwick, S., y cols. (2010). An efficient orientation filter for inertial and inertial/magnetic sensor arrays. *Report x-io and University of Bristol (UK), 25*, 113–118.
- Montagny, S. (2021). Lora–lorawan and internet of things. *Université Savoie Mont Blanc. Available on: https://www.univ-smb.fr/lorawan/en/free-book/*. (Accessed on 23.03. 2022).
- Muttillo, M. (2019). Structural health continuous monitoring of buildings—a modal parameters identification system. En *2019 4th international conference on smart and sustainable technologies (splitech)* (pp. 1–4).
- Nagayama, B. F., Tomonori y Spencer Jr. (2007). Structural health monitoring using smart sensors. *Newmark Structural Engineering Laboratory Report Series 001.*

- Ogdol, J. M., Ogdol, G., y Samar, B.-L. (2018, 12). Iot-based framework for a centralized monitoring of solid waste disposal facilities.
- Proakis, J. G., y Manolaki, D. G. (1999). Digital signal processing. principles, algorithms, and applications. *Journal of Computer Science and Technology*, 1(1), 1.
- Rytter, A. (1993). Vibration based inspection of civil engineering structures ph. d. *Aalborg University, Aalborg, Denmark*.
- Semtech. (2015). *Sx1276/77/78/79 - 137 mhz to 1020 mhz low power long range transceiver datasheet*. Descargado de <https://www.semtech.com/products/wireless-rf/lora-connect/sx1278> (Consultado: 10/01/2024)
- Shi, Z., Law, S., y Zhang, L. (1998). Structural damage localization from modal strain energy change. *Journal of sound and vibration*, 218(5), 825–844.
- Siemens. (2019). *What is the fourier transform?* Descargado de <https://community.sw.siemens.com/s/article/what-is-the-fourier-transform> (Consultado: 04/05/2024)
- Sohn, H. (2007). Effects of environmental and operational variability on structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851), 539–560.
- Survey, U. S. G. (2011). *Earthquake hazards program*. Descargado de <http://earthquake.usgs.gov/earthquakes/shakemap/background.php> (Consultado: 04/05/2024)
- Systems, E. (2016). *Esp-idf programming guide*. Descargado de <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/index.html> (Consultado: 15/11/2024)
- Worden, K., y Friswell, M. I. (2009). Modal-vibration-based damage identification. *Encyclopedia of Structural Health Monitoring*.
- Zhang, W., Sun, L., y Sun, S. (2017). Bridge-deflection estimation through inclinometer data considering structural damages. *Journal of Bridge*

Engineering, 22(2), 04016117.