Universidad Rey Juan Carlos

ESCUELA DE INGENIERÍA DE FUENLABRADA

INGENIERÍA EN SISTEMAS AUDIOVISUALES Y MULTIMEDIA

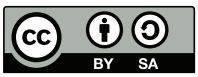
TRABAJO FIN DE GRADO

MEDICIÓN DE EFICIENCIA ENERGÉTICA EN LA NAVEGACIÓN WEB

Autor: Sandra Nicole Solórzano Carcelén

Tutor: Dr. Jesús María González Barahona

Curso académico 2024/2025



©2025 Sandra Nicole Solórzano Carcelén

Algunos derechos reservados

Este documento se distribuye bajo la licencia

“Atribución-CompartirIgual 4.0 Internacional” de Creative Commons,

disponible en

https://creativecommons.org/licenses/by-sa/4.0/deed.es

Dedicado a

mi familia, gracias por estar ahí.

II

Agradecimientos

Quisiera expresar mi más profundo y sincero agradecimiento a mi familia, cuyo apoyo incondicional ha sido el pilar esencial que me sostuvo a lo largo de este largo camino.

En especial, a mi padre, por su guía y sus sabios consejos; a mi hermana, por ser mi pilar moral, firme e inquebrantable; y a mi mami, por ser la luz constante que iluminó cada paso que di en este recorrido.

Mi reconocimiento también va para Alba y Ericson, sin cuya ayuda invaluable la culminación de este grado no habría sido posible. Y, de manera muy especial, agradezco a mi pareja, quien ha sido mi apoyo inquebrantable, mi refugio y mi fuerza constante, que me ha acompañado en cada paso de este desafío.

Finalmente, agradezco a los profesores que me inspiraron con su conocimiento, a los compañeros con quienes compartí risas y aprendizajes, y, muy especialmente, a mi tutor de TFG, cuya orientación experta y apoyo constante fueron fundamentales para el desarrollo exitoso de este trabajo.

Resumen

El presente Trabajo de Fin de Grado tiene como objetivo principal analizar y cuantificar el consumo energético asociado a la navegación web, combinando mediciones obtenidas tanto a nivel del sistema operativo como mediante un enchufe inteligente que mide el consumo eléctrico total del equipo. El propósito es identificar patrones de consumo relacionados con diferentes páginas web y validar la fiabilidad de las fuentes de datos disponibles para evaluar su impacto energético.

Para llevar a cabo este análisis se desarrolló un sistema automatizado que, utilizando Python, controla la navegación secuencial por un conjunto de páginas web definidas en la entrada. Las mediciones energéticas se obtienen, por un lado, a través del sistema Linux mediante sensores proporcionados por `hwmon`, principalmente el sensor interno de la CPU; y por otro, mediante un enchufe inteligente Shelly que mide el consumo global del sistema. Los datos recopilados se procesan con herramientas de análisis y visualización como Pandas, Matplotlib y Seaborn para facilitar la interpretación y comparación.

Este proyecto se enmarca dentro del creciente interés por la eficiencia energética y la sostenibilidad en el ámbito tecnológico, donde la huella energética del uso diario de dispositivos y aplicaciones web se vuelve cada vez más relevante. Además, aborda un área multidisciplinar que integra conocimientos de informática, sistemas operativos, telemetría, y análisis estadístico, y puede contribuir a futuras iniciativas de optimización energética y concienciación ambiental en el sector audiovisual y tecnológico.

Summary

This Final Degree Project aims to analyze and quantify the energy consumption associated with web browsing by combining measurements obtained both at the operating system level and through a smart plug that records the total electrical consumption of the device. The main goal is to identify consumption patterns related to different websites and validate the reliability of available data sources to assess the energy impact of them.

To carry out this analysis, an automated system was developed using Python to sequentially navigate a set of websites defined in the input. Energy measurements are collected on one hand from Linux system sensors accessible via `hwmon`, primarily the internal CPU sensor; and on the other hand, from a Shelly smart plug measuring the overall system consumption. The gathered data is processed using analysis and visualization tools such as Pandas, Matplotlib, and Seaborn to facilitate interpretation and comparison.

This project is framed within the growing interest in energy efficiency and sustainability in the technology field, where the energy footprint of daily device and web application use is increasingly relevant. Additionally, it integrates multidisciplinary knowledge from computer science, operating systems, telemetry, and statistical analysis, potentially contributing to future energy optimization initiatives and environmental awareness efforts in audiovisual and technological sectors.

Índice general

1. Introducción **1**

1.1. Objetivos 1

1.1.1. Objetivo general 1

1.1.2. Objetivos específicos 2

1.2. Contribuciones del trabajo 2

1.3. Estructura del documento 3

1.4. Planificación temporal 4

2. Tecnologías utilizadas y trabajos relacionados **5**

2.1. Tecnologías utilizadas 5

2.1.1. Python 5

2.1.2. Kernel de Linux 8

2.1.3. Shelly Plug 9

2.1.4. Herramientas auxiliares 10

2.2. Trabajos relacionados 11

3. Desarrollo del proyecto **13**

3.1. Metodología de trabajo 13

3.1.1. Sprint 0: Análisis preliminar y definición del propósito 14

3.1.2. Sprint 1: Investigación sobre obtención de datos desde el sistema 14

3.1.3. Sprint 2: Experimentación con herramientas del sistema 14

3.1.4. Sprint 3: Primera integración con enchufe inteligente (Tuya) 15

3.1.5. Sprint 4: Primer sistema automatizado de medición y visualización . . 15

3.1.6. Sprint 5: Incorporación del enchufe inteligente Shelly 15

3.1.7. Sprint 6: Integración de fuentes de datos para análisis conjunto 16
3.1.8. Sprint 7-8. Validación cruzada y análisis de resultados 17
3.2. Arquitectura de funcionamiento 17
3.3. Implementación 19
3.3.1. Automatización de la navegación web 19
3.3.2. Extracción de datos del kernel de Linux 20
3.3.3. Obtención de datos con el enchufe inteligente Shelly Plug S 21
3.3.4. Sincronización y almacenamiento de datos 26
3.4. Procesamiento y representación gráfica 27
3.5. Reproducibilidad del sistema 28

4. Experimentos y validación **31**
4.1. Configuración experimental 31
4.2. Metodología de medición 32
4.2.1. Recopilación de datos 32
4.2.2. Escenarios de prueba 33
4.2.3. Análisis de resultados 34
4.2.4. Correlación entre mediciones 36
4.2.5. Conclusiones generales sobre el análisis estadístico 37
4.2.6. Limitaciones del sensor interno 40
4.2.7. Conclusión final 41

5. Resultados **43**
5.1. Ranking de webs más visitadas en el mundo 43
5.1.1. Análisis temporal del consumo energético 45
5.1.2. Relación entre el consumo energético total y el uso de la CPU 47
5.1.3. Conclusión de resultados 53

6. Conclusiones **55**
6.1. Valoración personal y conclusiones del proyecto 56
6.2. Consecución de objetivos 57
6.3. Aplicación de lo aprendido 58

6.4. Lecciones aprendidas 59
6.5. Trabajos futuros 60
6.5.1. Limitaciones y posibles mejoras 61

A. Código fuente **65**

B. Datos **71**

C. Gráficas y tablas complementarias **83**

Bibliografía **93**

Índice de figuras

3.1. Arquitectura del sistema 18
3.2. Proceso de navegación automática 20
3.3. Resultado con el comando sensors en el PC de prueba 23

4.1. Estadísticas de Hwmon para el fondo. 34
4.2. Estadísticas de Hwmon para la sesión. 35
4.3. Estadísticas de Shelly para el fondo. 35
4.4. Estadísticas de Shelly para la sesióno. 36
4.5. Correlación global entre Shelly y Hwmon 36

5.1. Energía total consumida por cada web en orden descendente 44
5.2. Páginas web analizadas. Las subfiguras (a) y (b) muestran ejemplos de los sitios con mayor consumo, mientras que (c) y (d) ilustran aquellos con el menor. . . . 45
5.3. Análisis temporal. Periodo = 60 s. 47
5.4. Análisis temporal. Periodo = 120 s. 48
5.5. Análisis temporal. Periodo = 240 s. 48
5.6. Correlación global entre uso de CPU y consumo energético total 53

C.1. Gráficas de correlación de uso de CPU vs. consumo total para cada sitio web . 91

Capítulo 1

Introducción

Nuestra dependencia a la tecnología crece al mismo ritmo que esta evoluciona. Está inmersa en actividades de nuestra vida cotidiana, como son el estudio, el trabajo o el entretenimiento. Entre todas ellas, hay una que podría destacarse, y es la navegación web. Consultar el tiempo, comprar ropa en línea o ver los resultados de un partido son solo algunos ejemplos del amplio abanico de utilidades que le damos a este servicio, el cual es usado diariamente por millones de personas en todo el mundo.

Sin embargo, pocas veces nos detenemos a pensar en el impacto energético que conlleva esta actividad aparentemente inofensiva, pero tan común. ¿Cómo influyen estas consultas en nuestro consumo eléctrico? ¿Existen diferencias entre una web y otra?

El objetivo de este proyecto es analizar el consumo de potencia eléctrica asociado a la navegación web y cómo determinados factores, como el diseño o la transferencia de datos, pueden influir en un mayor gasto energético. Demostrando que, aunque esta actividad pueda tener un bajo consumo al hacer una consulta individual, su acumulación puede reflejar valores significativos.

1.1. Objetivos

1.1.1. Objetivo general

Analizar el consumo de potencia eléctrica asociado a la navegación en diferentes páginas web mediante un sistema automatizado que navegue por cada una y mida el consumo desde una

fuente. Los datos obtenidos se almacenan para su posterior comparación y análisis, con el fin de determinar el impacto energético de cada web. De un conjunto determinado concluir cuál tiene mayor consumo energético, algunas causas y posibles soluciones.

1.1.2. Objetivos específicos

Se establecen los siguientes objetivos específicos, instrumentales y secundarios para el desarrollo del proyecto:

- Desarrollar un sistema automatizado en lenguaje Python que permita abrir una serie de páginas web de forma controlada.
- Extraer datos de consumo de potencia desde dos fuentes distintas: el kernel de Linux y un enchufe inteligente con medidor de potencia.
- Sincronizar y almacenar los datos obtenidos de ambas fuentes de manera estructurada para facilitar su análisis.
- Procesar los datos registrados con el fin de detectar variaciones en el consumo eléctrico durante la navegación.
- Representar gráficamente los resultados obtenidos para comparar el impacto energético de diferentes páginas web.

1.2. Contribuciones del trabajo

Este proyecto aporta varias contribuciones relevantes en el ámbito del análisis energético aplicado a la navegación web:

- Desarrollo de un sistema automatizado que integra la navegación secuencial por un conjunto de páginas web con la medición simultánea del consumo energético, facilitando así la recopilación de datos precisos y reproducibles.
- Implementación de una metodología dual de medición, combinando sensores internos del sistema operativo Linux con dispositivos externos (enchufe inteligente), lo que permite una validación cruzada y mejora la fiabilidad de los resultados.

- Análisis comparativo del consumo energético asociado a diferentes páginas web, proporcionando información útil para identificar patrones de consumo y posibles áreas de optimización en el diseño web orientado a la eficiencia energética.
- Generación de representaciones gráficas que permiten una visualización clara y comprensible del impacto energético de la navegación web, apoyando conclusiones fundamentadas.
- Contribución al creciente cuerpo de conocimiento sobre sostenibilidad y eficiencia energética en tecnologías de la información, enfocándose en un aspecto cotidiano y masivo como es la navegación en Internet.

Estas contribuciones sientan las bases para futuros trabajos que busquen profundizar en la optimización energética del software y el hardware en el ámbito digital.

1.3. Estructura del documento

El contenido de la memoria se distribuye de la siguiente manera:

- **Capítulo 1: Introducción**
  Se presenta el contexto general del proyecto, los objetivos planteados y la motivación que lo impulsa.
- **Capítulo 2: Tecnologías utilizadas y trabajos relacionados**
  Se detallan las tecnologías, herramientas y lenguajes empleados a lo largo del desarrollo, junto con una revisión de trabajos previos relevantes que permiten situar el proyecto en su marco teórico y técnico.
- **Capítulo 3: Desarrollo del proyecto**
  Se describe la arquitectura general del sistema, la metodología de trabajo adoptada y cada una de las fases de implementación, incluyendo la automatización de la navegación web, la recolección de métricas del kernel y la integración con el enchufe inteligente.
- **Capítulo 4: Experimentos y validación**
  Se explican los procedimientos experimentales diseñados para evaluar el comportamiento

energético del sistema ante una sola web, detallando la configuración utilizada, la metodología de medición y el proceso de validación de los datos obtenidos.

- **Capítulo 5: Resultados**

  Se analizan los datos recogidos durante la evaluación de un conjunto determinado de páginas web, presentando métricas estadísticas, visualizaciones gráficas y análisis de correlación que permiten interpretar los patrones observados en el consumo energético.

- **Capítulo 6: Conclusiones**

  Se resumen los hallazgos más relevantes del proyecto, se evalúa el grado de cumplimiento de los objetivos y se proponen posibles líneas de mejora y desarrollo futuro.

- **Capítulo 7: Anexo**

  Se incluyen materiales complementarios que apoyan el contenido del documento, como fragmentos de código, ejemplos de datos recogidos o configuraciones técnicas utilizadas durante la implementación y experimentación.

1.4. Planificación temporal

El desarrollo de este Trabajo Fin de Grado se ha llevado a cabo durante un período aproximado de seis meses, distribuidos principalmente entre las actividades académicas regulares y el trabajo específico del proyecto. Debido a la combinación con otras responsabilidades, gran parte del esfuerzo dedicado a este trabajo se concentró en fines de semana y periodos vacacionales, lo que requirió una planificación cuidadosa para cumplir con los objetivos en los plazos establecidos.

La planificación se estructuró en diferentes fases: investigación y revisión bibliográfica inicial, diseño y desarrollo del sistema automatizado, recopilación y procesamiento de datos, análisis de resultados y elaboración del informe final. Cada fase contó con un calendario orientativo que facilitó el seguimiento del progreso y la identificación temprana de posibles desviaciones.

Una descripción más detallada de cada fase se especifica en la Sección 3.1.

Capítulo 2

Tecnologías utilizadas y trabajos relacionados

En este capítulo se presentan las principales tecnologías empleadas en el desarrollo del proyecto, así como algunos trabajos y soluciones previas que guardan relación con el mismo.

Primero, se describen las herramientas y lenguajes utilizados para implementar el sistema, como el lenguaje de programación *Python*, el kernel de Linux y el enchufe para obtener los datos de potencia. También se mencionan otras tecnologías auxiliares que complementaron el desarrollo del mismo, como herramientas gráficas y entornos de desarrollo.

Conjuntamente, se exponen una serie de trabajos relacionados que han servido como referencia para el planteamiento y la ejecución de este proyecto.

2.1. Tecnologías utilizadas

2.1.1. Python

Python[12] es un lenguaje de programación interpretado, de alto nivel y propósito general, que destaca por su sintaxis clara y expresiva. Esta característica facilita un desarrollo ágil y un mantenimiento sencillo del código, haciéndolo especialmente útil en contextos académicos, científicos o industriales. Su diseño promueve la legibilidad, algo que resulta clave en entornos donde la iteración rápida, la reproducibilidad de experimentos y la integración con otras tecnologías son esenciales.

Desde su creación, Python ha ganado una enorme popularidad, esto gracias a una comunidad activa que, en conjunto, ha creado un gran ecosistema de bibliotecas que cubre una amplia variedad de aplicaciones. Desde cálculos matemáticos básicos hasta tecnología actual como la inteligencia artificial, análisis de datos, visualización, automatización de tareas y control de dispositivos, son solo algunas de ellas.

En este Trabajo de Fin de Grado, Python ha desempeñado un papel fundamental. Ha sido el lenguaje elegido para desarrollar los scripts encargados de automatizar la ejecución de los experimentos, comunicarse con dispositivos externos como el enchufe inteligente, registrar los datos recogidos y analizarlos posteriormente mediante técnicas estadísticas y representaciones gráficas. Su versatilidad ha permitido integrar componentes muy diversos, desde herramientas de bajo nivel como el kernel de Linux hasta servicios web y bibliotecas avanzadas para el tratamiento de datos, todo ello dentro de una arquitectura modular y coherente.

Entre las bibliotecas utilizadas, destacan las siguientes:

- **requests**[13] Esta librería externa permite realizar peticiones **HTTP** de manera sencilla y eficiente, soportando métodos como `GET`, `POST`, `PUT` y `DELETE`. También facilita la gestión de cabeceras, parámetros y respuestas en formato **JSON**. En el proyecto se ha utilizado para establecer la comunicación con el enchufe inteligente, el cual ofrece una `API REST` a través de la que se han consultado periódicamente variables como potencia, voltaje y corriente. Consultas necesarias para analizar el consumo energético durante la navegación web.

- **pandas**[10] Librería de referencia para el análisis de datos en *Python*, especialmente en contextos científicos e industriales. Proporciona estructuras de datos como `Series` y `DataFrame`, que permiten trabajar con datos tabulares de manera eficiente. En este trabajo, se ha utilizado para leer los archivos CSV generados durante los experimentos, filtrar datos por intervalos temporales o dominios web, calcular estadísticas básicas y preparar los datos para su posterior análisis.

- **matplotlib.pyplot**[17] Biblioteca estándar para la creación de gráficas en 2D. Ha sido clave para representar visualmente la evolución del consumo energético a lo largo del tiempo, así como para comparar distintos escenarios de navegación. Se ha aprovechado

su capacidad de personalización para adaptar etiquetas, leyendas, colores y estilos a las necesidades del proyecto.

- **seaborn**[9] Complementa a matplotlib con una interfaz de alto nivel que permite generar gráficos estadísticos sofisticados —como mapas de calor, diagramas de violín o gráficos de dispersión con líneas de regresión— con menos código. En el proyecto, ha facilitado la exploración visual de patrones de consumo energético entre diferentes páginas web, así como la detección de posibles relaciones relevantes entre variables.

- **scipy.stats (pearsonr)**[18] Módulo de la biblioteca SciPy que ofrece herramientas estadísticas avanzadas. En particular, la función pearsonr ha sido empleada para calcular el coeficiente de correlación lineal de Pearson entre las mediciones del enchufe inteligente y las extraídas directamente del sistema a través del kernel. Este análisis ha permitido validar la coherencia y fiabilidad de ambas fuentes de datos.

- **subprocess**[14] Módulo de la biblioteca estándar de Python que permite ejecutar comandos del sistema operativo, además de capturar su salida, errores y códigos de retorno. Se ha utilizado para automatizar completamente la ejecución de los experimentos: lanzar el navegador, abrir URL específicas, controlar pestañas mediante herramientas como xdotool y finalizar procesos al término de cada escenario de navegación. Esta automatización ha sido fundamental para mantener condiciones de prueba homogéneas y reproducibles.

La elección de Python como lenguaje principal ha sido clave para integrar de forma eficiente todos los elementos del sistema. Su capacidad para interactuar tanto con APIs como con el sistema operativo, sumada a la potencia de sus bibliotecas para análisis y visualización de datos, ha permitido construir una solución flexible, automatizada y robusta. Gracias a Python, se ha conseguido reducir el tiempo de desarrollo, facilitar el mantenimiento del código y garantizar la trazabilidad de todo el proceso experimental. En definitiva, ha actuado como un puente entre dispositivos físicos, herramientas software y el análisis de resultados, permitiendo que todo el flujo de trabajo funcione de forma coordinada, reproducible y escalable.

2.1.2. Kernel de Linux

El sistema operativo empleado en este proyecto ha sido una distribución ligera de Linux, en concreto Linuxt Mint Cinnamon, seleccionada por su bajo consumo de recursos, su estabilidad y su capacidad para ofrecer un control detallado sobre el hardware, pero manteniendo un buen nivel de gráficos. Estas características la convierten en una opción especialmente adecuada para tareas de monitorización energética, donde resulta fundamental disponer de un entorno controlado, reproducible y eficiente.

El núcleo de este sistema operativo, el **kernel de Linux**[8], permite acceder fácilmente a una amplia variedad de datos relacionados con el estado del hardware, como el consumo eléctrico del procesador, la temperatura de distintos componentes o los niveles de voltaje. Para la obtención de esta información, se ha hecho uso tanto del sistema de archivos virtual /sys, concretamente del directorio /sys/class/hwmon/ (conocido como **HWMON**[7]), además de valorar diversas herramientas de línea de comandos como cat, grep y powertop.

Durante el desarrollo del proyecto, se han recogido periódicamente datos energéticos directamente desde el sistema operativo. Estos datos incluyen el consumo eléctrico de la CPU y de otros componentes, y han sido recopilados mediante scripts automatizados que accedían a archivos del sistema, analizaban su contenido y registraban los valores relevantes.

El principal objetivo de utilizar el kernel de Linux ha sido disponer de una fuente de medición energética interna, accesible y fiable, que pudiera servir como punto de comparación frente a las mediciones externas obtenidas mediante el enchufe inteligente. De este modo, se han generado dos conjuntos de datos paralelos: por un lado, las mediciones internas obtenidas desde el sistema, y por otro, las externas proporcionadas por el enchufe. Ambos registros han sido sincronizados temporalmente para permitir su comparación directa.

Esta estrategia ha sido clave para evaluar la consistencia entre ambos métodos de medición. La comparación de los datos ha permitido validar la coherencia entre los valores obtenidos internamente y los registrados externamente, contribuyendo así a aumentar la fiabilidad del análisis. Asimismo, ha facilitado la detección de posibles errores o desviaciones en las estimaciones energéticas, reforzando la robustez del enfoque adoptado en este trabajo.

2.1.3. Shelly Plug

El enchufe inteligente utilizado en este proyecto pertenece a la plataforma Shelly, un sistema domótico que permite controlar y monitorizar el consumo eléctrico de los dispositivos conectados a través de una red wifi. En concreto, se ha utilizado el modelo Shelly Plug S Generación 3. Este tipo de dispositivo es capaz de medir en tiempo real parámetros como la potencia instantánea, la energía total acumulada, la tensión o la intensidad, lo que lo convierte en una herramienta muy útil para el análisis del comportamiento energético de un equipo informático.

Una de las funcionalidades más destacables del enchufe es la disponibilidad de una API REST[1] accesible desde la red local. Gracias a esta interfaz, es posible consultar sus mediciones de forma automática desde cualquier aplicación, sin necesidad de depender de software propietario o herramientas adicionales. En el contexto de este proyecto, se ha aprovechado esta característica para integrar el enchufe en el sistema de medición, lo que ha permitido recoger datos energéticos de forma automatizada y sin intervención manual.

Para obtener estas mediciones, se han desarrollado scripts en Python que realizan peticiones HTTP periódicas a la API del dispositivo, utilizando la biblioteca requests. Cada solicitud devuelve información sobre la potencia consumida en ese instante; esta es guardada junto con una marca temporal precisa, lo que permite registrar con exactitud el momento de cada lectura.

Las mediciones se han sincronizado con eventos específicos, como la navegación por distintas páginas web, con el fin de establecer una relación directa entre la actividad realizada y el consumo energético registrado. Todos los datos obtenidos se han almacenado en archivos CSV, lo que ha facilitado su análisis posterior y su comparación con las métricas recogidas desde el propio sistema operativo.

El objetivo principal de emplear este enchufe inteligente ha sido disponer de una fuente de medición energética externa, completamente independiente del propio sistema operativo y del entorno software del ordenador analizado. Al tratarse de un dispositivo físico separado, sus registros no se ven condicionados por las herramientas de monitorización del sistema, actuando así como referencia objetiva.

Esta característica ha resultado especialmente valiosa a la hora de comparar los datos externos del enchufe con los internos obtenidos a través del kernel de Linux, permitiendo verificar la coherencia y consistencia entre ambas fuentes. Esta comparación ha sido clave para validar la fiabilidad del sistema de medición y reforzar la solidez del enfoque experimental adoptado en

el proyecto.

2.1.4. Herramientas auxiliares

Además de las tecnologías principales empleadas para el desarrollo del sistema y la recopilación de datos, a lo largo del proyecto se han utilizado diversas herramientas complementarias, las cuales han resultado fundamentales en distintas fases del trabajo. Aunque no forman parte directa del sistema implementado, han sido clave para organizar el proyecto, desarrollar el código de manera eficiente, gestionar los cambios realizados y redactar adecuadamente la memoria del Trabajo de Fin de Grado. Estas herramientas son:

- **Pycharm**: Para el desarrollo en Python se ha empleado **PyCharm**[4], un entorno de desarrollo integrado (IDE) muy completo que facilita la escritura, prueba y depuración del código. Entre sus funcionalidades destacan el autocompletado, la detección temprana de errores, el depurador paso a paso y una navegación eficiente entre archivos del proyecto. Además, su soporte para entornos virtuales y la gestión de bibliotecas ha contribuido a mantener el entorno de trabajo organizado y aislado durante todo el desarrollo.

- **Git**: La gestión del código fuente se ha realizado mediante Git[15], un sistema de control de versiones distribuido que ha permitido registrar los cambios de forma estructurada y segura. Gracias a él, ha sido posible retroceder a versiones anteriores en caso de errores, experimentar con nuevas funcionalidades sin afectar al código principal y mantener un historial completo del progreso del proyecto. Asimismo, se ha utilizado GitLab[3] como repositorio remoto, en concreto el de la EIF[1], lo que ha facilitado el almacenamiento en la nube y el acceso al proyecto desde distintos dispositivos.

- **LaTeX**: La redacción de este documento se ha llevado a cabo utilizando LaTeX[5], una herramienta ampliamente utilizada en el ámbito académico para la creación de documentos técnicos y científicos. Su capacidad para generar textos bien estructurados, con fórmulas, tablas, gráficos y referencias automáticas, ha permitido mantener una presentación clara y coherente a lo largo de toda la memoria. Además, ha facilitado la automatización de tareas como la creación del índice, la gestión de bibliografía o la numeración de

[1] https://gitlab.eif.urjc.es/

figuras y tablas, mejorando así la calidad y el orden del documento final. Su implementación se ha dado a través de Overleaf[2], una plataforma colaborativa online para la edición de documentos LaTeX.

2.2. Trabajos relacionados

En los últimos años, el interés por la sostenibilidad y la eficiencia energética en el ámbito de la informática ha experimentado un notable crecimiento. Este proyecto se inscribe dentro de una línea de investigación más amplia que analiza el impacto energético del software, especialmente en dispositivos cliente.

Entre los trabajos más relevantes destaca el de Tene et al. (2021) [16], quienes estudian el consumo energético de distintas páginas web en dispositivos móviles, relacionando el peso y la complejidad del diseño web con el uso de batería. Su estudio demuestra que optimizaciones en el front-end pueden contribuir a reducir significativamente el consumo energético en el usuario final.

Asimismo, es fundamental mencionar el trabajo de Fettweis y Zimmermann (2008) [2], que introducen el concepto de "*Green ICT*" o tecnologías de la información y comunicación ecológicas. Proponen un diseño integral de sistemas y servicios digitales que aborde la eficiencia energética desde el software hasta el hardware, promoviendo un enfoque sostenible en toda la cadena tecnológica.

Un aporte complementario es el estudio de Pérez et al. (2019) [11], quienes desarrollan una herramienta para el análisis energético en estaciones de trabajo Linux, empleando sensores internos del sistema operativo junto con dispositivos externos para realizar mediciones paralelas. Su enfoque de doble medición ha servido de inspiración para la arquitectura de este proyecto.

Más recientemente, el trabajo de Zaidman (2024) [21] titulado "An Inconvenient Truth in Software Engineering? The Environmental Impact of Testing Open Source Java Projects" pone en evidencia el impacto ambiental de procesos rutinarios en el desarrollo de software, como la ejecución continua de pruebas automáticas. El autor destaca que estas prácticas pueden generar una huella energética significativa, especialmente a gran escala, lo que refuerza la idea de que el software, aunque intangible, conlleva consecuencias físicas y ambientales. Esta perspectiva

[2]https://es.overleaf.com/

se alinea directamente con los objetivos del presente trabajo, que busca visibilizar el consumo energético derivado de la navegación web, una actividad habitual y masiva.

Finalmente, a nivel práctico, existen iniciativas comunitarias como el *Website Carbon Calculator* [19], una herramienta online que estima el impacto ambiental de una página web en función de su tamaño, recursos cargados y eficiencia del hosting. Aunque ofrece cálculos aproximados, representa una referencia útil para la concienciación sobre la huella energética del ecosistema web.

En conjunto, estos trabajos demuestran que incluso actividades cotidianas como la navegación web pueden implicar una huella energética considerable cuando se consideran a gran escala. Esto subraya la importancia de contar con herramientas automatizadas para medir y analizar este impacto, tal y como se ha desarrollado en el presente proyecto.

Capítulo 3

Desarrollo del proyecto

Este capítulo recoge en detalle las fases técnicas de este proyecto. Describe la arquitectura del sistema, la metodología que se ha seguido, las decisiones de desarrollo más relevantes e implementación de las ideas principales. El objetivo ha sido construir un sistema capaz de medir y comparar el consumo energético asociado a la navegación web, combinando los datos obtenidos internamente del sistema operativo con datos externos recogidos desde un enchufe inteligente.

3.1. Metodología de trabajo

Para el desarrollo de este proyecto se ha adoptado una metodología ágil basada en Scrum, adaptada a las limitaciones de tiempo derivadas de la compaginación con un trabajo a jornada completa y el seguimiento simultáneo de otras asignaturas del grado. El trabajo se organizó en sprints de una o dos semanas, lo que permitió dividir el proyecto en fases manejables, con objetivos específicos y un enfoque iterativo que facilitó tanto el avance progresivo como la revisión y mejora continua de cada componente.

El proyecto se extendió durante seis meses, dedicando principalmente las tardes y los fines de semana. Esta planificación permitió mantener una constancia en el desarrollo, a pesar de las limitaciones horarias.

3.1.1. Sprint 0: Análisis preliminar y definición del propósito

Durante esta primera fase, el objetivo principal fue establecer una base sólida sobre la cual construir el proyecto. Se dedicó tiempo a la lectura detallada de trabajos relacionados, especialmente uno centrado en el análisis energético de sistemas, que sirvió de inspiración y referencia conceptual. Paralelamente, se llevó a cabo una reflexión crítica sobre el enfoque del proyecto, delimitando los objetivos generales y específicos. Esta etapa resultó clave para acotar el problema, definir una metodología de evaluación adecuada y comprender el valor que podría aportar una herramienta de análisis energético centrada en el comportamiento de páginas web.

3.1.2. Sprint 1: Investigación sobre obtención de datos desde el sistema

En este sprint se exploraron distintas formas de obtener información relativa al consumo energético desde el propio sistema operativo Linux. Se investigaron utilidades como powertop, lm-sensors y comandos directos como cat sobre archivos del sistema virtual /sys/class. Además, se realizó la instalación de una distribución ligera basada en Linux Mint Cinnamon sobre un entorno de pruebas, priorizando la eficiencia de recursos y la facilidad de uso. Esta configuración se mantendría durante todo el desarrollo. El objetivo fue identificar posibles fuentes de datos internas que permitieran registrar el comportamiento energético del sistema sin requerir hardware adicional.

3.1.3. Sprint 2: Experimentación con herramientas del sistema

En esta etapa se puso en práctica los conocimientos adquiridos anteriormente. Se realizaron múltiples pruebas con las herramientas exploradas (powertop, sensors, cat /sys/...), evaluando su capacidad de proporcionar datos útiles y estables. A través de sensors, se descubrió el subsistema hwmon, que ofrece una interfaz más estructurada para acceder a métricas térmicas y energéticas de distintos componentes del sistema, como CPU, placa base o GPU. Se analizaron los archivos disponibles en /sys/class/hwmon/ y se validó que ofrecían datos en tiempo real sobre el consumo del procesador, lo que motivó su elección como fuente principal de datos internos del sistema.

3.1.4. Sprint 3: Primera integración con enchufe inteligente (Tuya)

El objetivo de este sprint fue incorporar una fuente de medición energética externa. Se recibió un primer enchufe inteligente basado en la plataforma Tuya, con capacidad para monitorizar el consumo de dispositivos eléctricos conectados. Se estudió su funcionamiento mediante API REST, accediendo a los datos de potencia desde Python mediante requests. Sin embargo, el dispositivo requería conexión con la nube de Tuya, implicando autenticación mediante credenciales y tokens gestionados a través de su plataforma para desarrolladores. Las limitaciones en la frecuencia de peticiones y la dependencia de Internet afectaban la estabilidad y precisión de las mediciones, por lo que, tras varias pruebas, se concluyó que no era adecuado para el tipo de análisis requerido. Se descartó su uso y se inició la búsqueda de una alternativa más fiable y local.

3.1.5. Sprint 4: Primer sistema automatizado de medición y visualización

Una vez seleccionada hwmon como fuente de datos, se desarrolló un primer prototipo en Python capaz de automatizar la apertura de varias páginas web, registrar el consumo energético del procesador en tiempo real, y representar los resultados mediante visualizaciones gráficas. Se utilizaron bibliotecas como matplotlib y pandas para generar gráficas temporales, y se comenzó a diseñar una estructura de almacenamiento coherente en formato CSV, incluyendo etiquetas de tiempo y nombre de la página web cargada en cada momento. Este sistema permitía obtener un perfil energético básico por cada página. También se realizaron las primeras reflexiones sobre métricas de análisis, como media y desviación típica, preparando el terreno para comparaciones futuras.

3.1.6. Sprint 5: Incorporación del enchufe inteligente Shelly

Durante este sprint se introdujo un nuevo dispositivo de medición energética: el Shelly Plug S, un enchufe inteligente con capacidad para monitorizar el consumo eléctrico de dispositivos conectados y que permite el acceso a sus datos a través de una API REST alojada localmente. Esta característica lo hacía especialmente adecuado para el proyecto, al no depender de plataformas en la nube ni requerir autenticación externa. Se procedió a su configuración dentro de la red local y se validó su correcta comunicación con el entorno de desarrollo.

A continuación, se diseñó y desarrolló un script en Python que accedía a los endpoints de la API del Shelly, solicitando los datos de potencia activa a intervalos definidos. Se verificó la precisión temporal del dispositivo y se implementó un sistema de almacenamiento en archivos CSV para registrar los datos obtenidos con su correspondiente marca temporal. Estos datos representaban el consumo energético total del sistema (PC completo) y constituían una fuente externa complementaria a la información extraída directamente del kernel de Linux. Esta etapa finalizó con un conjunto de mediciones satisfactorias que mostraban lecturas estables y coherentes, aptas para el análisis comparativo posterior.

3.1.7. Sprint 6: Integración de fuentes de datos para análisis conjunto

Durante este sprint comenzó una de las fases más importantes del proyecto: la integración de las dos fuentes principales de datos —los obtenidos mediante la herramienta hwmon desde el kernel de Linux y los proporcionados por el enchufe inteligente— con el objetivo de unificarlos para su posterior análisis comparativo.

Se diseñaron y desarrollaron dos scripts independientes pero complementarios. El primero automatizaba la navegación por un conjunto definido de páginas web mientras registraba los datos de consumo energético del procesador a través de hwmon. Estos datos se etiquetaban dinámicamente según la página web activa en cada momento, generando así una trazabilidad precisa de los eventos.

El segundo script, dedicado al enchufe inteligente, capturaba de forma continua el consumo total del equipo mediante consultas periódicas a su API local. A diferencia del primero, esta fuente no incluía etiquetas por página, sino una serie temporal continua de valores de potencia.

Se comenzó a trabajar en una estrategia para sincronizar ambos conjuntos de datos a través de sus marcas de tiempo, de forma que se pudiera relacionar la energía consumida globalmente con los momentos en que el sistema se encontraba navegando por cada página. Esta integración permitió generar un único archivo combinado donde cada punto de datos incluyera tanto la fuente del kernel como la del enchufe, junto con etiquetas temporales o de contexto.

En paralelo, se empezó a explorar cómo realizar comparaciones significativas entre ambas fuentes. Para ello, se investigaron técnicas estadísticas básicas como el cálculo de medias, desviaciones típicas y análisis de la distribución normal. Estas métricas servirían más adelante para evaluar la coherencia entre ambas mediciones y observar patrones de consumo.

3.1.8. Sprint 7-8. Validación cruzada y análisis de resultados

Una vez establecidos los mecanismos de sincronización y recopilación unificada de datos, este sprint se centró en analizar e interpretar los resultados obtenidos. Se aplicaron técnicas estadísticas para comparar ambas fuentes y evaluar su correlación.

Concretamente, se calcularon métricas como la media, la desviación estándar y la varianza de los consumos registrados por cada sistema para diferentes páginas web. Estos indicadores permitieron obtener una visión cuantitativa del comportamiento energético durante la navegación web, y establecer si existía consistencia entre lo reportado por el sistema operativo y el enchufe inteligente.

Además, se utilizó la función pearsonr del módulo scipy.stats para determinar el coeficiente de correlación lineal de Pearson entre las dos series de datos. Esta medida permitió evaluar el grado de relación lineal entre ambas fuentes, proporcionando una validación cruzada de los resultados y reforzando la fiabilidad de los datos recogidos.

Durante este sprint también se generaron las primeras representaciones gráficas que permitían observar visualmente el consumo energético a lo largo del tiempo y la relación entre las distintas fuentes de medición. Estas gráficas facilitaron la detección de posibles inconsistencias y ayudaron a identificar patrones relevantes en el comportamiento energético de las páginas web analizadas.

Esta metodología ha permitido mantener un flujo de trabajo ordenado, adaptativo y centrado en resultados, asegurando una trazabilidad clara del desarrollo y una documentación coherente de cada fase del proyecto.

3.2. Arquitectura de funcionamiento

El sistema desarrollado se compone de varios módulos interconectados cuya finalidad es automatizar la navegación web, registrar el consumo energético asociado a cada sesión y analizar los datos obtenidos. La arquitectura general se puede dividir en cuatro componentes principales:

- **Automatización de navegación web:** Este módulo se encarga de acceder a las páginas web seleccionadas de forma automática mediante scripts en Python. Se controla el tiempo de navegación y se garantiza que cada sesión sea reproducible. Se utilizó el navegador

Figura 3.1: Arquitectura del sistema

Firefox.

- **Captura de datos del sistema (hwmon):** En paralelo a la navegación, se registran métricas internas del sistema como el uso de CPU, utilizando la interfaz hwmon del kernel de Linux. Esta información se almacena como serie temporal para su posterior sincronización con los datos energéticos.

- **Medición de consumo eléctrico (Shelly Plug S):** El enchufe inteligente Shelly Plug S registra en tiempo real la potencia eléctrica consumida por el ordenador durante cada sesión. La comunicación con el dispositivo se realiza a través de peticiones HTTP periódicas que consultan su API local.

- **Sincronización y análisis de datos:** Una vez recogidos los datos de hwmon y del Shelly, estos se sincronizan temporalmente y se procesan mediante scripts en Python utilizando bibliotecas como pandas, matplotlib, scipy.stats y seaborn. Los datos se corrigen, se integran y se representan gráficamente para su análisis y comparación.

El sistema completo ha sido diseñado para ser reproducible, extensible y modular, permitiendo ajustes en la frecuencia de muestreo, la lista de páginas web o el método de análisis sin necesidad de reestructurar el código base. Además, la arquitectura desarrollada facilita la

automatización de todo el flujo de trabajo experimental, desde la recogida de datos hasta su visualización.

3.3. Implementación

La implementación del sistema se ha dividido en varios módulos funcionales que operan de manera coordinada para obtener, procesar y relacionar los datos energéticos con las páginas web visitadas. Cada componente se ha desarrollado en Python, aprovechando su capacidad de interacción tanto con el sistema operativo como con dispositivos externos. A continuación, se describen los subcomponentes más relevantes.

3.3.1. Automatización de la navegación web

Para analizar el impacto energético de distintas páginas web, se diseñó un script que automatiza la apertura y navegación secuencial por un conjunto de URL. Estas URL se suministran al sistema mediante un archivo JSON, lo que permite gestionar fácilmente la lista de sitios a analizar. Esta automatización se llevó a cabo mediante el navegador Firefox, lanzado desde Python utilizando el módulo `subprocess`, garantizando un control preciso sobre el proceso de carga y el tiempo de visita en cada página.

El proceso completo se estructura en cinco fases:

1. **Lanzamiento del navegador:** se abre una nueva ventana de Firefox con una página en blanco, asegurando un entorno limpio de partida.

2. **Carga de la URL:** tras una breve pausa, se abre una nueva pestaña con la página web correspondiente a la URL seleccionada.

3. **Temporización y captura:** la página permanece abierta durante un periodo fijo (por ejemplo, 60 segundos), permitiendo estabilizar el consumo y registrar suficientes muestras de potencia y uso de CPU.

4. **Cierre y transición:** al finalizar el tiempo asignado, se cierra la pestaña activa mediante herramientas como `xdotool`, y se repite el proceso con la siguiente URL.

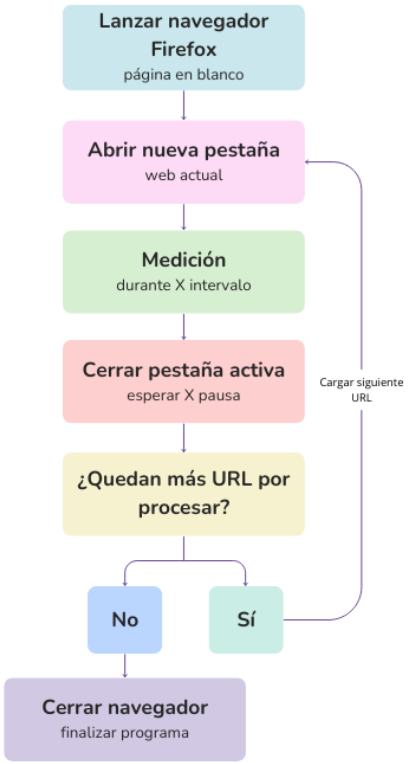


Figura 3.2: Proceso de navegación automática

5. **Finalización:** una vez procesadas todas las URL, se cierra completamente el navegador para liberar recursos.

Durante todo el proceso, el script (Ver Anexo A) registra la URL activa junto a las marcas temporales, lo que permite asociar de forma precisa las muestras energéticas recogidas con la página correspondiente. Esta automatización asegura la reproducibilidad del experimento, elimina la intervención manual y garantiza condiciones homogéneas entre sesiones de navegación.

3.3.2. Extracción de datos del kernel de Linux

El segundo módulo se encarga de capturar datos de consumo energético directamente desde el sistema operativo. Para la obtención de las métricas internas del sistema se empleó la interfaz `hwmon` del kernel de Linux, disponible en `/sys/class/hwmon`. Esta interfaz permite acceder a sensores de voltaje, corriente, potencia, temperatura, entre otros.

En el equipo utilizado, el sensor encargado de medir la potencia de la CPU se corresponde con el módulo `fam15h_power`, específico de procesadores AMD basados en la arquitectura Family 15h (Bulldozer, Piledriver y derivados)[6]. Este módulo permite acceder a una estimación de la potencia total consumida por el procesador en tiempo real.

El valor de potencia se encuentra ubicado generalmente en un archivo de nombre similar a:

```
/sys/class/hwmon/hwmonX/power1_input
```

Dado que la ubicación de los sensores puede variar entre equipos u orden de carga de los módulos, fue necesario verificar manualmente la correspondencia entre los archivos `hwmon` y los sensores de la CPU. Esto se puede hacer utilizando herramientas como `lm-sensors` o la inspección directa de los nombres de los archivos disponibles. Para identificar correctamente el módulo activo, se utilizó el comando:

```
cat /sys/class/hwmon/hwmonX/name
```

Esta información también puede verificarse mediante el comando `sensors` del paquete `lm-sensors`. En el equipo utilizado, la salida de dicho comando muestra los distintos módulos de sensores disponibles, tal y como se observa en la Figura 3.3. En ella se puede comprobar que el módulo `fam15h_power` es el encargado de proporcionar la estimación de potencia consumida por la CPU.

Una vez localizado el archivo correspondiente, se accedió a su contenido desde Python. El valor leído se encuentra en microwatios (`µW`), por lo que fue necesario dividirlo entre 1 000 000 para convertirlo a vatios (W) y permitir así su comparación directa con los datos de potencia obtenidos del enchufe Shelly. En el Código 3.1 se muestra un fragmento de código que realiza esta operación.

El script completo se encuentra en el Anexo A.1.

3.3.3. Obtención de datos con el enchufe inteligente Shelly Plug S

El tercer componente del sistema corresponde a un enchufe inteligente *Shelly Plug S*, utilizado para medir la potencia activa total consumida por el equipo a través de la red eléctrica. Este dispositivo fue seleccionado por su capacidad para funcionar en red local y por ofrecer

```python
import glob
import time

def get_power_field():
    hwmon_paths = glob.glob("/sys/class/hwmon/hwmon*/name")
    for path in hwmon_paths:
        with open(path, "r") as f:
            if "fam15h_power" in f.read().strip():
                return path.replace("name", "power1_input")
    return None
def save_power(power_field):
    try:
        with open(power_field, 'r') as file:
            power = int(file.read().strip()) / 1_000_000
        return {
            "timestamp": time.strftime("%Y%m%dT%H:%M:%S"),
            "power": power
        }
    except KeyboardInterrupt:
        print('\nStopping capture...')

# Ejemplo de uso
while True:
    power_field = get_power_field()
    power = save_power(power_field)
    if power is not None:
        print(f"Potencia CPU: {power:.3f} W")
    time.sleep(1) # Intervalo de muestreo
```

Código 3.1: Acceso y lectura del archivo de potencia de la CPU

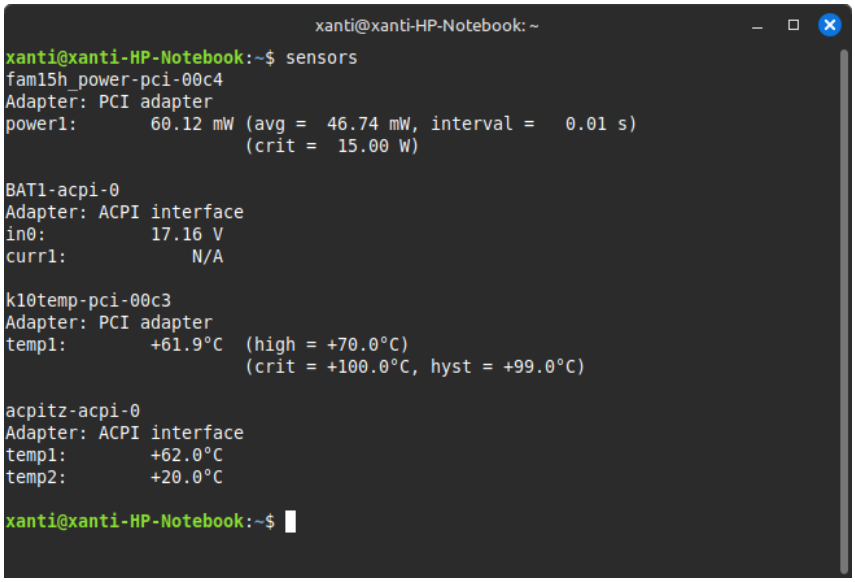


Figura 3.3: Resultado con el comando sensors en el PC de prueba

una API REST sencilla, lo que evita dependencias con servicios en la nube y garantiza mayor control sobre las mediciones.

El Shelly fue previamente configurado en la red wifi local, asignándole una dirección IP estática. El acceso a los datos se realizó mediante peticiones HTTP periódicas directamente a su API REST. La información de consumo puede obtenerse mediante una simple petición GET a la siguiente URL:

`http://<IP\_del\_enchufe>/rpc/Switch.GetStatus?id=0`

La respuesta de la API se encuentra en formato JSON e incluye, entre otros, el campo `apower`, que indica la potencia activa instantánea consumida por el dispositivo expresada en vatios (W). Un ejemplo de esta respuesta puede verse en el Código 3.2.

El Código 3.3 muestra un ejemplo del script de Python utilizado para obtener la lectura de potencia activa del dispositivo Shelly mediante una petición HTTP a su API local. El script implementado realiza consultas periódicas (por ejemplo, cada segundo) a la dirección IP del dispositivo, accediendo al endpoint que devuelve el valor de potencia instantánea. Cada medición se almacena junto con una marca temporal (`timestamp`) en un archivo CSV, con el objetivo de facilitar su posterior sincronización con las métricas registradas por el sistema.

A diferencia del módulo anterior, este método proporciona una visión global del consumo energético del equipo, incluyendo no solo la CPU, sino también periféricos, ventiladores y

```
{
    "id": 0,
    "source": "init",
    "output": false,
    "apower": 0.0,
    "voltage": 231.4,
    "freq": 50.0,
    "current": 0.000,
    "aenergy": {
        "total": 1718.000,
        "by_minute": [0.000, 0.000, 0.000],
        "minute_ts": 1752176820
    },
    "ret_aenergy": {
        "total": 0.000,
        "by_minute": [0.000, 0.000, 0.000],
        "minute_ts": 1752176820
    },
    "temperature": {
        "tC": 41.5,
        "tF": 106.6
    }
}
```

Código 3.2: Ejemplo de respuesta al obtener el estado del enchufe desde la API

```python
import requests
import time

def get_power(ip):
    url = f"http://{IP_SHELLY}/rpc/Switch.GetStatus?id=0"
    try:
        response = requests.get(url, timeout=5)
        if response.status_code == 200:
            return {
                "timestamp": time.strftime("%Y%m%dT%H:%M:%S"),
                "power": data.get("apower", 0.0)
            }
    except Exception as e:
        print(f"Error while getting status: {response.status_code}")
        return None

# Ejemplo de uso
ip_shelly = "192.168.1.50"
while True:
    response = get_power(ip_shelly)
    if response is not None:
        print(f"Potencia: {response[power]} W")
    time.sleep(1)  # Intervalo de muestreo
```

Código 3.3: Lectura de potencia del Shelly Plug S mediante petición HTTP

otros componentes que pueden no ser directamente accesibles desde el kernel. Esta perspectiva resulta esencial para evaluar el consumo total del sistema de forma realista y no limitar el análisis únicamente al uso del procesador.

El script completo se encuentra en el Anexo A.2.

3.3.4. Sincronización y almacenamiento de datos

Una vez obtenidos los datos desde ambos orígenes —interno (`hwmon`) y externo (enchufe inteligente Shelly)—, se procede a su sincronización utilizando el tiempo como eje común. Ambos scripts registran marcas temporales en formato *UNIX timestamp*, lo que permite una alineación precisa de las muestras.

Las dos fuentes se configuraron para registrar datos en intervalos regulares de 1 segundo. Cada muestra se almacena junto con una marca temporal basada en el reloj del sistema para garantizar la consistencia entre series. Para ello se utilizó el módulo `time` de Python,

La duración de cada sesión se definía previamente (por ejemplo, 60, 120 o 240 segundos), y el proceso de captura se ejecutaba de forma continua durante ese intervalo. Una vez completada la sesión, los datos eran cargados y procesados utilizando la biblioteca `pandas` para su posterior análisis y representación gráfica.

Para facilitar el tratamiento conjunto de la información, se desarrolló una herramienta que combina ambos conjuntos de datos y genera un único archivo estructurado con las siguientes columnas:

- **Timestamp:** marca temporal.
- **Sesión/URL activa:** identificador de la página web cargada en ese instante.
- **Potencia Hwmon):** valor estimado del consumo de CPU (en W).
- **Potencia Shelly:** potencia activa total medida por el enchufe inteligente (en W).

Este archivo unificado sirvió como base para los análisis estadísticos posteriores, tales como la correlación entre ambos métodos de medición, la comparación del consumo energético por página web y la visualización temporal del comportamiento energético durante cada sesión.

El sistema fue diseñado con el objetivo de ser fácilmente replicable. Cualquier nuevo conjunto de páginas puede ser evaluado simplemente modificando la lista de URL de entrada. Esto

gracias a que los scripts gestionan automáticamente la navegación, la toma de muestras y la generación del archivo de salida listo para análisis.

3.4. Procesamiento y representación gráfica

Tras la integración de las mediciones procedentes tanto del sistema operativo como del enchufe inteligente, el siguiente paso consistió en procesar y analizar los datos con el objetivo de extraer conclusiones relevantes sobre el consumo energético durante la navegación web.

Para ello, se empleó Python como entorno principal de análisis, apoyándose en bibliotecas especializadas como Pandas, NumPy, Matplotlib y Seaborn. Estas herramientas facilitaron la transformación de los datos brutos en información visual accesible y permitieron realizar comparaciones entre las distintas fuentes y situaciones de navegación.

El flujo de procesamiento seguido fue el siguiente:

- **Carga y limpieza de datos:** Los datos combinados se importaron en estructuras DataFrame, donde se eliminaron registros incompletos y se verificó la correspondencia temporal entre los distintos orígenes de datos.
- **Agrupación por URL:** Utilizando las etiquetas generadas durante la navegación automatizada, se agruparon las muestras según la página web activa en cada instante. Esto permitió calcular métricas específicas por sitio, como la potencia media o la mediana.
- **Cálculo de correlaciones:** Para evaluar la fiabilidad de los datos obtenidos desde el sistema operativo, se empleó la función `pearsonr` de la biblioteca `scipy.stats`, con el fin de medir la correlación lineal entre ambas fuentes de datos.
- **Generación de gráficos:** Los resultados se representaron mediante distintas visualizaciones, entre las que destacan:
  - *Series temporales:* para comparar el consumo medido en diferentes intervalos de tiempo entre páginas web.
  - *Diagramas de barras:* que muestran el consumo medio por página web, facilitando la identificación de diferencias entre ellas.
  - *Gráficos de dispersión:* que ilustran la relación entre ambos sistemas de medición.

Estas visualizaciones resultaron fundamentales para interpretar los resultados, detectar patrones y anomalías, y sirvieron como soporte visual en la memoria del TFG, facilitando una exposición clara y comprensible de las conclusiones obtenidas.

3.5. Reproducibilidad del sistema

Con el objetivo de garantizar la trazabilidad y reproducibilidad del experimento, se adoptaron buenas prácticas tanto en la organización del código como en el almacenamiento de los datos. El sistema ha sido diseñado de forma modular y documentada, permitiendo que cualquier usuario con un entorno similar pueda replicar los resultados obtenidos.

A continuación, se detallan los requisitos y pasos necesarios para ejecutar el sistema:

- **Sistema operativo:** Linux (preferiblemente una distribución ligera, como Ubuntu o Debian).
- **Lenguaje de programación:** Python 3.10 o superior.
- **Bibliotecas utilizadas:**
  - `requests` para la comunicación con Shelly.
  - `pandas` para la manipulación de datos.
  - `matplotlib` y `seaborn` para la visualización.
  - `scipy.stats` para el análisis estadístico.
- **Hardware necesario:**
  - Ordenador conectado a una fuente de alimentación a través de un enchufe inteligente Shelly Plug S.
  - Acceso a la interfaz `/sys/class/hwmon` para lectura de sensores.
- **Configuración del entorno:**
  - Asegurar conectividad local con el Shelly Plug (vía IP estática o detección automática).

- Identificar correctamente el sensor de potencia de CPU en el directorio `hwmon`.

- **Ejecución:** El script principal permite definir la duración de la sesión, lanzar la navegación web automatizada y capturar datos de forma sincronizada. Los resultados se almacenan en archivos CSV para su análisis posterior.

Todo el código fuente ha sido organizado por módulos, acompañado de comentarios explicativos y documentación interna. Esto permite no solo reproducir los experimentos realizados, sino también adaptar el sistema a nuevas plataformas, navegadores o páginas web de interés.

Tabla 3.1: Entorno de ejecución y versiones de software utilizadas

Componente	**Versión / Distribución**
Sistema operativo	Linux Mint 22.1
Kernel de Linux	6.8.0-63-generic
Python	3.12
Navegador web	Firefox 124.0.2
Shelly Plug S	Firmware 1.12.1 (mDNS activado)
Requests (Python)	2.31.0
Pandas	2.0.3
Matplotlib	3.7.2
Seaborn	0.12.2
SciPy	1.11.1

Capítulo 4

Experimentos y validación

Este capítulo describe los experimentos diseñados para validar la fiabilidad y coherencia de los datos obtenidos durante el desarrollo del proyecto. Se presentan los casos de prueba implementados para comprobar el funcionamiento del sistema de medición a partir de distintas fuentes: por un lado, a nivel interno mediante el kernel de Linux (hwmon), y por otro, mediante un enchufe inteligente Shelly conectado a la red eléctrica.

El objetivo principal de esta fase fue verificar que ambos mecanismos de adquisición de datos proporcionan resultados consistentes, suficientemente estables y representativos para ser utilizados en el análisis del impacto energético de la navegación web. A partir de los datos recopilados, se llevaron a cabo análisis estadísticos orientados a evaluar la estabilidad de las mediciones y la correlación entre las fuentes.

4.1. Configuración experimental

Con el fin de comprobar la reproducibilidad y fiabilidad de las mediciones, se diseñó un experimento controlado basado en la repetición sistemática del proceso de monitorización. Para ello, se estableció un entorno experimental compuesto por dos equipos portátiles, un dispositivo de medición externo y diversas herramientas de software. A continuación, se detalla la configuración física y lógica empleada.

- **Hardware:**
  - **Equipo 1 (PC bajo prueba):** Portátil HP Notebook 15-ba039ns, equipado con un

procesador APU AMD Quad-Core A10-9600P, 8,GB de RAM y gráficos dedicados AMD Radeon™ R7 M440 (4,GB DDR3). Este equipo fue el principal objeto de medición energética.

- **Equipo 2 (PC complementario):** Portátil HP Omen 17-an0xx con procesador Intel Core i7, 8,GB de RAM y tarjeta gráfica dedicada NVIDIA GeForce GTX 1050. Este equipo se utilizó como soporte para el desarrollo de scripts y la recopilación de datos del enchufe.
- **Shelly Plug S Gen3:** Dispositivo de medición externo conectado a la red eléctrica.

- **Software:**
  - **Equipo 1:** Linux Mint Cinnamon.
  - **Equipo 2:** Windows 10 Home.
  - **Shelly:** API oficial del fabricante.
- **Entorno de pruebas:** Se retiró la batería del Equipo 1 para evitar interferencias en las mediciones debidas a la carga. Este equipo se alimentó exclusivamente mediante corriente alterna a través del dispositivo Shelly. El Equipo 2 no requirió configuración especial.

4.2. Metodología de medición

Se desarrolló un script que ejecutaba de forma automática y controlada el script de hwmon un número determinado de veces; mientras tanto, en paralelo, se ejecutaba el script para la comunicación con el dispositivo Shelly.

4.2.1. Recopilación de datos

Ambas herramientas de medición —el script de hwmon y el de Shelly— se ejecutaron simultáneamente para cada prueba. El script de hwmon se ejecutó directamente en el Equipo 1 (PC bajo prueba), mientras que el script de comunicación con Shelly se ejecutó en el Equipo 2 para evitar sobrecargar los datos del Equipo 1 con otros procesos.

- **Frecuencia de muestreo:** 1 muestra por segundo (1 Hz).

- **Duración de cada sesión de medición:** 240 segundos (4 minutos).
- **Duración total de cada ejecución:** aproximadamente 12 minutos, que comprenden:
  1. 4 minutos de medición activa en estado de fondo.
  2. 2 minutos de pausa tras el lanzamiento del navegador para estabilizar el sistema antes de la medición.
  3. 4 minutos de medición activa en estado de sesión.
  4. Tiempo adicional para cierre controlado de procesos y pausa antes de iniciar la siguiente medición.
- **Número de ejecuciones:** Para asegurar la validez estadística y minimizar la influencia de valores atípicos, cada escenario de prueba se ejecutó 50 veces. Esta repetición permitió obtener una muestra robusta para análisis cuantitativos, como cálculo de medias, desviaciones estándar y distribución de los valores registrados.

4.2.2. Escenarios de prueba

Durante cada ejecución, se definieron dos escenarios principales para evaluar el comportamiento energético bajo distintas condiciones:

- **Potencia de Fondo (Background Power):** Representa el consumo energético base del sistema sin ejecutar tareas computacionales relevantes. Al iniciar la medición, únicamente estaba activo el script correspondiente; no se ejecutaba otro software controlado. Este escenario establece la línea base para comparar futuros consumos en estado activo.
- **Potencia de Sesión (Session Power):** Corresponde al consumo energético durante la visualización de una página web. Tras la pausa inicial, se abrió el navegador con una página en blanco y, tras un periodo de estabilización, se cargó una página concreta: YouTube. El consumo durante esta interacción se registró para observar el impacto energético de una sesión real de navegación.

Los datos de ambas fuentes se almacenaron en archivos CSV para su posterior sincronización, análisis y comparación. La sincronización temporal entre hwmon y Shelly se gestionó mediante marcas de tiempo y control estricto de las fases de medición (fondo o sesión).

4.2.3. Análisis de resultados

El análisis de las gráficas generadas a partir de las ejecuciones repetidas permite evaluar la repetibilidad, estabilidad y diferencias en el consumo energético medido por dos fuentes: el enchufe inteligente Shelly, que mide el consumo total del sistema, y hwmon, que reporta valores desde el kernel de Linux para componentes específicos.

La Figura 4.1 muestra el consumo energético de la CPU en estado inactivo, según hwmon. Se observa un consumo muy bajo, con media y mediana en torno a 0.02–0.03 W. La variabilidad entre ejecuciones es prácticamente nula, evidenciando alta repetibilidad. Las barras de error, correspondientes a la desviación estándar, son mínimas o imperceptibles, confirmando la estabilidad del sistema en reposo. Estos resultados reflejan un comportamiento energético constante y predecible en condiciones de inactividad.

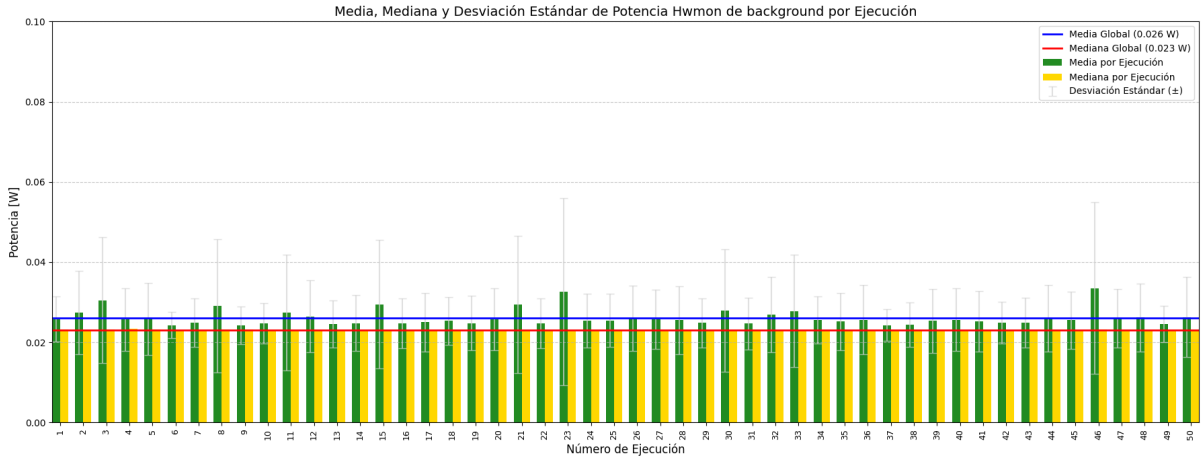


Figura 4.1: Estadísticas de Hwmon para el fondo.

La Figura 4.2 muestra el consumo registrado por hwmon durante la fase activa. Los valores son prácticamente idénticos a los del estado de reposo (0.02–0.03 W), con media y mediana muy próximas, lo que indica una distribución simétrica y estable. La desviación estándar es mínima, demostrando consistencia dentro de cada sesión. Estos resultados sugieren que el sensor no detecta diferencias significativas entre estado activo e inactivo, posiblemente porque las tareas realizadas no generan carga relevante sobre los componentes monitorizados (la CPU) o porque la actividad se concentra en otros elementos, como red o almacenamiento.

La Figura 4.3 presenta las estadísticas de potencia del sistema completo en reposo, medidas mediante Shelly. Se observan medias y medianas entre 9 y 12 W, reflejando el consumo base sin

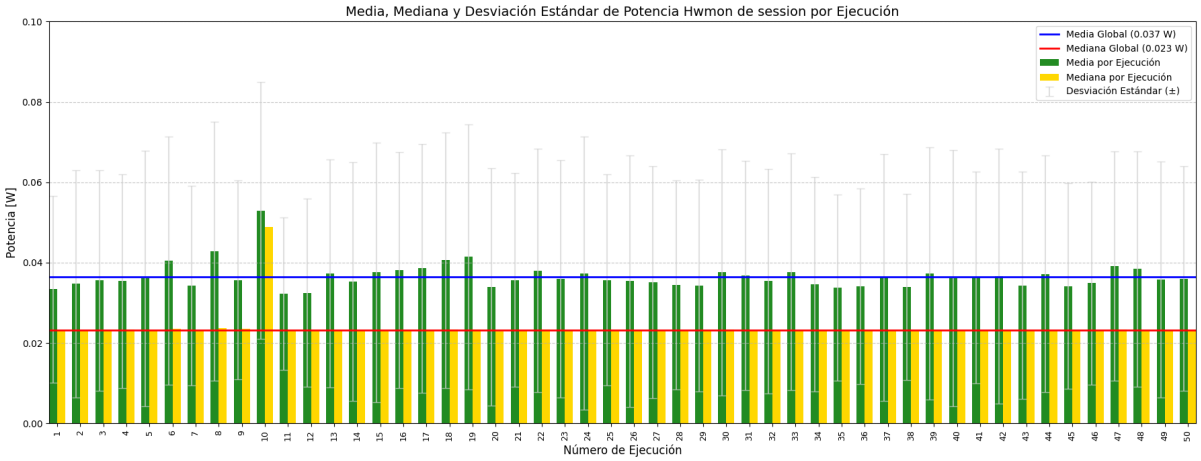


Figura 4.2: Estadísticas de Hwmon para la sesión.

carga activa. Las ejecuciones muestran alta repetibilidad y uniformidad. La proximidad entre media y mediana indica una distribución simétrica sin valores atípicos relevantes. La desviación estándar baja confirma estabilidad energética en reposo. Estos resultados validan a Shelly como herramienta adecuada para establecer la línea base del consumo total, capturando también componentes distintas a la CPU.

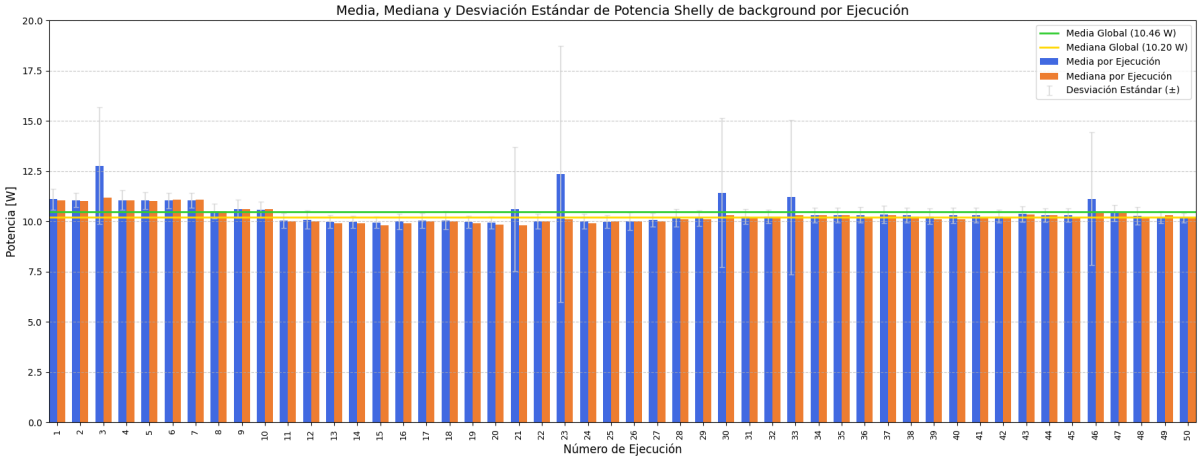


Figura 4.3: Estadísticas de Shelly para el fondo.

La Figura 4.4 muestra el consumo total durante la sesión activa. Se observa un ligero incremento respecto al fondo, con medias y medianas en torno a 11–13 W, lo que indica que la navegación en YouTube incrementa el consumo global. La distribución sigue siendo simétrica y estable, con baja desviación estándar. La diferencia detectada por Shelly entre fondo y sesión confirma que el sistema capta la variación energética inducida por la actividad.

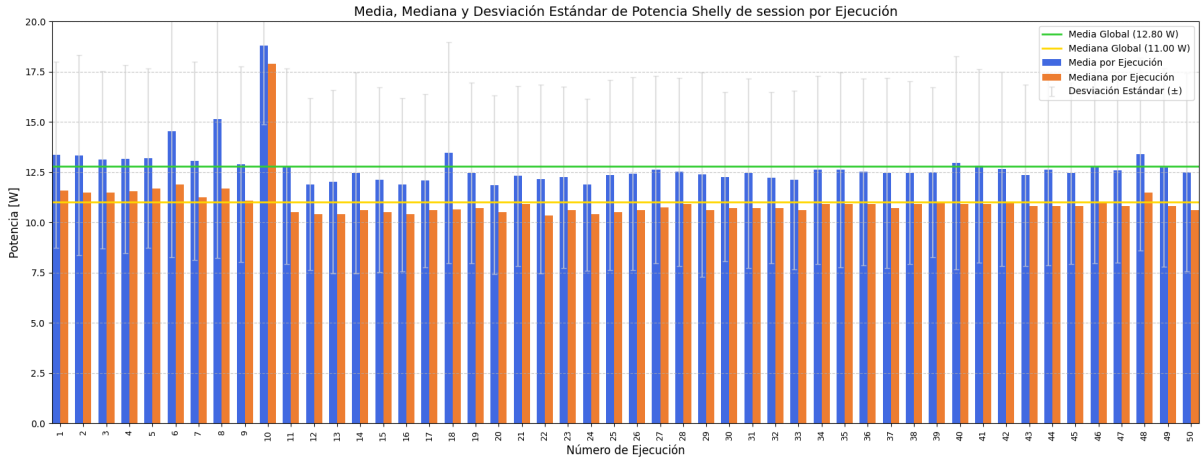


Figura 4.4: Estadísticas de Shelly para la sesióno.

4.2.4. Correlación entre mediciones

El análisis de correlación aporta una nueva dimensión a la interpretación de los datos. La Figura 4.5 evidencia una **correlación de Pearson de 0.5**, lo que indica una relación **positiva moderada** entre el consumo interno de la CPU y el consumo total del sistema. En otras palabras, cuando el consumo detectado por hwmon aumenta, también lo hace el consumo total registrado por Shelly, aunque no de forma estrictamente lineal ni con alta precisión.

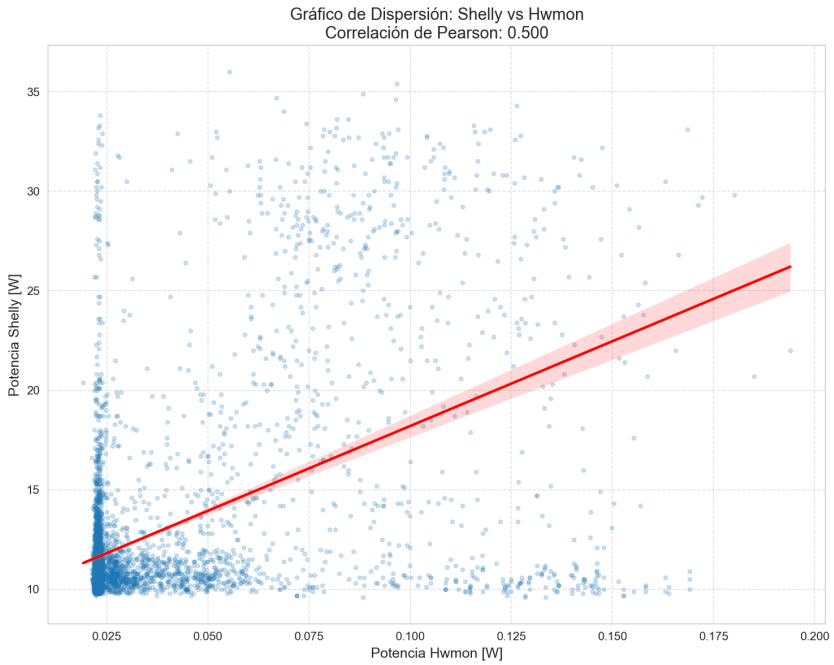


Figura 4.5: Correlación global entre Shelly y Hwmon

Esta correlación moderada implica que:

- Hwmon **no capta el consumo total** del sistema, sino solo el de los componentes monitorizados.
- La dispersión en la nube de puntos sugiere la **influencia de muchos otros factores** sobre el consumo global, como discos duros, red, ventiladores, eficiencia de la fuente de alimentación y periféricos.
- La **línea de regresión ascendente** indica que, aunque no es el único factor, el consumo registrado por hwmon sí tiene un impacto en el total.

Los coeficientes de Pearson por ejecución presentados en la Tabla 4.1 refuerzan esta conclusión. La mayoría de las sesiones muestran correlaciones comprendidas entre 0.40 y 0.60, lo cual confirma que la relación, aunque presente, **varía entre ejecuciones** y depende de la actividad específica de otros componentes no reflejados por este sensor de hwmon.

4.2.5. Conclusiones generales sobre el análisis estadístico

Los resultados obtenidos a partir de este experimento permiten extraer conclusiones sólidas sobre la estabilidad, la correlación y la capacidad de análisis energético de ambos sistemas de medición.

Estabilidad y repetibilidad

La consistencia en las mediciones fue un aspecto destacado. El enchufe inteligente Shelly mostró una excelente repetibilidad experimental, ofreciendo valores altamente consistentes entre ejecuciones. La media y la mediana de la potencia en cada fase (reposo y sesión) se mantuvieron estables, con desviaciones estándar bajas o incluso despreciables. Esta baja dispersión valida la fiabilidad del sistema de medición de Shelly bajo condiciones controladas.

Por otro lado, en el sensor interno, aunque también mostró alta consistencia en sus lecturas entre ejecuciones, es decir, una buena repetibilidad, estas lecturas presentaron valores anómalos o subestimados en comparación con el consumo real esperado. Este punto se abordará en detalle más adelante.

Tabla 4.1: Coeficientes de Correlación de Pearson entre Shelly y Hwmon por Sesión

Ejecución	**Correlación**	**Ejecución**	**Correlación**	**Ejecución**	**Correlación**
01	0.522	18	0.600	35	0.486
02	0.634	19	0.467	36	0.299
03	0.488	20	0.524	37	0.409
04	0.568	21	0.394	38	0.420
05	0.615	22	0.490	39	0.604
06	0.440	23	0.529	40	0.477
07	0.495	24	0.532	41	0.411
08	0.644	25	0.656	42	0.571
09	0.615	26	0.505	43	0.303
10	0.284	27	0.354	44	0.568
11	0.459	28	0.433	45	0.466
12	0.482	29	0.360	46	0.487
13	0.530	30	0.515	47	0.388
14	0.392	31	0.488	48	0.575
15	0.543	32	0.542	49	0.669
16	0.503	33	0.480	50	0.540
17	0.511	34	0.404	Overall	0.500

Las estadísticas globales (ver Tablas 4.2 y 4.3) refuerzan estas observaciones, destacando diferencias significativas en las magnitudes registradas por cada monitorizador:

Tabla 4.2: Estadísticas globales de la potencia de Fondo

Sensor	**Media [W]**	**Mediana [W]**	**Desviación Estándar [W]**
Shelly	10.46	10.20	1.50
Hwmon	0.026	0.023	0.010

- En estado de **reposo**, Shelly registró una media de 10.46 W y desviación estándar de 1.50 W, mientras que hwmon mostró un consumo medio mucho menor (0.026 W), con una desviación estándar de solo 0.010 W.

Tabla 4.3: Estadísticas globales de la potencia de la sesión

Sensor	**Media [W]**	**Mediana [W]**	**Desviación Estándar [W]**
Shelly	12.79	11.00	4.87
Hwmon	0.036	0.023	0.028

- Durante la **sesión activa**, Shelly aumentó su media a 12.79 W con mayor variabilidad (4.87 W de desviación estándar), mientras que hwmon subió ligeramente a 0.036 W, con una desviación estándar también mayor (0.028 W).

Estas diferencias no solo confirman la variación energética entre las fases, sino que también evidencian la naturaleza complementaria de ambos sensores: **Shelly mide el consumo total del sistema**, mientras que **hwmon monitoriza únicamente componentes internos**, específicamente la CPU. De ahí que el incremento de consumo entre fases sea mucho más pronunciado en los datos de Shelly y que las magnitudes registradas por el sensor sean menores.

Correlación entre medidores

El análisis de correlación entre el consumo registrado por el enchufe inteligente Shelly y los datos obtenidos mediante hwmon muestra una relación positiva moderada, con un coeficiente de Pearson alrededor de 0.5. Esto indica que el consumo interno de la CPU tiene un impacto significativo en el consumo total del sistema, pero no explica por completo su variabilidad.

La correlación moderada evidencia que hwmon monitoriza solo una parte del consumo total, limitada a ciertos componentes específicos, mientras que otros factores como discos duros, red, ventiladores, fuente de alimentación y periféricos también influyen en el consumo global, generando dispersión en los datos.

La variabilidad de las correlaciones entre sesiones, entre 0.40 y 0.60, (Ver Tabla 4.4) indica que esta relación depende del contexto y la actividad particular de los componentes no monitorizados por hwmon en cada ejecución. En conjunto, estos resultados sugieren que aunque hwmon es útil para estimar tendencias del consumo, es necesario considerar otras fuentes de consumo para obtener una visión completa del gasto energético del sistema.

Tabla 4.4: Distribución de los coeficientes de correlación por intervalo

Intervalo de correlación	Número de ejecuciones
Mayor a 0.60	6
Entre 0.50 y 0.60	17
Entre 0.40 y 0.50	17
Menor a 0.40	10

Implicaciones para la medición energética

A partir de estos hallazgos, se pueden extraer las siguientes implicaciones clave:

- **La combinación de sensores es esencial**: ningún sensor por sí solo puede proporcionar una imagen completa del consumo. La sinergia entre una medición total (Shelly) y otra a nivel de componente (hwmon) permite un análisis más preciso y segmentado.
- **Hwmon es útil como indicador local**, pero no es representativo del sistema completo. Su utilidad radica en evaluar la carga sobre componentes específicos.
- **Modelos de predicción energética basados únicamente en hwmon serían limitados**. Dada la correlación moderada y la alta dispersión, se requerirían modelos más complejos o múltiples variables para estimar con precisión el consumo total del sistema.

4.2.6. Limitaciones del sensor interno

Es crucial señalar una limitación importante relacionada con el sensor fam15h_power dentro del PC de pruebas. Se observó que este sensor reportaba valores anómalamente bajos, como 57 mW en reposo y aproximadamente 170 mW bajo carga completa de CPU. Estos valores no concuerdan con el consumo real estimado para procesadores AMD de la serie Family 15h/16h, cuyo TDP (Thermal Design Power) suele rondar los 15 W. Esta discrepancia sugiere que el sensor no proporciona una estimación del consumo total del paquete de la CPU, sino que probablemente mide la potencia de un dominio o subsistema interno específico del procesador. Esta interpretación es coherente con las limitaciones en la telemetría de potencia de las arquitecturas AMD de dicha generación, que no ofrecen la misma granularidad o precisión que las arquitecturas más modernas.

A pesar de esta limitación en la precisión absoluta de sus mediciones, se verificó que los valores reportados por el sensor fam15h_power respondían de manera predecible a las variaciones de carga, reflejando incrementos significativos cuando la CPU era sometida a actividad intensiva (aunque fuese en el orden de los mW). Por esta razón, se decidió mantener estas mediciones como un complemento valioso a los datos obtenidos mediante el enchufe inteligente. El objetivo fue validar tendencias y reforzar la interpretación global del comportamiento energético del sistema, aportando una perspectiva más detallada sobre el consumo de la CPU a pesar de la subestimación de su magnitud total.

4.2.7. Conclusión final

En resumen, el análisis estadístico demuestra que la metodología de medición propuesta es robusta y válida para estudiar el comportamiento energético tanto global (mediante el consumo total del sistema medido por Shelly) como específico de la CPU (a través del sensor hwmon). Aunque **Shelly demostró una excelente repetibilidad**, el sensor interno hwmon, si bien consistente en sus lecturas, reportó valores subestimados para el consumo de la CPU. No obstante, su capacidad para **reflejar las tendencias de consumo en respuesta a la carga** lo convierte en un complemento útil. La correlación moderada observada entre ambos monitorizadores subraya la importancia de considerar múltiples fuentes de datos para entender de forma más holística el impacto energético de los procesos digitales, compensando las limitaciones individuales de cada herramienta y ofreciendo una perspectiva más completa.

Capítulo 5

Resultados

En este capítulo se presentan y analizan los resultados obtenidos tras la ejecución del sistema automatizado diseñado para medir el consumo energético asociado a la navegación web. Se describen las métricas extraídas de las dos fuentes de datos principales —el sistema operativo mediante hwmon y el enchufe inteligente— y se contrastan para validar su fiabilidad y coherencia.

Asimismo, se incluyen representaciones gráficas que facilitan la interpretación de los datos, evidenciando patrones, variaciones y posibles anomalías en el consumo de energía en función de las páginas web visitadas. Los resultados permiten responder a los objetivos planteados, proporcionando una visión clara del impacto energético de distintos sitios web y destacando las limitaciones y potenciales mejoras del sistema.

5.1. Ranking de webs más visitadas en el mundo

Con el objetivo de evaluar el impacto energético de la navegación web sobre páginas de alto tráfico, se seleccionaron las 50 webs más visitadas del mundo según el ranking de Wikipedia[20] (Se adjunta el listado completo en el Anexo B.3). A cada una de ellas se le aplicó el mismo proceso automatizado de carga y visualización, y se registraron los datos de potencia mediante el enchufe inteligente durante cada sesión.

Para obtener una estimación precisa del consumo energético atribuible exclusivamente a la carga de las páginas, se realizó una medición de referencia del consumo en reposo del sistema (es decir, sin actividad de navegación). Se calculó la media de esta medición de fondo y se restó

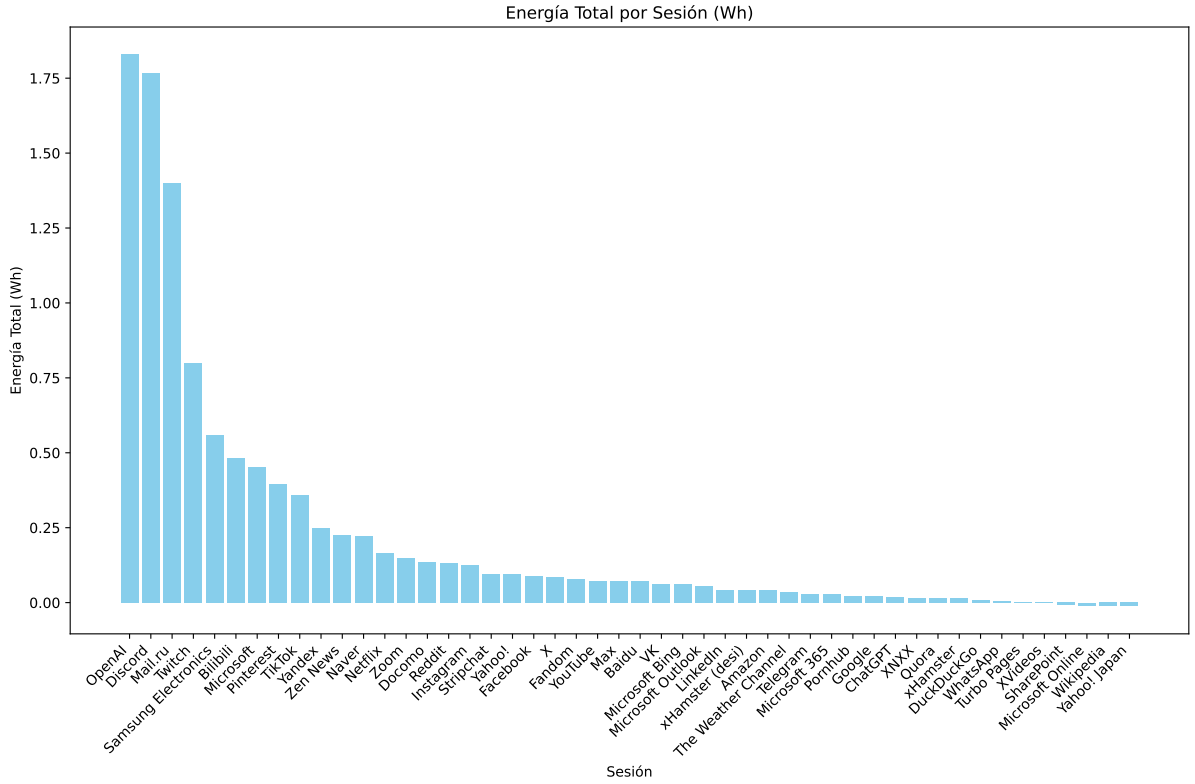


Figura 5.1: Energía total consumida por cada web en orden descendente

a cada uno de los registros de potencia obtenidos durante las sesiones web. De este modo, se eliminó el consumo base del equipo, aislando el gasto energético asociado a la actividad de red y procesamiento inducida por la navegación.

Posteriormente, se estimó el consumo energético total E para cada página mediante la fórmula:

$$E = P_{\text{ajustada}} \cdot t \tag{5.1}$$

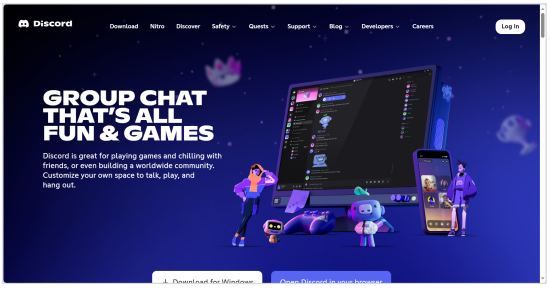
donde P_{ajustada} es la potencia instantánea corregida con la media de fondo (en vatios) y t es el tiempo de navegación considerado para cada página (en segundos).

La gráfica 5.1 representa el consumo energético total (en Wh) durante una sesión de carga y navegación en cada una de las 50 páginas web. Los valores mostrados ya han sido corregidos para descontar el consumo base del equipo en reposo. Cada barra corresponde a una única página y los datos están ordenados de mayor a menor consumo energético.

Entre las páginas que registran un mayor consumo energético destacan OpenAI y Discord, lo cual puede deberse a la complejidad de sus elementos multimedia y a la presencia de múlti-

ples scripts ejecutándose en segundo plano. En particular, ambas páginas incluyen contenido audiovisual que se reproduce automática y repetidamente en sus portadas, lo que incrementa significativamente la demanda de recursos del sistema. En contraste, sitios como Wikipedia o Yahoo! Japan muestran un impacto energético considerablemente menor, incluso por debajo de la media del consumo de fondo, probablemente debido a sus interfaces más ligeras, un diseño centrado en texto y una menor carga de elementos dinámicos. Estas webs comparten una estructura sencilla y optimizada, lo que se refleja en su reducido consumo energético.

(a) OpenAI



(b) Discord

(c) Wikipedia

(d) Yahoo! Japan

Figura 5.2: Páginas web analizadas. Las subfiguras (a) y (b) muestran ejemplos de los sitios con mayor consumo, mientras que (c) y (d) ilustran aquellos con el menor.

Esta representación visual permite observar de forma clara la variabilidad del consumo entre distintas plataformas web, aportando una perspectiva útil sobre el impacto energético de la navegación según el destino.

5.1.1. Análisis temporal del consumo energético

Con el objetivo de analizar el comportamiento energético de cada página web a lo largo del tiempo, se realizó una serie de mediciones durante un periodo continuo de 240 segundos por

sesión.

Las Figuras 5.3, 5.4 y 5.5 muestran la evolución del consumo de potencia eléctrica registrado por el enchufe inteligente (Shelly) a lo largo de distintos periodos de análisis. Cada figura representa el comportamiento energético de las sesiones de navegación durante una duración total de 60, 120 y 240 segundos, respectivamente. En ellas, cada línea corresponde a una sesión específica asociada a una de las páginas web analizadas, y refleja la potencia instantánea registrada a intervalos de muestreo regulares. Estas gráficas permiten comparar visualmente cómo varía el consumo energético en función del tiempo y detectar patrones específicos según el tipo de contenido y comportamiento de cada sitio web.

Estas representaciones permiten observar no solo la magnitud del consumo energético, sino también su comportamiento temporal: si se trata de un pico puntual, un consumo sostenido o una secuencia de picos intermitentes. Con ello, es posible identificar dos patrones generales de comportamiento energético:

- **Carga inicial intensa y estabilización:** páginas que muestran un pico de consumo al principio, coincidiendo con la carga de recursos estáticos (HTML, CSS, JavaScript, imágenes, etc.), seguido de un descenso y estabilización. Este patrón indica que la actividad del navegador disminuye tras completar la carga inicial, lo que sugiere una baja actividad dinámica.

- **Consumo sostenido o variable a lo largo del tiempo:** otras webs mantienen un nivel de consumo elevado de forma continua, o presentan oscilaciones notables durante toda la sesión. Esto puede atribuirse a la presencia de contenido dinámico, como animaciones, reproducción de vídeo en bucle, publicidad interactiva o procesos que actualizan constantemente el contenido de la página (por ejemplo, chats o dashboards).

Por ejemplo, OpenAI muestra un consumo energético elevado y sostenido durante aproximadamente los primeros 150 segundos de la sesión, lo que podría deberse a la ejecución continua de procesos dinámicos en segundo plano. En contraste, páginas como Wikipedia o Yahoo! Japan presentan un pico de consumo al inicio, asociado a la carga de recursos estáticos, seguido de una estabilización en niveles muy bajos, lo que sugiere una arquitectura más estática y eficiente desde el punto de vista energético. Por otro lado, sitios como Stripchat muestran un

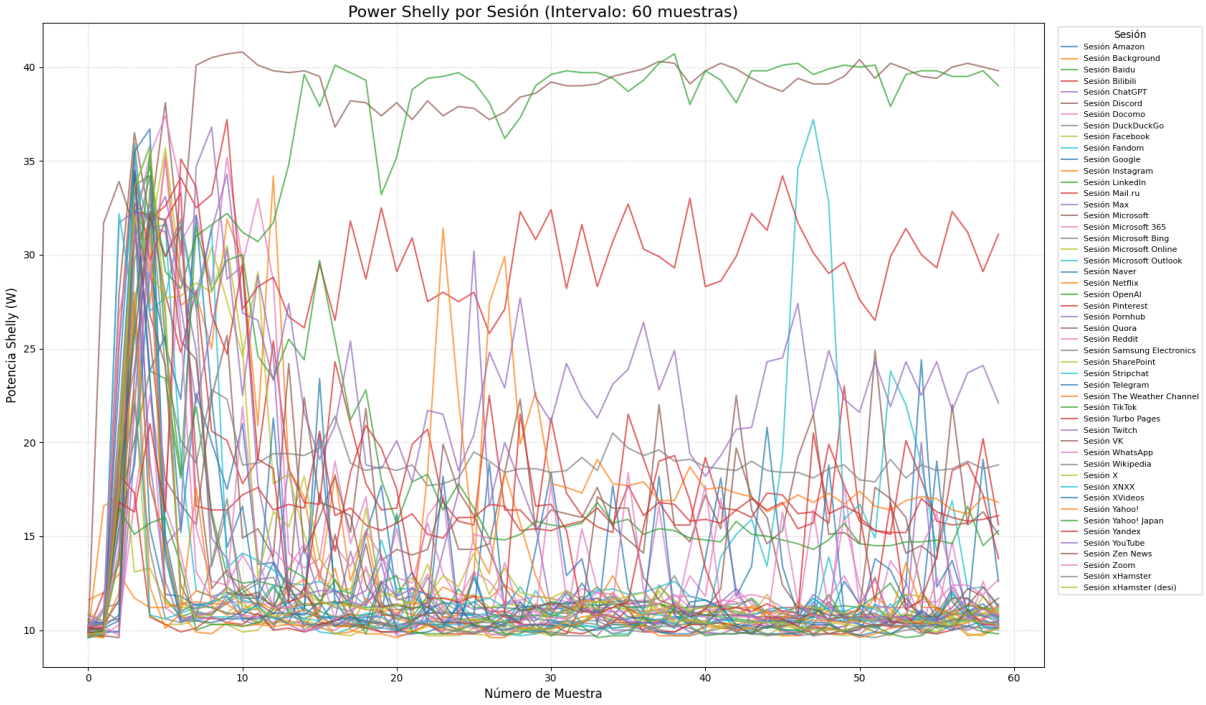


Figura 5.3: Análisis temporal. Periodo = 60 s.

comportamiento intermitente, con picos de consumo elevados aproximadamente cada 50 muestras, mientras que el resto del tiempo mantienen un consumo más moderado y relativamente estable.

Esta representación permite identificar diferencias importantes no visibles en el análisis de consumo total, revelando cómo algunas páginas siguen utilizando recursos del sistema, incluso cuando no hay interacción del usuario.

5.1.2. Relación entre el consumo energético total y el uso de la CPU

Con el objetivo de analizar la relación entre el consumo energético total del sistema y el uso interno de recursos, se calculó la correlación de Pearson entre los datos de potencia obtenidos mediante el enchufe inteligente Shelly y los valores de carga de CPU registrados por el sensor interno del sistema a través de hwmon.

Para cada una de las 50 páginas web analizadas, se extrajeron las series temporales de consumo energético (en vatios) y de porcentaje de uso de CPU durante la sesión. A partir de estas series se calcularon dos métricas:

- **Coeficiente de correlación de Pearson** entre la potencia registrada por Shelly y el uso

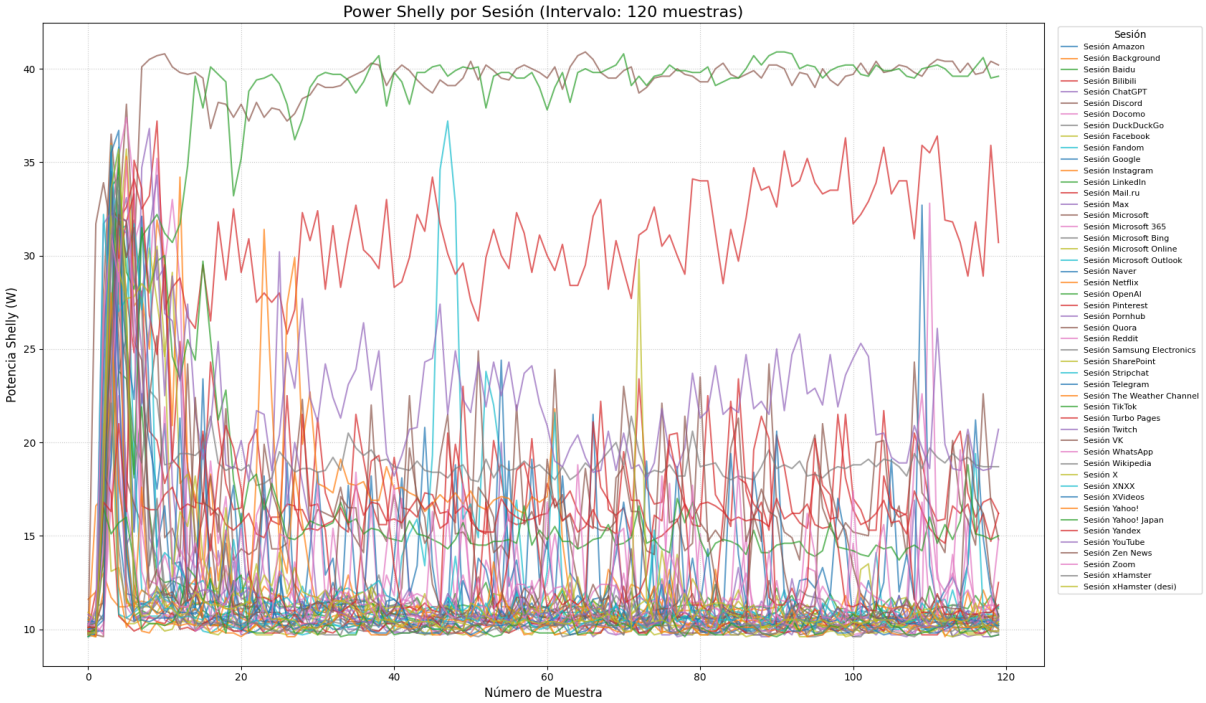


Figura 5.4: Análisis temporal. Periodo = 120 s.

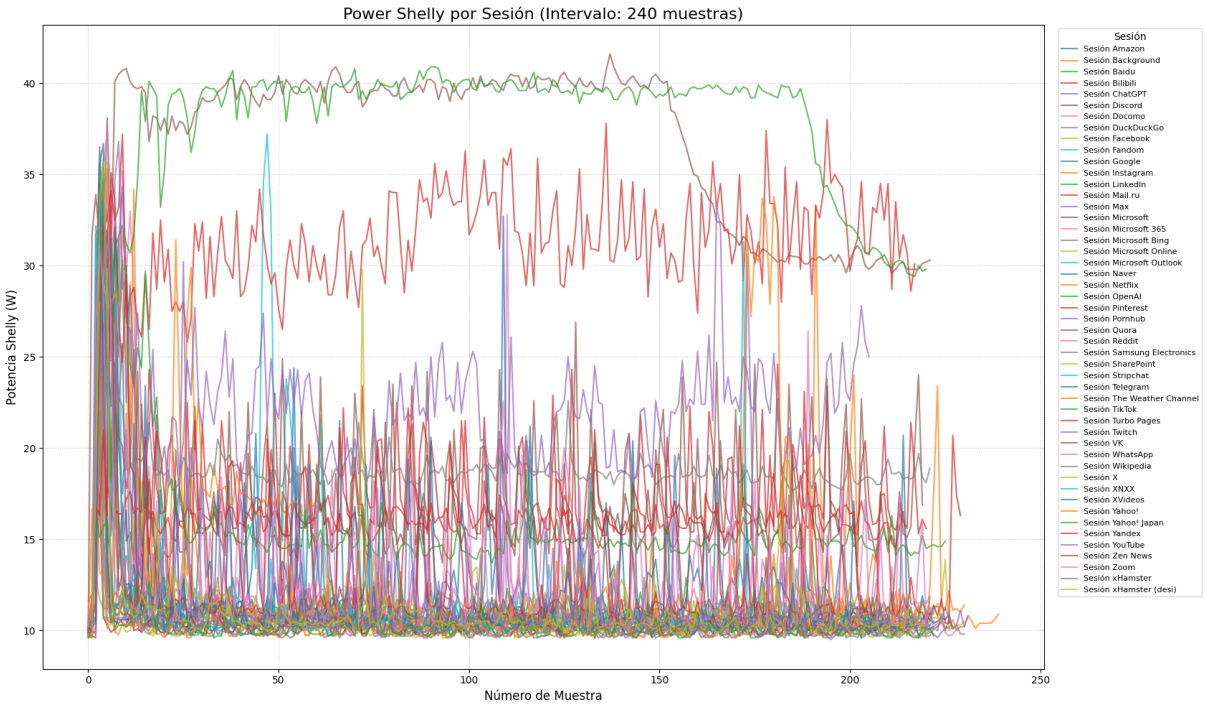


Figura 5.5: Análisis temporal. Periodo = 240 s.

de CPU (hwmon). Este valor indica el grado de asociación lineal entre ambos parámetros durante la sesión de navegación.

$$r_{XY} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{5.2}$$

- **Porcentaje medio de uso de CPU** sobre el consumo energético total, expresado como:

$$\text{Relación CPU/Shelly} = \frac{P_{\text{CPU}}}{P_{\text{total}}} \times 100 \tag{5.3}$$

donde P_{CPU} representa la potencia media estimada del uso de CPU y P_{total} la potencia media total registrada por Shelly.

Los resultados obtenidos se presentan en forma de tablas y gráficas comparativas para cada web. Esta visualización permite identificar qué páginas presentan una mayor correspondencia entre la carga de CPU y el consumo energético total, y cuáles implican un uso significativo de otros recursos del sistema (GPU, red, disco, etc.) o de componentes externos al procesador.

Correlación entre consumo energético total y uso de CPU

Para evaluar el grado de asociación entre el consumo energético total medido externamente (Shelly) y la carga interna del sistema (CPU), se calcularon los coeficientes de correlación de Pearson para cada sesión de navegación. En este análisis se compararon las series temporales de potencia registrada por el enchufe Shelly y el porcentaje de uso de CPU reportado por hwmon.

La Tabla 5.1 muestra los coeficientes de correlación para las 50 páginas web más visitadas. Valores cercanos a 1 indican una asociación lineal fuerte entre la actividad de la CPU y el consumo energético total, mientras que valores próximos a 0 reflejan baja correlación.

En general, la mayoría de webs presentan una correlación moderada entre ambos parámetros, destacando casos como Facebook (0.78), Yahoo! (0.77) e Instagram (0.68), donde el comportamiento de la CPU está estrechamente vinculado al consumo energético del sistema. En cambio, otras webs como Discord (0.02), OpenAI (0.13) y Samsung Electronics (-0.29) presentan una correlación muy baja o incluso negativa, lo que sugiere que otros componentes (como la GPU, red o procesos externos) podrían tener un mayor peso en su demanda energética.

Las gráficas de dispersión correspondientes a cada una de las 50 sesiones individuales se han incluido en el Anexo C.1. Estas visualizaciones permiten examinar con mayor detalle el

Tabla 5.1: Correlación de Pearson entre Shelly y CPU (hwmon) por página web

Sesión	**Correlación**	**Sesión**	**Correlación**
Google	0.24	Microsoft Bing	0.41
YouTube	0.64	Zen News	0.67
Facebook	0.78	Microsoft Online	0.40
Instagram	0.68	Pinterest	0.21
Baidu	0.39	Naver	0.32
Wikipedia	0.26	Turbo Pages	0.41
X	0.53	Bilibili	0.33
Yahoo!	0.77	VK	0.59
Yandex	0.39	Mail.ru	0.13
WhatsApp	0.48	Samsung Electronics	-0.29
XVideos	0.45	Discord	0.02
ChatGPT	0.47	Max	0.66
TikTok	0.56	Microsoft	0.23
Reddit	0.57	xHamster (desi)	0.54
Amazon	0.63	The Weather Channel	0.52
Pornhub	0.55	Twitch	0.36
Yahoo! Japan	0.23	Telegram	0.54
Docomo	0.66	Quora	0.35
Microsoft Outlook	0.50	SharePoint	0.24
LinkedIn	0.51	Fandom	0.55
XNXX	0.43	OpenAI	0.13
Netflix	0.60	Stripchat	0.50
Microsoft 365	0.26	DuckDuckGo	0.23
xHamster	0.43	Zoom	0.61

comportamiento específico de cada página web en relación con su consumo energético y el uso de CPU.

Proporción del uso de CPU respecto al consumo total

Además de la correlación, se calculó el porcentaje medio del uso de CPU sobre el consumo energético total registrado por Shelly. La Tabla 5.2 recoge este valor para cada web, expresado en porcentaje.

Es fundamental destacar que estos porcentajes se basan en las lecturas del sensor fam15h_power (hwmon), que, como se mencionó previamente, reportó valores anómalos y subestimados para el consumo de la CPU. Por lo tanto, los valores presentados a continuación deben interpretarse con cautela, entendiendo que no representan el porcentaje real del consumo total de la CPU, sino la proporción basada en las mediciones de este sensor específico.

En todos los casos, la contribución del uso de CPU se encontró entre el 0.22 % y el 0.31 % del consumo total (según las lecturas de hwmon). Este hallazgo, aunque basado en datos subestimados de la CPU, es indicativo de que la CPU, por sí sola, no es el componente principal responsable del consumo energético general en este tipo de carga de trabajo. Por el contrario, la mayor parte del consumo total medido por Shelly puede provenir de otros subsistemas del ordenador, como la GPU, el disco o la actividad de red. Los sitios web que registraron los porcentajes más altos de contribución de la CPU (según hwmon) fueron Mail.ru, Microsoft y Zen News (0.31 %), mientras que el más bajo fue Samsung Electronics (0.22 %).

Correlación global durante toda la medición

Además de los valores individuales, se generó una gráfica de dispersión global para visualizar la correlación general entre el uso de CPU y el consumo energético total durante todas las sesiones combinadas. En ella se representa cada punto como una muestra temporal con sus valores correspondientes de uso de CPU y potencia registrada por Shelly.

La Figura 5.6 muestra la relación general entre ambos parámetros a lo largo de todas las sesiones. La tendencia lineal observada sugiere que existe una correlación significativa entre la carga de CPU y el consumo energético global, aunque con variabilidad dependiente del tipo de contenido web.

La gráfica muestra una tendencia general ascendente, indicando una correlación positiva entre ambos parámetros. El coeficiente de correlación global obtenido es de 0.7, lo que refleja una asociación moderadamente fuerte entre el uso de CPU y el consumo total del sistema durante

Tabla 5.2: Porcentaje del consumo del CPU respecto al consumo total para cada web

Sesión	**Porcentaje %**	**Sesión**	**Porcentaje %**
Amazon	0.26	Instagram	0.27
Baidu	0.27	LinkedIn	0.27
Bilibili	0.30	Mail.ru	0.31
ChatGPT	0.28	Max	0.26
Discord	0.25	Microsoft	0.31
Docomo	0.25	Microsoft 365	0.27
DuckDuckGo	0.26	Microsoft Bing	0.29
Facebook	0.27	Microsoft Online	0.27
Fandom	0.28	Microsoft Outlook	0.28
Google	0.28	Naver	0.29
Netflix	0.26	Pinterest	0.30
OpenAI	0.30	Pornhub	0.27
Quora	0.25	Reddit	0.27
Samsung Electronics	0.22	SharePoint	0.25
Stripchat	0.26	Telegram	0.28
The Weather Channel	0.27	TikTok	0.26
Turbo Pages	0.27	Twitch	0.24
VK	0.25	WhatsApp	0.26
Wikipedia	0.25	X	0.26
XNXX	0.25	XVideos	0.27
Yahoo!	0.25	Yahoo! Japan	0.26
Yandex	0.27	YouTube	0.26
Zen News	0.31	Zoom	0.25
xHamster	0.28	xHamster (desi)	0.26

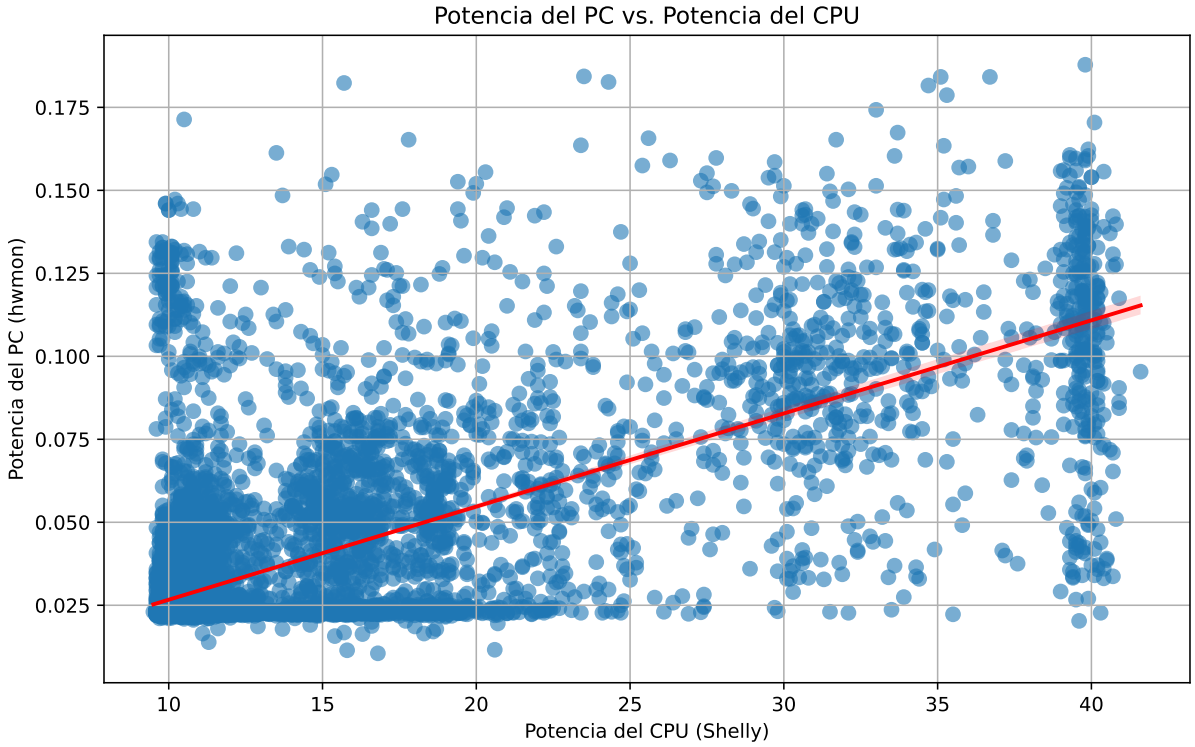


Figura 5.6: Correlación global entre uso de CPU y consumo energético total

las sesiones de navegación.

5.1.3. Conclusión de resultados

El análisis conjunto del comportamiento temporal del consumo energético y de la correlación entre el uso de CPU y el consumo total del sistema ha permitido obtener una visión detallada del impacto energético de la navegación web.

Por un lado, el estudio temporal reveló la existencia de varios patrones energéticos característicos según el tipo de web: desde un consumo constante (alto o bajo), hasta picos iniciales seguidos de estabilización, o bien picos intermitentes a lo largo del tiempo. Estas diferencias reflejan la complejidad técnica, el tipo de contenido y la dinámica de actualización de cada sitio. Por ejemplo, páginas como OpenAI o Discord presentan un consumo sostenido elevado, posiblemente debido a procesos activos en segundo plano, mientras que otras como Wikipedia o Yahoo! Japan destacan por un comportamiento más eficiente, con un pico inicial seguido de estabilidad.

Por otro lado, el análisis de correlación entre la potencia registrada por el enchufe inteli-

gente (Shelly) y el uso de CPU interno (hwmon) reveló una correlación global moderadamente fuerte (r = 0.7), aunque con grandes diferencias entre webs. Algunas, como Facebook o Yahoo!, mostraron una relación alta entre la actividad de CPU y el consumo energético, mientras que otras, como Discord, OpenAI o Samsung Electronics, presentaron correlaciones muy bajas o negativas.

Es crucial recalcar que los valores de porcentaje medio del uso de la CPU respecto al consumo total se basaron en las lecturas del sensor interno, el cual reportó valores anómalos y subestimados para el consumo de la CPU, lo que impide realizar una estimación precisa del porcentaje real que representa respecto al consumo global. A pesar de esta limitación, los datos disponibles permiten sugerir que, en condiciones normales de navegación web, el consumo energético atribuido a la CPU representa una fracción menor en comparación con el consumo total medido por el enchufe inteligente. Esta observación es consistente con la naturaleza de las tareas ejecutadas, que no generan una carga intensiva sobre la CPU y probablemente involucran otros componentes del sistema como el almacenamiento, la red o la GPU integrada.

Estos resultados subrayan la necesidad de emplear tanto mediciones externas como internas para comprender de forma completa el perfil energético de una sesión de navegación. La naturaleza del contenido web, la arquitectura del sitio y el uso de recursos multimedia o dinámicos son factores clave que condicionan la demanda energética, más allá de la simple carga de CPU. Esta comprensión es esencial para avanzar hacia el diseño de servicios web más sostenibles y energéticamente eficientes.

Capítulo 6

Conclusiones

A lo largo de este Trabajo de Fin de Grado se ha llevado a cabo un estudio detallado sobre el impacto energético que supone la navegación web, combinando mediciones externas de potencia eléctrica con datos internos de uso de recursos del sistema. Este análisis ha permitido caracterizar de forma precisa el comportamiento energético de varias webs, evaluando no solo el consumo total por sesión, sino también su evolución temporal y su relación con la carga de CPU.

Los resultados obtenidos evidencian que la navegación web, incluso en ausencia de interacción directa del usuario, puede implicar un consumo energético significativo y sostenido en el tiempo. Se han identificado patrones de comportamiento distintos entre sitios: algunos con consumo elevado y constante, otros con picos iniciales asociados a la carga de recursos, y otros con actividad intermitente debida a la presencia de contenido dinámico o multimedia. Estos patrones reflejan la creciente complejidad de las plataformas web modernas y su dependencia de tecnologías que intensifican la demanda energética del sistema.

El análisis de correlación entre el uso de CPU y el consumo energético total mostró que, si bien existe una asociación moderada a nivel global, la CPU representa solo una fracción del consumo total del sistema, lo que pone de manifiesto el papel relevante de otros subsistemas como la GPU, la red o el almacenamiento en el impacto energético de la navegación. Este hallazgo subraya la necesidad de adoptar una perspectiva holística al evaluar la eficiencia energética del software, considerando no solo el uso de CPU sino el conjunto de recursos involucrados.

En un contexto de creciente preocupación por la sostenibilidad digital, este trabajo aporta una metodología reproducible y una base de conocimiento útil para reflexionar sobre el diseño,

optimización y evaluación energética de páginas web. Los resultados invitan a considerar la eficiencia energética como un criterio más en el desarrollo web, al mismo nivel que la accesibilidad, la usabilidad o el rendimiento.

6.1. Valoración personal y conclusiones del proyecto

La realización de este Trabajo de Fin de Grado ha supuesto un reto tanto a nivel técnico como personal, permitiéndome aplicar conocimientos adquiridos a lo largo de la carrera en un proyecto real, multidisciplinar y con impacto en el ámbito de la sostenibilidad digital.

A lo largo del desarrollo del sistema he podido consolidar competencias clave como la programación en Python, la lectura y análisis de datos desde sensores del sistema, el tratamiento de series temporales y el análisis estadístico. También he aprendido a trabajar con APIs de dispositivos externos, como el enchufe inteligente Shelly, y a sincronizar correctamente múltiples fuentes de datos heterogéneos.

Uno de los aspectos más valiosos de este proyecto ha sido enfrentarme a la integración de herramientas de bajo nivel, como **hwmon**, con fuentes externas de medición, combinando hardware y software de forma práctica. Este proceso me ha enseñado a resolver problemas técnicos con autonomía, documentarme eficazmente y validar los resultados de manera rigurosa.

En cuanto a las dificultades encontradas, cabe destacar la identificación y configuración de sensores del sistema, la sincronización precisa entre distintas fuentes de datos y la interpretación estadística de los resultados. Sin embargo, superar estos obstáculos ha reforzado mi capacidad de análisis, organización y toma de decisiones.

Finalmente, considero que este proyecto me ha permitido tener una visión más crítica y consciente del impacto energético que puede tener incluso una actividad tan cotidiana como la navegación web. Este conocimiento abre la puerta a futuras líneas de investigación y desarrollo en el ámbito de la eficiencia energética del software, un campo cada vez más relevante en el contexto actual.

6.2. Consecución de objetivos

El objetivo general de este Trabajo de Fin de Grado consistía en diseñar y desarrollar una herramienta que permitiera medir y analizar el impacto energético de la navegación web, combinando datos provenientes del sistema operativo y de un enchufe inteligente. Este objetivo se considera alcanzado. Se ha implementado un sistema funcional capaz de automatizar la navegación por páginas web, recopilar datos energéticos desde múltiples fuentes, procesarlos y representar visualmente los resultados para facilitar su análisis.

Respecto a los objetivos específicos definidos al inicio del proyecto, se puede detallar lo siguiente:

- **Automatizar la navegación por sitios web definidos por el usuario.** Este objetivo se ha cumplido mediante la implementación de un script en Python que permite cargar una lista de URLs desde un archivo JSON y navegar automáticamente por ellas utilizando el navegador Firefox a través del módulo `subprocess`.

- **Extraer información sobre el consumo energético desde el sistema operativo.** Se ha logrado integrar la lectura de datos energéticos desde el subsistema `hwmon` del kernel de Linux. No obstante, se detectaron limitaciones en la fiabilidad de ciertos sensores, como `fam15h_power`, que reportaban valores anómalamente bajos. A pesar de ello, se conservaron estas lecturas como medida comparativa para validar tendencias generales.

- **Medir el consumo energético total del sistema con un enchufe inteligente.** Este objetivo se ha cumplido satisfactoriamente mediante el uso de un enchufe inteligente Shelly, que permitió obtener lecturas precisas de consumo global a través de su API REST, integradas correctamente con el sistema de captura.

- **Sincronizar y procesar los datos energéticos obtenidos.** Se desarrolló un sistema de normalización temporal e integración de datos que permitió alinear las muestras de ambas fuentes, incluso con diferentes frecuencias de muestreo. Esto facilitó un análisis conjunto de los datos y la obtención de métricas significativas.

- **Visualizar los resultados mediante representaciones gráficas.** Este objetivo también fue alcanzado mediante el uso de bibliotecas como Matplotlib y Seaborn. Se generaron

visualizaciones variadas (series temporales, diagramas de barras, gráficos de dispersión y boxplots) que facilitaron la interpretación de los datos y la detección de patrones de consumo.

- **Evaluar la correlación entre fuentes de medición.** Se utilizó la función `pearsonr` de `scipy.stats` para calcular la correlación lineal entre las mediciones del enchufe y del sistema operativo. Aunque los valores absolutos del sensor de CPU no fueron fiables, las tendencias sí mostraron cierta consistencia, permitiendo realizar un análisis relativo útil.

- **Extraer conclusiones sobre el impacto energético de la navegación web.** A partir del análisis de los datos obtenidos y su representación gráfica, fue posible identificar diferencias de consumo entre páginas web, así como inferir el comportamiento energético de diferentes estados del sistema (reposo y actividad). Esto permitió alcanzar el objetivo de extraer conclusiones fundamentadas.

En resumen, todos los objetivos planteados han sido alcanzados en mayor o menor medida. Las limitaciones encontradas, principalmente relacionadas con la precisión del sensor del sistema, fueron identificadas y mitigadas mediante el uso de fuentes complementarias de datos y un análisis crítico de los resultados.

6.3. Aplicación de lo aprendido

A lo largo del desarrollo del Trabajo de Fin de Grado, se han puesto en práctica numerosos conocimientos y competencias adquiridos durante la carrera, especialmente aquellos relacionados con la programación, el análisis de datos y la comprensión de sistemas audiovisuales e informáticos.

Las asignaturas **Informática I** e **Informática II** fueron fundamentales para asentar una base sólida en programación, especialmente en Python, lenguaje central en el desarrollo del proyecto. Gracias a estas asignaturas, se adquirió fluidez en el uso del lenguaje, así como familiaridad con bibliotecas ampliamente utilizadas como numpy, pandas y matplotlib, las cuales resultaron esenciales para el procesamiento de datos y la representación gráfica de los resultados.

La asignatura **Protocolos para la Transmisión de Audio y Video en Internet** ayudó a profundizar en el ecosistema Python desde una perspectiva aplicada al sector audiovisual, fo-

mentando el análisis técnico de los flujos de datos en entornos conectados. Esta experiencia permitió comprender mejor el contexto de ejecución de sistemas complejos y distribuidos, como el desarrollado en este TFG.

Desde el punto de vista analítico, la asignatura **Estadística para Sistemas Audiovisuales** proporcionó herramientas esenciales para interpretar rigurosamente los datos obtenidos. Conceptos como media, mediana, desviación estándar y correlación fueron fundamentales para evaluar la fiabilidad de las fuentes de medición y para comparar el comportamiento energético de diferentes páginas web.

Asimismo, asignaturas como **Tecnologías de Televisión en Internet** y **Construcción de Servicios y Aplicaciones Audiovisuales en Internet**, si bien no se centraron directamente en los aspectos técnicos tratados en el TFG, sí resultaron valiosas como fuente de inspiración. Estas materias ofrecieron una visión actualizada sobre las tendencias del consumo digital, el diseño de servicios en red y la eficiencia en la entrega de contenidos, despertando el interés por explorar la relación entre plataformas web y consumo energético, eje central del presente proyecto.

En conjunto, el TFG ha permitido integrar de forma práctica y crítica competencias adquiridas a lo largo de la carrera, consolidando habilidades en desarrollo de software, análisis de datos, automatización y evaluación de sistemas audiovisuales en red.

6.4. Lecciones aprendidas

El desarrollo del Trabajo de Fin de Grado ha supuesto una experiencia formativa integral, en la que se han puesto a prueba tanto conocimientos técnicos como habilidades personales y metodológicas.

Una de las principales lecciones aprendidas ha sido la importancia de la planificación y la gestión del tiempo. Coordinar tareas como la recolección de datos, la automatización de pruebas, el análisis estadístico y la redacción de documentación requirió una organización rigurosa y un enfoque iterativo. Adaptarse a los contratiempos, como problemas de compatibilidad o pérdida de datos, también fomentó la flexibilidad y la toma de decisiones fundamentadas.

Desde el punto de vista técnico, el proyecto permitió afianzar el dominio de Python y sus bibliotecas orientadas al análisis de datos, así como profundizar en el uso de herramientas del

sistema operativo Linux para la monitorización de hardware mediante interfaces como hwmon. Asimismo, se consolidaron habilidades en la automatización de procesos, el tratamiento de datos heterogéneos y su representación gráfica para la extracción de conclusiones.

A nivel metodológico, se aprendió a trabajar con datos reales, imperfectos y a menudo ruidosos, lo que implicó aplicar técnicas de limpieza, normalización y validación. También se adquirió experiencia en la interpretación crítica de resultados, identificando limitaciones en las fuentes de medición y contrastando datos desde diferentes perspectivas.

Finalmente, el TFG reforzó la capacidad de integrar conocimientos de distintas áreas del grado para abordar un problema técnico con impacto real. La combinación de informática, estadística, sistemas operativos y visión crítica sobre la eficiencia energética en entornos digitales permitió desarrollar un proyecto transversal y con sentido aplicado.

6.5. Trabajos futuros

El presente Trabajo de Fin de Grado ha sentado las bases para el estudio del consumo energético asociado a la navegación web desde una perspectiva práctica y reproducible. No obstante, se identifican diversas líneas de mejora y ampliación que podrían abordarse en desarrollos futuros.

En primer lugar, sería relevante integrar sensores de medición más precisos y específicos para cada componente del sistema, como dispositivos de medición externa conectados directamente a la CPU o GPU, o herramientas como Intel RAPL o AMD uProf, que permiten estimaciones más fiables del consumo energético a nivel de microarquitectura. Esto permitiría superar las limitaciones observadas con hwmon y mejorar la granularidad de los datos.

También el ampliar el número de sensores considerados en la medición del consumo energético. El análisis actual se centró en el sensor `fam15h_power`, orientado a la CPU, pero futuras versiones del sistema podrían integrar lecturas provenientes de otros sensores accesibles mediante `hwmon`, como los relacionados con la GPU, discos duros, controladores de voltaje, ventiladores o temperatura del chipset. Esta integración permitiría obtener una visión más completa y detallada del comportamiento energético del sistema durante la navegación web.

También se plantea como mejora la ampliación del conjunto de páginas analizadas, incorporando sitios con diferentes niveles de complejidad: desde aplicaciones web dinámicas hasta

plataformas multimedia intensivas, como servicios de streaming o redes sociales. Esta diversidad permitiría identificar patrones energéticos más variados y evaluar el impacto real de cada tipo de contenido web.

Otra línea de trabajo prometedora es la inclusión de múltiples navegadores (como Chrome, Chromium, Edge, Brave o incluso navegadores en modo texto como Lynx o w3m), lo cual permitiría comparar la eficiencia energética entre distintos motores de renderizado, configuraciones y modos de navegación (gráfico versus texto, por ejemplo). Asimismo, se podría extender el análisis a distintos equipos con diferentes configuraciones de hardware, evaluando el impacto del tipo de procesador, la presencia o ausencia de GPU dedicada, la arquitectura del sistema y las tecnologías de gestión de energía propias de cada plataforma. Esta comparación entre entornos permitiría entender mejor la influencia del hardware en el consumo energético asociado a la navegación web, y abriría la puerta a recomendaciones más específicas y contextualizadas según el dispositivo.

Desde el punto de vista metodológico, se podría desarrollar un sistema de análisis en tiempo real que integre y sincronice todas las fuentes de datos (enchufe inteligente, sensores internos y navegador), junto con una visualización interactiva que facilite la interpretación inmediata de eventos de alto consumo.

Finalmente, una extensión con gran potencial consiste en aplicar técnicas de aprendizaje automático para clasificar páginas web según su perfil energético o predecir su consumo a partir de características estructurales y métricas de tráfico. Este tipo de aproximación podría tener aplicaciones prácticas tanto en la optimización del diseño web como en la concienciación sobre sostenibilidad digital.

De hecho, los resultados de este proyecto podrían servir como base para campañas de sensibilización dirigidas a usuarios y desarrolladores web, fomentando prácticas de diseño más eficientes y sostenibles. En un contexto global donde la huella energética de las tecnologías digitales crece constantemente, promover una navegación web más consciente puede contribuir de forma significativa a la reducción del consumo energético a gran escala.

6.5.1. Limitaciones y posibles mejoras

Aunque el sistema desarrollado ha permitido realizar un análisis detallado del consumo energético asociado a la navegación web, existen una serie de limitaciones técnicas que podrían

abordarse en futuros trabajos para mejorar la precisión, escalabilidad y alcance del estudio.

- **Medición global de potencia:** El uso de un enchufe inteligente como Shelly Plug S ofrece una visión global del consumo energético del equipo, pero no permite distinguir entre los distintos componentes del sistema (CPU, GPU, red, almacenamiento, etc.). Esto limita la capacidad de atribuir el consumo a procesos específicos.
- **Granularidad temporal:** Las mediciones se realizaron con un intervalo de muestreo de 1 segundo. Si bien esto fue suficiente para el análisis general, una mayor frecuencia de muestreo permitiría detectar eventos más breves o picos de consumo transitorios.
- **Dependencia del entorno de pruebas:** Las mediciones están condicionadas por el hardware y el sistema operativo utilizados. Diferencias en la eficiencia energética de los componentes o en la configuración del navegador pueden influir en los resultados. Esto limita la generalización de las conclusiones a otros entornos.
- **Limitaciones del sensor hwmon:** La lectura desde `hwmon` proporciona valores aproximados de consumo de CPU, pero puede variar según el soporte del hardware y la precisión del driver utilizado. Además, no se capturan métricas de GPU u otros dispositivos.
- **Carga inicial del navegador:** Algunas sesiones incluyeron la apertura del navegador en cada prueba. Aunque se trató de aislar esta carga, parte del consumo podría estar influido por procesos de inicialización ajenos a la web evaluada.

Como posibles líneas de mejora, se propone:

- Integrar herramientas avanzadas como `perf`, `powerstat` o interfaces como RAPL para obtener mediciones energéticas más específicas a nivel de componentes.
- Añadir soporte para analizar el impacto energético de distintos navegadores o configuraciones (por ejemplo, modo oscuro, bloqueadores de anuncios, renderizado sin GPU).
- Realizar el estudio en múltiples equipos y arquitecturas para obtener una visión más general del impacto energético de la navegación web.
- Automatizar completamente la ejecución y el análisis mediante scripts que incluyan gestión de errores, visualización directa y exportación de informes.

Estas mejoras permitirían ampliar el alcance del sistema desarrollado y aumentar su utilidad en futuros estudios relacionados con sostenibilidad digital, eficiencia energética del software o análisis de rendimiento web.

Anexo A

Código fuente

```python
import sys
import time
import json
import csv
import subprocess
import glob

OUTPUT_FILE = "power_log_hwmon.csv"
INTERVAL = 0.5
DURATION = 60
PAUSE = 10

if len(sys.argv) > 2:
    print("Uso: python3 power_logger_hwmon.py <webs.json>")
    sys.exit()

webs_json = sys.argv[1]

def get_power_field():
    hwmon_paths = glob.glob("/sys/class/hwmon/hwmon*/name")
    for path in hwmon_paths:
        with open(path, "r") as f:
```

```
23              if "fam15h_power" in f.read().strip():
24                  return path.replace("name", "power1_input")
25      return None
26
27  POWER_FIELD = get_power_field()
28
29  if POWER_FIELD is None:
30      print(f'Error: El archivo de consumo de potencia no existe en el
        ↪  sistema.')
31      exit(1)
32
33  print("Starting power meter: hwmon")
34
35  with open(webs_json) as file:
36      URLS = json.load(file)
37
38  with open(OUTPUT_FILE, 'w', newline='') as file:
39      writer = csv.writer(file)
40      writer.writerow(["timestamp", "power", "session"])
41
42  def save_power(session):
43      start_time = time.time()
44      try:
45          while time.time() - start_time < DURATION:
46              with open(POWER_FIELD, 'r') as file:
47                  power = int(file.read().strip()) / 1_000_000
48
49              timestamp = time.strftime("%Y%m%dT%H:%M:%S")
50
51              with open(OUTPUT_FILE, 'a', newline='') as file:
52                  writer = csv.writer(file)
53                  writer.writerow([timestamp, power, session])
54
```

```
55              print(f'{timestamp} - {power:.2f} W - {session}')
56              time.sleep(INTERVAL)
57      except KeyboardInterrupt:
58          print('\nStopping capture...')
59
60  print("\n Background")
61  save_power("Background")
62  time.sleep(PAUSE)
63
64  print("\n BlankTab")
65  subprocess.Popen(["firefox", "--new-window", "about:blank"],
66                   stdout=subprocess.DEVNULL,
                     ↪  stderr=subprocess.DEVNULL)
67  time.sleep(DURATION)
68  save_power("BlankTab")
69  time.sleep(PAUSE)
70
71  for name, url in URLS.items():
72      print(f'\n {name}')
73      subprocess.run(["firefox", "--new-tab", url],
74                     stdout=subprocess.DEVNULL,
                       ↪  stderr=subprocess.DEVNULL)
75      save_power(name)
76      subprocess.run(["xdotool", "key", "Ctrl+w"],
77                     stdout=subprocess.DEVNULL,
                       ↪  stderr=subprocess.DEVNULL)
78      time.sleep(PAUSE)
79
80  subprocess.run(["pkill", "-f", "firefox"],
    ↪  stdout=subprocess.DEVNULL, stderr=subprocess.DEVNULL)
81  print(f"\nCapture completed")
```

Código A.1: Sistema de medición de Hwmon y navegación automática

```python
import requests
import time
import csv

IP_SHELLY = "***.***.*.**"
OUTPUT_FILE = "power_log_shelly.csv"
INTERVAL = 0.5

with open(OUTPUT_FILE, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["timestamp", "power"])

print("Starting power meter: shelly")

def get_power():
    url = f"http://{IP_SHELLY}/rpc/Switch.GetStatus?id=0"
    try:
        response = requests.get(url, timeout=5)
        if response.status_code == 200:
            data = response.json()
            return {
                "timestamp": time.strftime("%Y%m%dT%H:%M:%S"),
                "power": data.get("apower", 0.0)
            }
        else:
            print(f"Error while getting status:
             ↪  {response.status_code}")
            return None
    except requests.RequestException as e:
        print(f"Connection error while getting status: {e}")
        return None

```

```
with open(OUTPUT_FILE, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["timestamp", "power"])

print("Start test")
try:
    while True:
        data = get_power()
        if data:
            with open(OUTPUT_FILE, mode='a', newline='') as file:
                writer = csv.writer(file)
                writer.writerow([
                    data["timestamp"],
                    data["power"]
                ])
            print(f'{data["timestamp"]} - {data["power"]} W')

        time.sleep(INTERVAL)
except KeyboardInterrupt:
    print(f"Stopping capture")

print(f"\nCapture completed")
```

Código A.1: Sistema de medición de Shelly

Anexo B

Datos

Tabla B.1: Estadísticas del consumo del fondo

Run	Media		Mediana		STD	
	Shelly	**Hwmon**	**Shelly**	**Hwmon**	**Shelly**	**Hwmon**
01	11.11	0.026	11.05	0.023	0.52	0.006
02	11.06	0.027	11.00	0.023	0.36	0.010
03	12.77	0.030	11.20	0.023	2.90	0.016
04	11.07	0.026	11.05	0.023	0.47	0.008
05	11.05	0.026	11.00	0.023	0.42	0.009
06	11.05	0.024	11.10	0.023	0.39	0.003
07	11.05	0.025	11.10	0.023	0.37	0.006
08	10.51	0.029	10.50	0.023	0.36	0.017
09	10.62	0.024	10.60	0.023	0.45	0.005
10	10.58	0.025	10.60	0.023	0.39	0.005
11	10.05	0.027	10.00	0.023	0.36	0.014
12	10.08	0.026	10.00	0.023	0.45	0.009
13	9.99	0.025	9.90	0.023	0.30	0.006
14	9.98	0.025	9.90	0.023	0.30	0.007
15	9.95	0.029	9.80	0.023	0.28	0.016
16	10.00	0.025	9.90	0.023	0.39	0.006

Continúa en la siguiente página

Tabla B.1 – Continuación de la página anterior

Run	Media		Mediana		STD	
	Shelly	**Hwmon**	**Shelly**	**Hwmon**	**Shelly**	**Hwmon**
17	10.06	0.025	10.00	0.023	0.37	0.007
18	10.05	0.025	10.00	0.023	0.45	0.006
19	9.96	0.025	9.90	0.023	0.31	0.007
20	9.95	0.026	9.85	0.023	0.31	0.008
21	10.61	0.029	9.80	0.023	3.09	0.017
22	10.00	0.025	10.00	0.023	0.36	0.006
23	12.35	0.033	10.10	0.023	6.37	0.023
24	10.00	0.025	9.90	0.023	0.37	0.007
25	9.99	0.025	10.00	0.023	0.33	0.007
26	10.01	0.026	10.00	0.023	0.43	0.008
27	10.07	0.026	10.00	0.023	0.34	0.007
28	10.17	0.025	10.10	0.023	0.43	0.009
29	10.16	0.025	10.10	0.023	0.37	0.006
30	11.43	0.028	10.30	0.023	3.71	0.015
31	10.24	0.025	10.20	0.023	0.37	0.006
32	10.24	0.027	10.25	0.023	0.34	0.009
33	11.21	0.028	10.30	0.023	3.84	0.014
34	10.32	0.026	10.30	0.023	0.38	0.006
35	10.32	0.025	10.30	0.023	0.37	0.007
36	10.31	0.026	10.20	0.023	0.38	0.009
37	10.34	0.024	10.30	0.023	0.44	0.004
38	10.31	0.024	10.25	0.023	0.36	0.006
39	10.26	0.025	10.10	0.023	0.38	0.008
40	10.30	0.026	10.10	0.023	0.39	0.008
41	10.30	0.025	10.20	0.023	0.37	0.008
42	10.26	0.025	10.20	0.023	0.32	0.005
43	10.37	0.025	10.35	0.023	0.39	0.006

Continúa en la siguiente página

Tabla B.1 – Continuación de la página anterior

Run	Media		Mediana		STD	
	Shelly	**Hwmon**	**Shelly**	**Hwmon**	**Shelly**	**Hwmon**
44	10.32	0.026	10.30	0.023	0.33	0.008
45	10.31	0.026	10.20	0.023	0.34	0.007
46	11.13	0.033	10.40	0.023	3.31	0.021
47	10.42	0.026	10.40	0.023	0.41	0.007
48	10.28	0.026	10.20	0.023	0.45	0.009
49	10.21	0.025	10.30	0.023	0.30	0.005
50	10.18	0.026	10.20	0.023	0.24	0.010
Overall	10.46	0.026	10.20	0.023	1.50	0.010

Tabla B.2: Estadísticas del consumo de la sesión

	Media		Mediana		STD	
Run	**Shelly**	**Hwmon**	**Shelly**	**Hwmon**	**Shelly**	**Hwmon**
01	13.37	0.033	11.60	0.023	4.63	0.023
02	13.35	0.035	11.50	0.023	4.98	0.028
03	13.13	0.036	11.50	0.023	4.41	0.027
04	13.15	0.035	11.55	0.023	4.68	0.027
05	13.20	0.036	11.70	0.023	4.46	0.032
06	14.54	0.040	11.90	0.023	6.27	0.031
07	13.07	0.034	11.25	0.023	4.93	0.025
08	15.13	0.043	11.70	0.024	6.91	0.032
09	12.89	0.036	11.10	0.023	4.86	0.025
10	18.79	0.053	17.90	0.049	3.91	0.032
11	12.79	0.032	10.50	0.023	4.87	0.019
12	11.90	0.032	10.40	0.023	4.28	0.023
13	12.02	0.037	10.40	0.023	4.55	0.028
14	12.46	0.035	10.60	0.023	5.01	0.030
15	12.13	0.038	10.50	0.023	4.61	0.032
16	11.88	0.038	10.40	0.023	4.32	0.029
17	12.08	0.039	10.60	0.023	4.32	0.031
18	13.48	0.041	10.65	0.023	5.51	0.032
19	12.46	0.041	10.70	0.023	4.50	0.033
20	11.86	0.034	10.50	0.023	4.45	0.030
21	12.31	0.036	10.90	0.023	4.49	0.027
22	12.16	0.038	10.35	0.023	4.71	0.030
23	12.25	0.036	10.60	0.023	4.51	0.030
24	11.87	0.037	10.40	0.023	4.27	0.034
25	12.35	0.036	10.50	0.023	4.73	0.026
26	12.42	0.035	10.60	0.023	4.80	0.031

Continúa en la siguiente página

Tabla B.2 – Continuación de la página anterior

Run	Media		Mediana		STD	
	Shelly	**Hwmon**	**Shelly**	**Hwmon**	**Shelly**	**Hwmon**
27	12.63	0.035	10.75	0.023	4.67	0.029
28	12.51	0.034	10.90	0.023	4.69	0.026
29	12.38	0.034	10.60	0.023	5.09	0.026
30	12.27	0.038	10.70	0.023	4.22	0.031
31	12.45	0.037	10.70	0.023	4.71	0.029
32	12.23	0.035	10.70	0.023	4.25	0.028
33	12.11	0.038	10.60	0.023	4.45	0.029
34	12.61	0.035	10.90	0.023	4.68	0.027
35	12.61	0.034	10.90	0.023	4.85	0.023
36	12.52	0.034	10.90	0.023	4.65	0.024
37	12.46	0.036	10.70	0.023	4.74	0.031
38	12.47	0.034	10.90	0.023	4.54	0.023
39	12.49	0.037	11.00	0.023	4.23	0.031
40	12.97	0.036	10.90	0.023	5.29	0.032
41	12.81	0.036	10.90	0.023	4.82	0.026
42	12.67	0.037	11.00	0.023	4.84	0.032
43	12.36	0.034	10.80	0.023	4.51	0.028
44	12.63	0.037	10.80	0.023	4.79	0.029
45	12.45	0.034	10.80	0.023	4.52	0.026
46	12.81	0.035	11.00	0.023	4.84	0.025
47	12.59	0.039	10.80	0.023	4.58	0.029
48	13.41	0.038	11.50	0.023	4.80	0.029
49	12.72	0.036	10.80	0.023	4.93	0.029
50	12.49	0.036	10.60	0.023	4.95	0.028
Overall	**12.80**	**0.037**	**11.00**	**0.023**	**4.87**	**0.029**

Tabla B.3: Ranking de webs más visitadas en el mundo y sus enlaces.

Nombre	**Enlace**
Google	https://www.google.com
YouTube	https://www.youtube.com
Facebook	https://www.facebook.com
Instagram	https://www.instagram.com
Baidu	https://www.baidu.com
Wikipedia	https://www.wikipedia.org
Twitter	https://www.twitter.com
Yahoo!	https://www.yahoo.com
Yandex	https://www.yandex.ru
WhatsApp	https://www.whatsapp.com
X	https://www.x.com
XVideos	https://www.xvideos.com
ChatGPT	https://www.chatgpt.com
TikTok	https://www.tiktok.com
Reddit	https://www.reddit.com
Amazon	https://www.amazon.com
Pornhub	https://www.pornhub.com
Yahoo! Japan	https://www.yahoo.co.jp
Docomo	https://www.docomo.ne.jp
Microsoft Outlook	https://www.live.com
LinkedIn	https://www.linkedin.com
XNXX	https://www.xnxx.com
Netflix	https://www.netflix.com
Microsoft 365	https://www.office.com
xHamster	https://www.xhamster.com
Microsoft Bing	https://www.bing.com
Zen News	https://www.dzen.ru

Continúa en la siguiente página

Tabla B.3 – Continuación de la página anterior

Nombre	**Enlace**
Microsoft Online	https://www.microsoftonline.com
Pinterest	https://www.pinterest.com
Naver	https://www.naver.com
Turbo Pages	https://www.turbopages.org
Bilibili	https://www.bilibili.com
VK	https://www.vk.com
Mail.ru	https://www.mail.ru
Samsung Electronics	https://www.samsung.com
Discord	https://www.discord.com
Max	https://www.max.com
Microsoft	https://www.microsoft.com
xHamster (desi)	https://xhamster.desi
The Weather Channel	https://www.weather.com
Twitch	https://www.twitch.tv
Telegram	https://t.me
Quora	https://www.quora.com
SharePoint	https://www.sharepoint.com
Fandom	https://www.fandom.com
OpenAI	https://www.openai.com
Stripchat	https://www.stripchat.com
DuckDuckGo	https://www.duckduckgo.com
Zoom	https://www.zoom.us

Tabla B.4: Consumo de Energía por Sesión (Valores corregidos)

Sesión	**Energía Total (J)**	**Energía Total (Wh)**	**Energía Total (kWh)**
Amazon	146.57	0.04071	0.00004071
Baidu	255.57	0.07099	0.00007099
Bilibili	1740.97	0.48360	0.00048360
ChatGPT	60.17	0.01671	0.00001671
Discord	6359.77	1.76660	0.00176660
Docomo	486.67	0.13519	0.00013519
DuckDuckGo	28.87	0.00802	0.00000802
Facebook	319.29	0.08869	0.00008869
Fandom	278.67	0.07741	0.00007741
Google	75.47	0.02096	0.00002096
Instagram	450.17	0.12505	0.00012505
LinkedIn	154.07	0.04280	0.00004280
Mail.ru	5036.59	1.39905	0.00139905
Max	258.27	0.07174	0.00007174
Microsoft	1624.83	0.45134	0.00045134
Microsoft 365	104.45	0.02901	0.00002901
Microsoft Bing	218.45	0.06068	0.00006068
Microsoft Online	-35.23	-0.00979	-0.00000979
Microsoft Outlook	202.27	0.05619	0.00005619
Naver	795.27	0.22091	0.00022091
Netflix	597.15	0.16588	0.00016588
OpenAI	6583.27	1.82869	0.00182869
Pinterest	1425.97	0.39610	0.00039610
Pornhub	83.27	0.02313	0.00002313
Quora	52.25	0.01451	0.00001451
Reddit	474.59	0.13183	0.00013183
Samsung Electronics	2011.47	0.55874	0.00055874

Continúa en la siguiente página

Tabla B.4 – Continuación de la página anterior

Sesión	**Energía Total (J)**	**Energía Total (Wh)**	**Energía Total (kWh)**
SharePoint	-31.33	-0.00870	-0.00000870
Stripchat	342.47	0.09513	0.00009513
Telegram	105.97	0.02944	0.00002944
The Weather Channel	127.97	0.03555	0.00003555
TikTok	1287.17	0.35755	0.00035755
Turbo Pages	6.97	0.00194	0.00000194
Twitch	2875.53	0.79876	0.00079876
VK	219.39	0.06094	0.00006094
WhatsApp	20.07	0.00558	0.00000558
Wikipedia	-43.81	-0.01217	-0.00001217
X	304.89	0.08469	0.00008469
XNXX	53.49	0.01486	0.00001486
XVideos	-0.25	-0.00007	-0.00000007
Yahoo!	338.77	0.09410	0.00009410
Yahoo! Japan	-43.83	-0.01217	-0.00001217
Yandex	897.63	0.24934	0.00024934
YouTube	260.57	0.07238	0.00007238
Zen News	814.57	0.22627	0.00022627
Zoom	530.47	0.14735	0.00014735
xHamster	48.27	0.01341	0.00001341
xHamster (desi)	153.27	0.04258	0.00004258

Tabla B.5: Consumo de Energía por Sesión

Sesión	**Energía Total (J)**	**Energía Total (Wh)**	**Energía Total (kWh)**
Amazon	2667.90	0.74108	0.00074108
Baidu	2776.90	0.77136	0.00077136
Bilibili	4262.30	1.18397	0.00118397
ChatGPT	2581.50	0.71708	0.00071708
Discord	8881.10	2.46697	0.00246697
Docomo	3008.00	0.83556	0.00083556
DuckDuckGo	2550.20	0.70839	0.00070839
Facebook	2830.20	0.78617	0.00078617
Fandom	2800.00	0.77778	0.00077778
Google	2596.80	0.72133	0.00072133
Instagram	2971.50	0.82542	0.00082542
LinkedIn	2675.40	0.74317	0.00074317
Mail.ru	7547.50	2.09653	0.00209653
Max	2779.60	0.77211	0.00077211
Microsoft	4167.00	1.15750	0.00115750
Microsoft 365	2636.20	0.73228	0.00073228
Microsoft Bing	2750.20	0.76394	0.00076394
Microsoft Online	2486.10	0.69058	0.00069058
Microsoft Outlook	2723.60	0.75656	0.00075656
Naver	3316.60	0.92128	0.00092128
Netflix	3128.90	0.86914	0.00086914
OpenAI	9104.60	2.52906	0.00252906
Pinterest	3947.30	1.09647	0.00109647
Pornhub	2604.60	0.72350	0.00072350
Quora	2584.00	0.71778	0.00071778
Reddit	2985.50	0.82931	0.00082931
Samsung Electronics	4532.80	1.25911	0.00125911

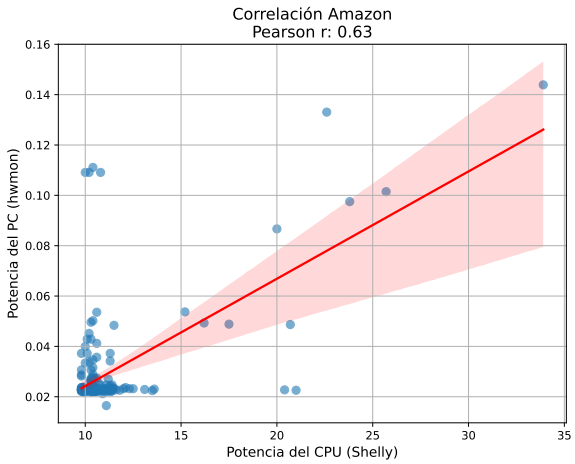
Continúa en la página siguiente

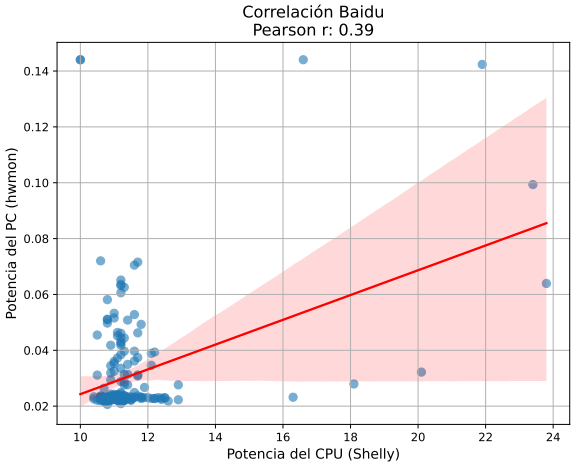
Tabla B.5 – Continuación de la página anterior

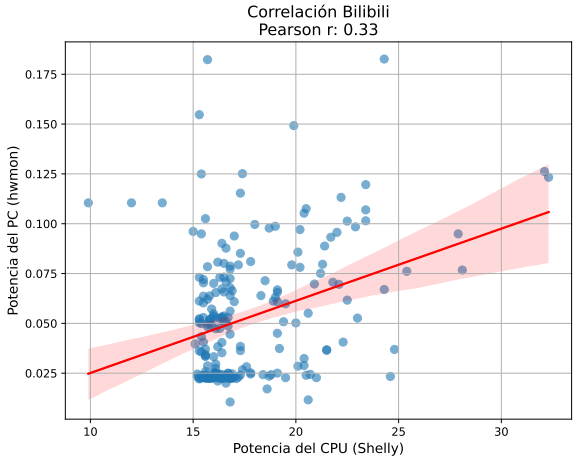
Sesión	**Energía Total (J)**	**Energía Total (Wh)**	**Energía Total (kWh)**
SharePoint	2490.00	0.69167	0.00069167
Stripchat	2863.80	0.79550	0.00079550
Telegram	2627.30	0.72981	0.00072981
The Weather Channel	2649.30	0.73592	0.00073592
TikTok	3808.50	1.05792	0.00105792
Turbo Pages	2528.30	0.70231	0.00070231
Twitch	5417.70	1.50492	0.00150492
VK	2730.30	0.75842	0.00075842
WhatsApp	2541.40	0.70594	0.00070594
Wikipedia	2467.10	0.68531	0.00068531
X	2815.80	0.78217	0.00078217
XNXX	2564.40	0.71233	0.00071233
XVideos	2531.50	0.70319	0.00070319
Yahoo!	2860.10	0.79447	0.00079447
Yahoo! Japan	2477.50	0.68819	0.00068819
Yandex	3439.80	0.95550	0.00095550
YouTube	2781.90	0.77275	0.00077275
Zen News	3335.90	0.92664	0.00092664
Zoom	3051.80	0.84772	0.00084772
xHamster	2569.60	0.71378	0.00071378
xHamster (desi)	2674.60	0.74294	0.00074294

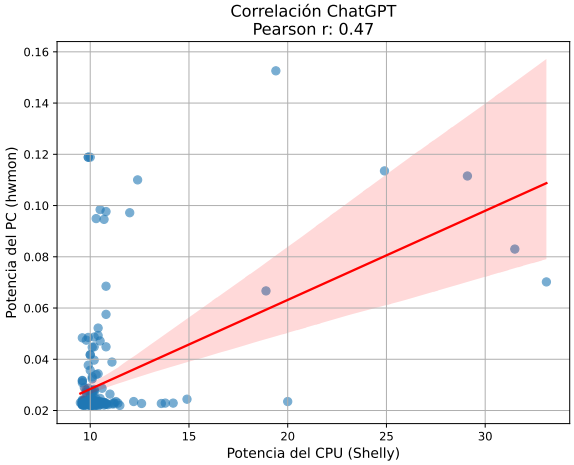
Anexo C

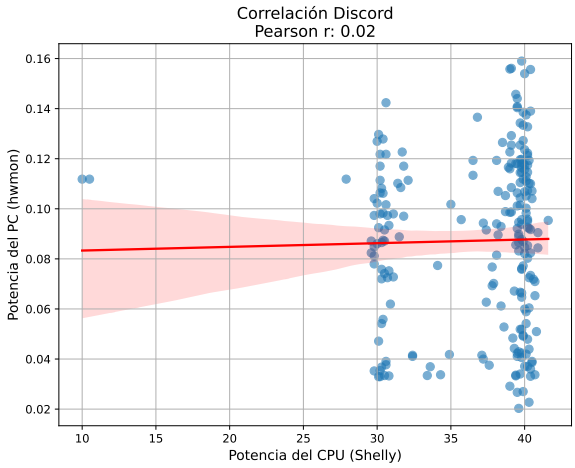
Gráficas y tablas complementarias

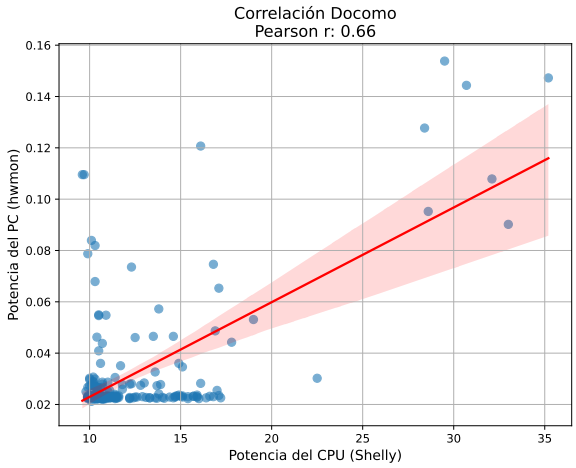


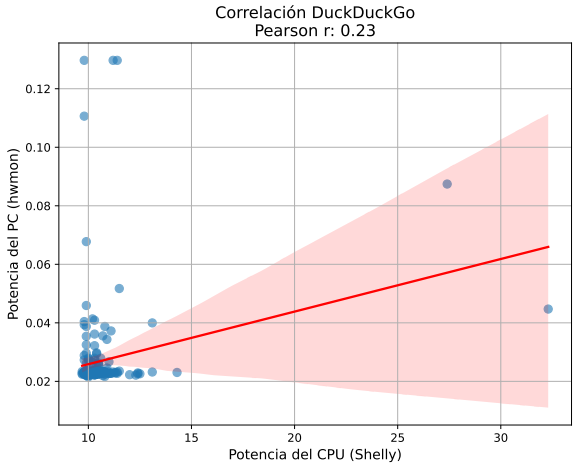


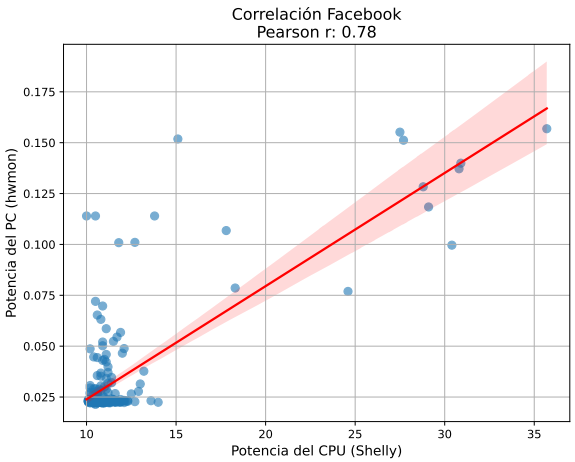


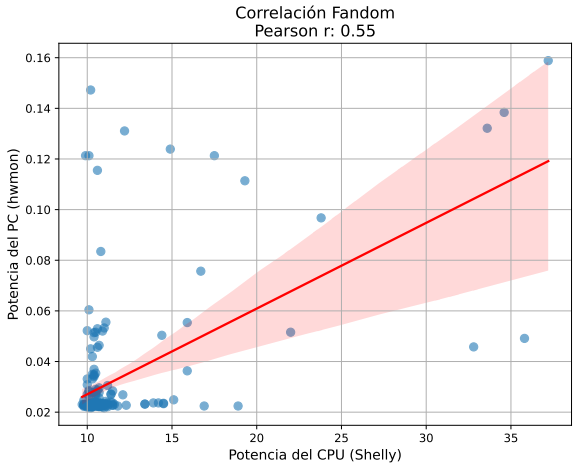


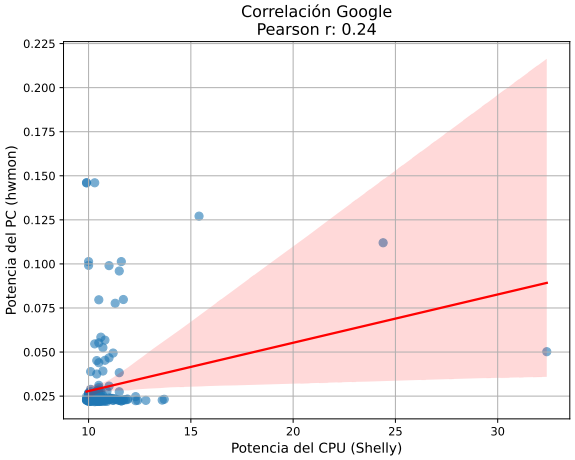

Correlación Discord
Pearson r: 0.02
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

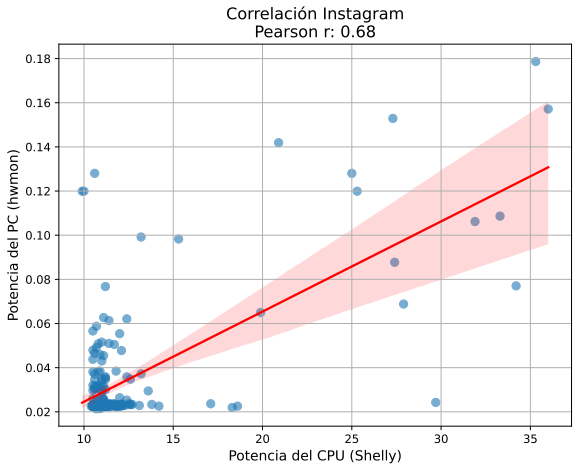

Correlación Docomo
Pearson r: 0.66
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

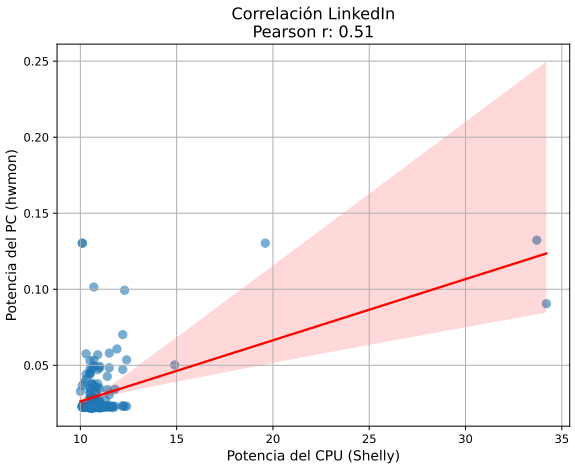

Correlación DuckDuckGo
Pearson r: 0.23
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

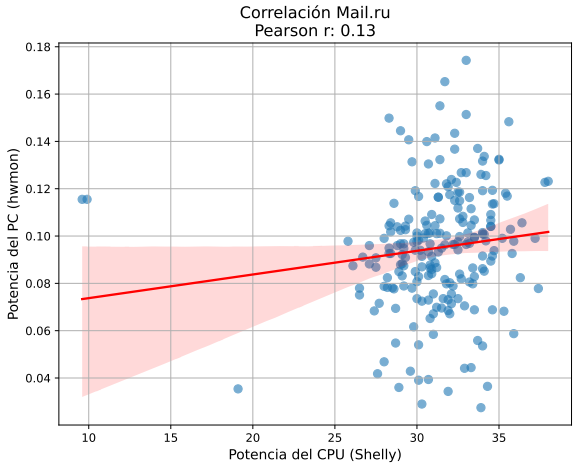

Correlación Facebook
Pearson r: 0.78
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

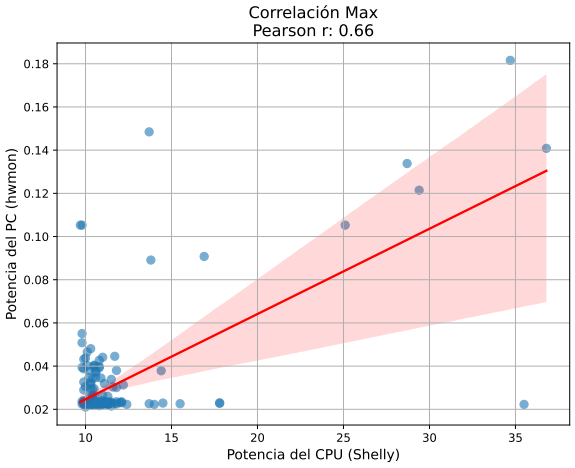

Correlación Fandom
Pearson r: 0.55
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

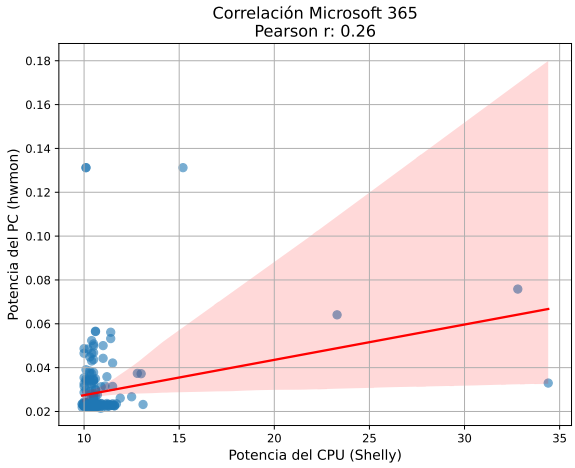

Correlación Google
Pearson r: 0.24
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

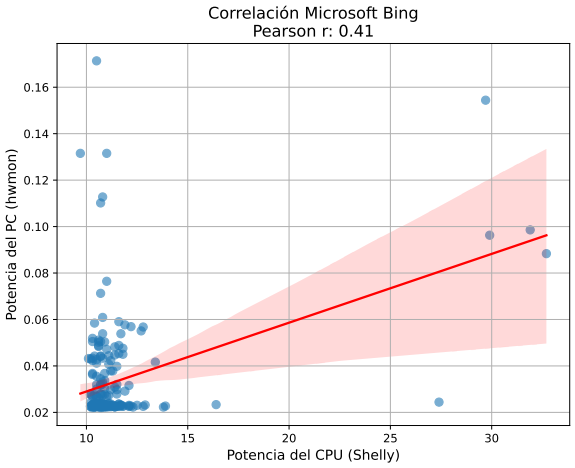

Correlación Instagram
Pearson r: 0.68
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

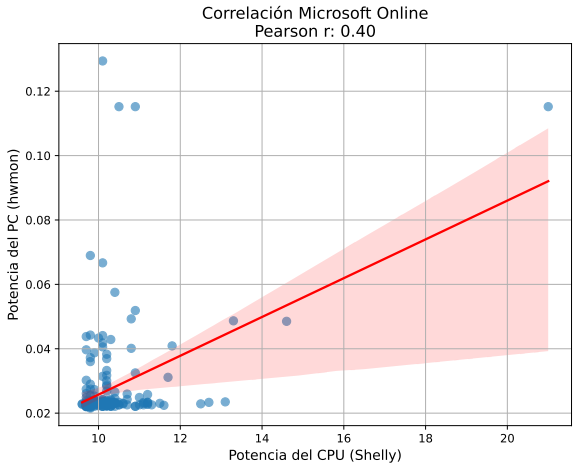

Correlación LinkedIn
Pearson r: 0.51
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

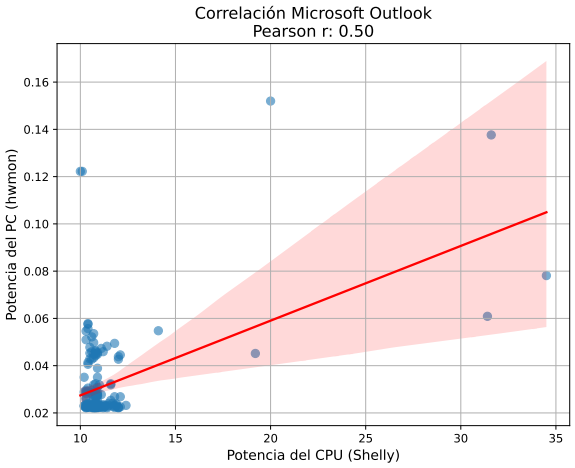

Correlación Mail.ru
Pearson r: 0.13
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

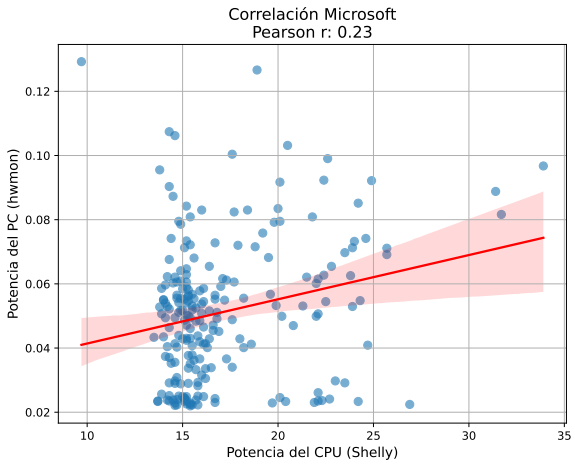

Correlación Max
Pearson r: 0.66
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

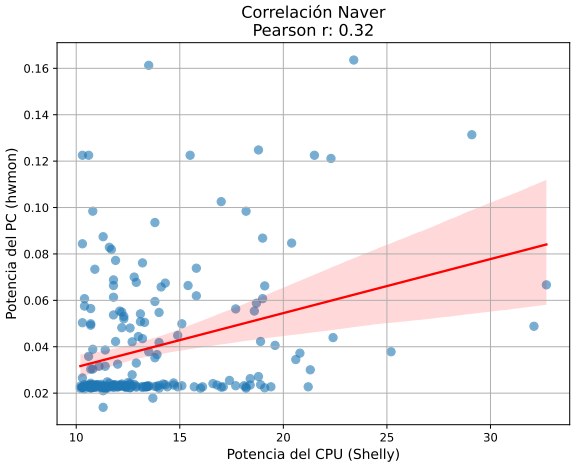

Correlación Microsoft 365
Pearson r: 0.26
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

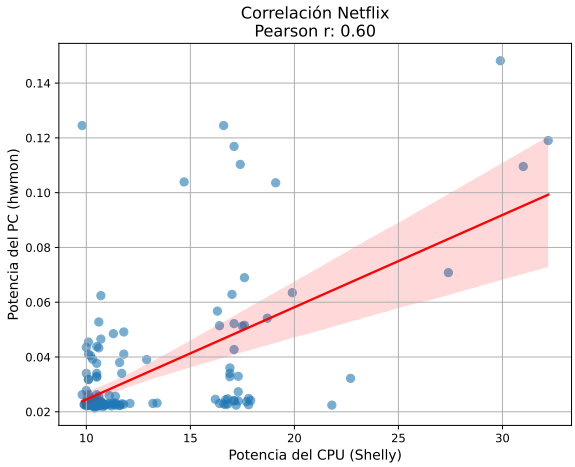

Correlación Microsoft Bing
Pearson r: 0.41
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

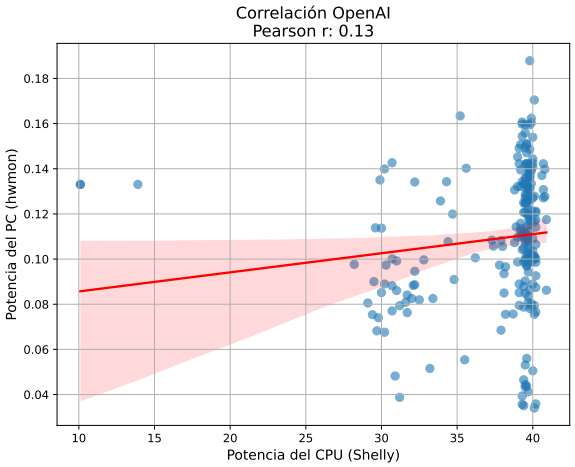

Correlación Microsoft Online
Pearson r: 0.40
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

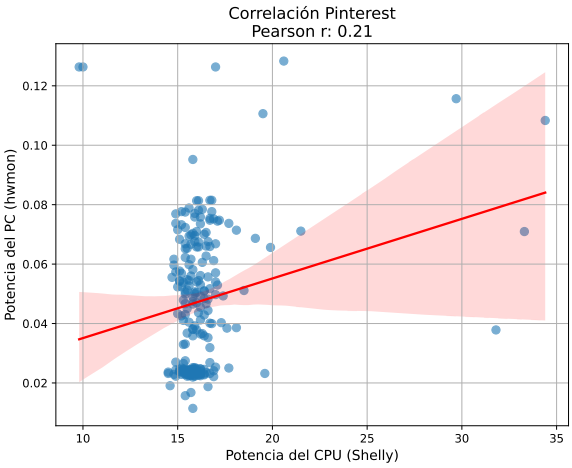

Correlación Microsoft Outlook
Pearson r: 0.50
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

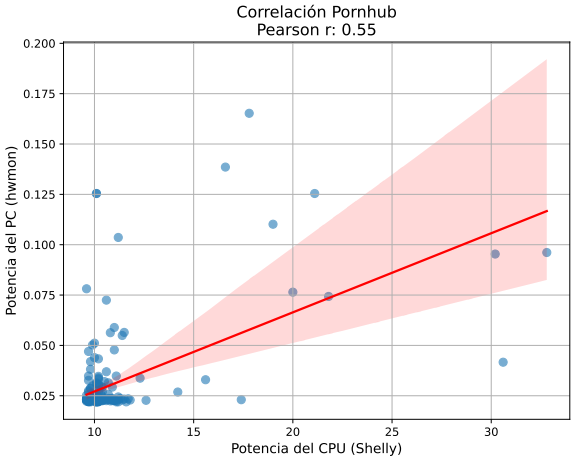

Correlación Microsoft
Pearson r: 0.23
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

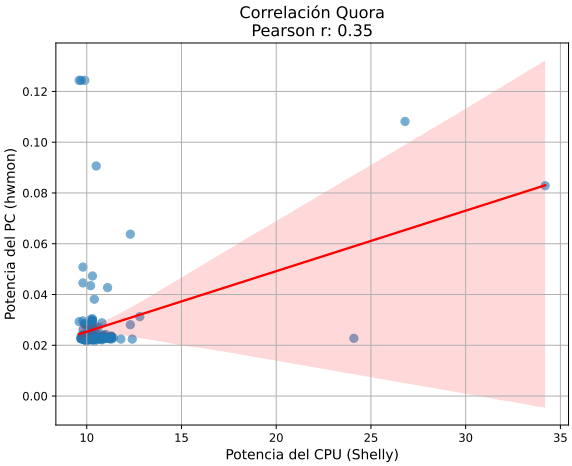

Correlación Naver
Pearson r: 0.32
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

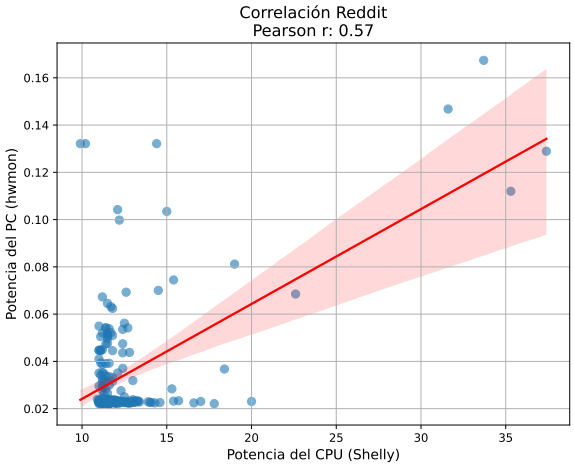

Correlación Netflix
Pearson r: 0.60
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

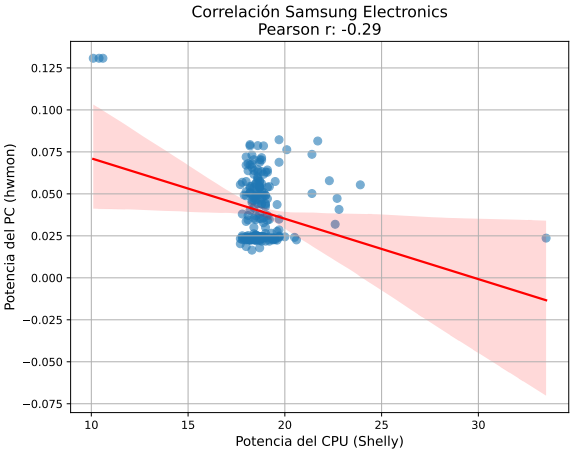

Correlación OpenAI
Pearson r: 0.13
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

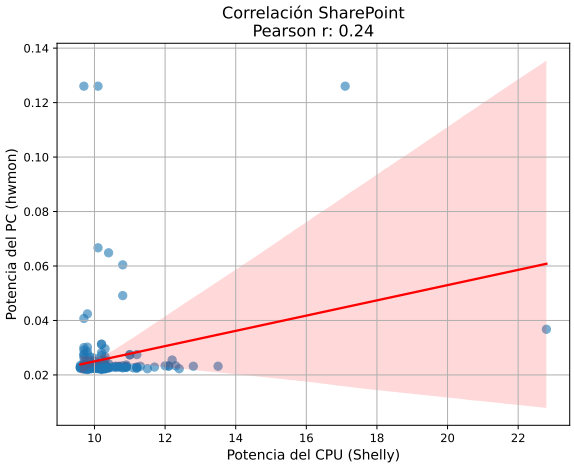

Correlación Pinterest
Pearson r: 0.21
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

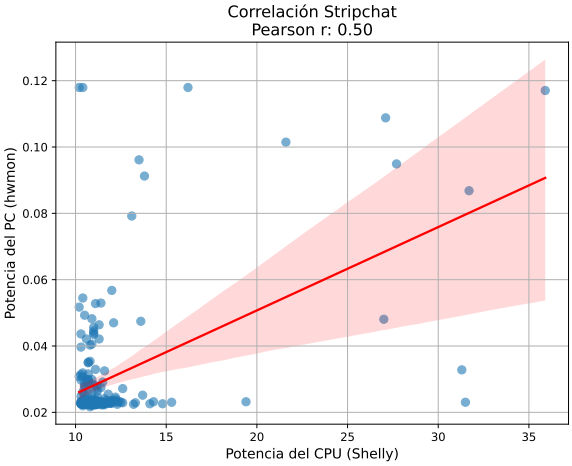

Correlación Pornhub
Pearson r: 0.55
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

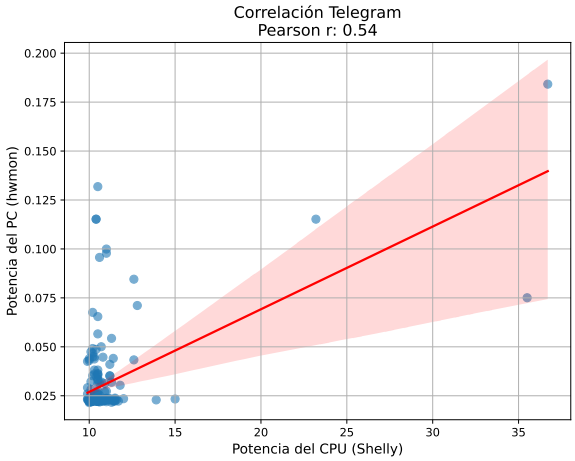

Correlación Quora
Pearson r: 0.35
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

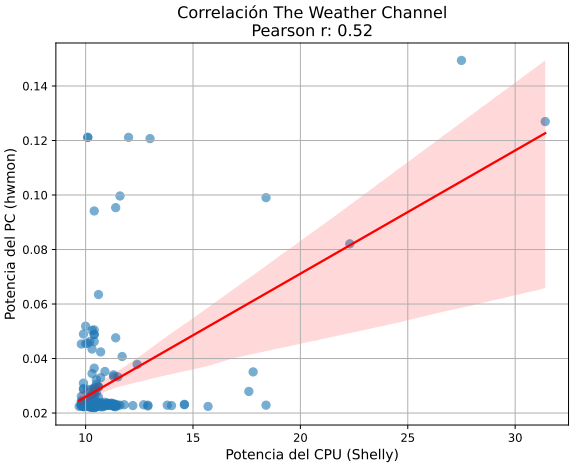

Correlación Reddit
Pearson r: 0.57
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

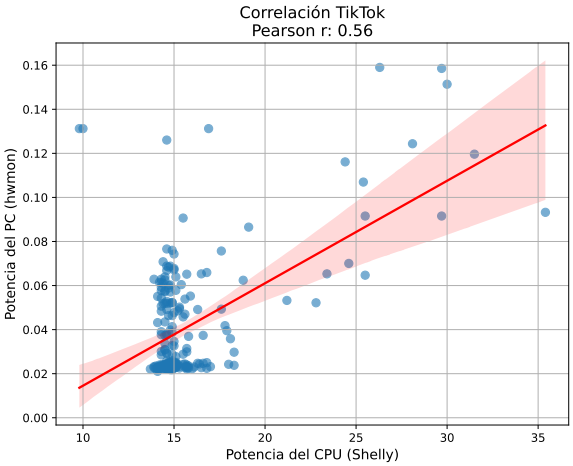

Correlación Samsung Electronics
Pearson r: -0.29
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

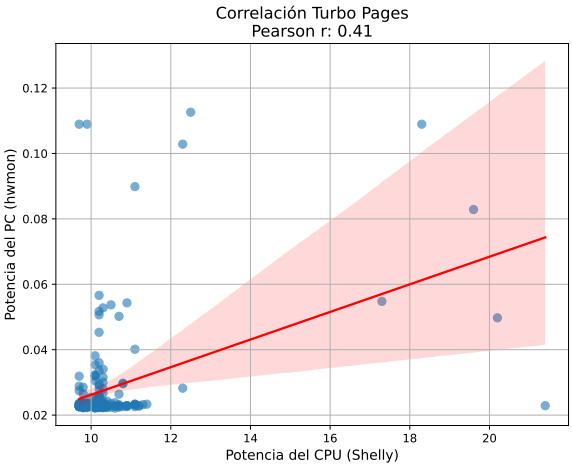

Correlación SharePoint
Pearson r: 0.24
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

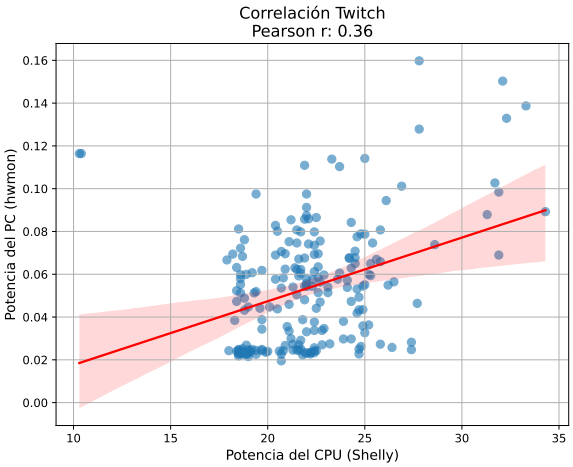

Correlación Stripchat
Pearson r: 0.50
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

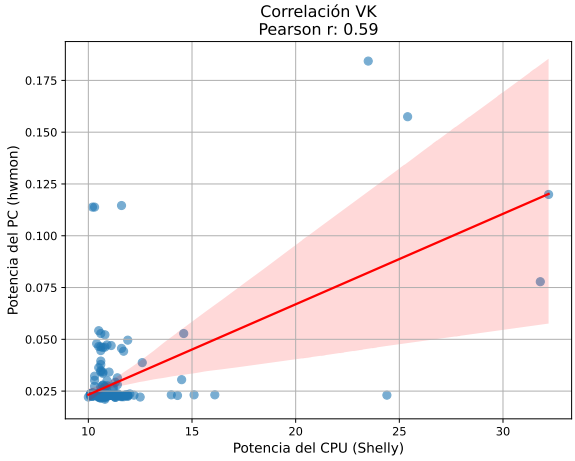

Correlación Telegram
Pearson r: 0.54
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

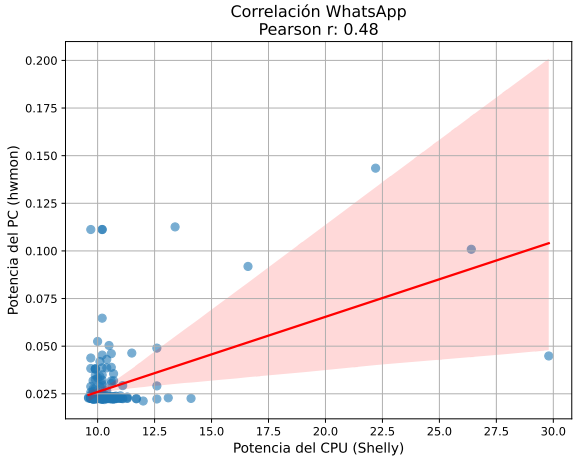

Correlación The Weather Channel
Pearson r: 0.52
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

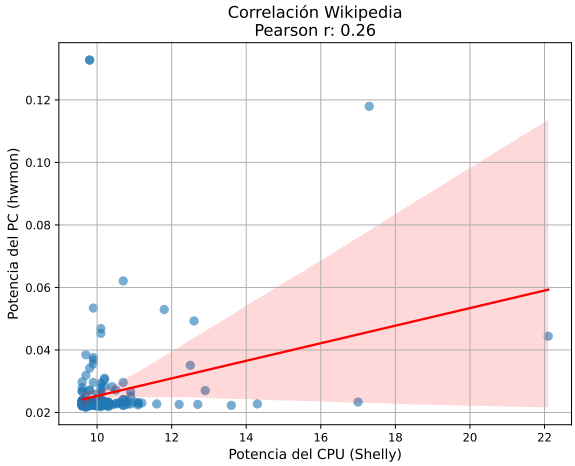

Correlación TikTok
Pearson r: 0.56
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

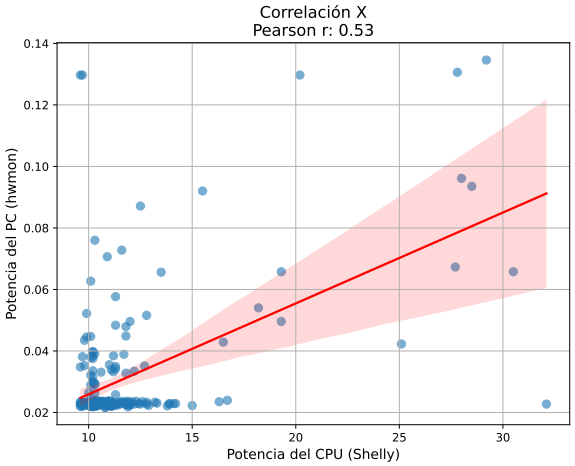

Correlación Turbo Pages
Pearson r: 0.41
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

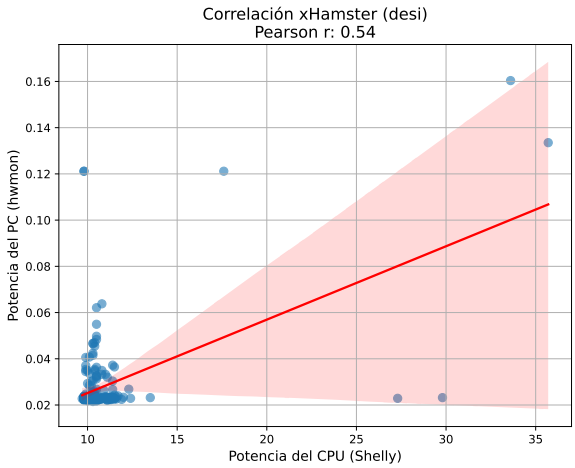

Correlación Twitch
Pearson r: 0.36
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

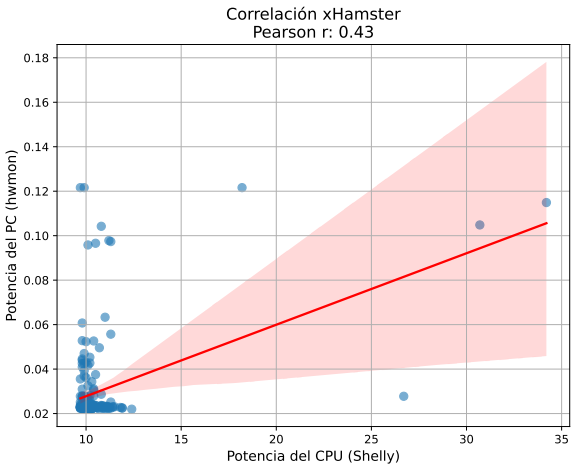

Correlación VK
Pearson r: 0.59
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

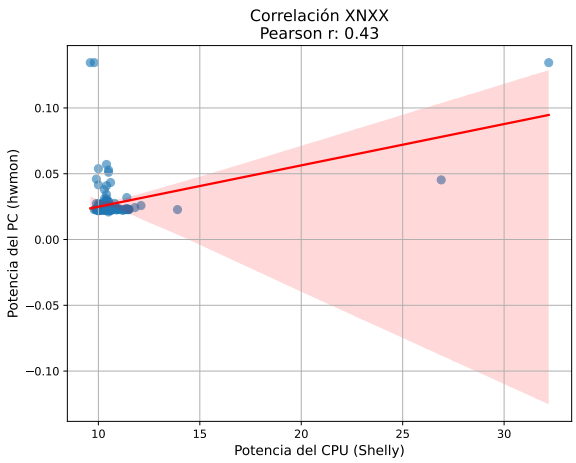

Correlación WhatsApp
Pearson r: 0.48
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

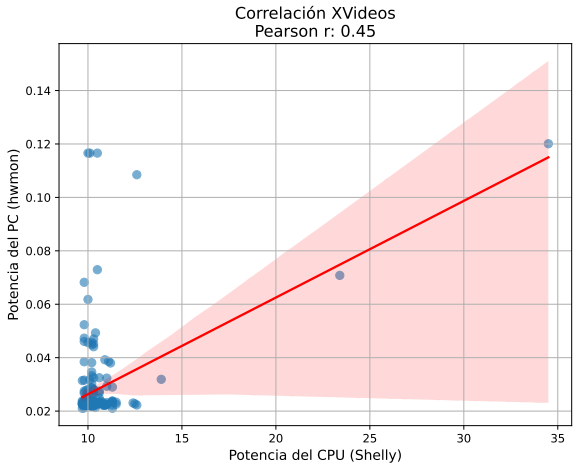

Correlación Wikipedia
Pearson r: 0.26
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

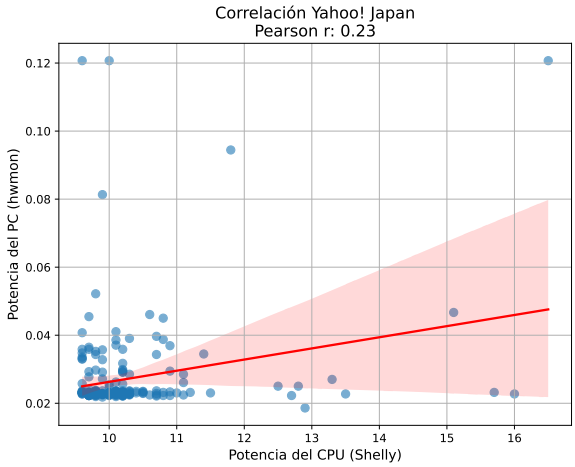

Correlación X
Pearson r: 0.53
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

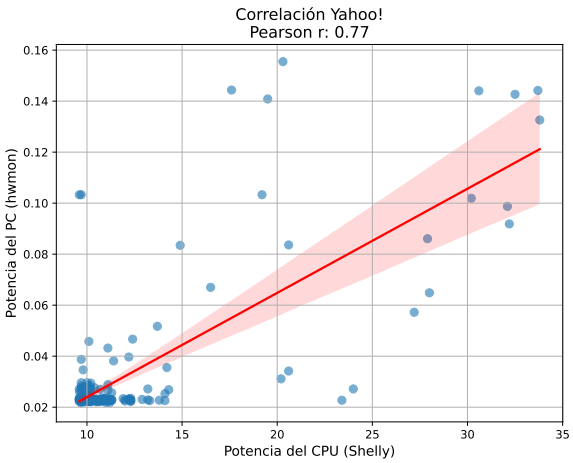

Correlación xHamster (desi)
Pearson r: 0.54
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

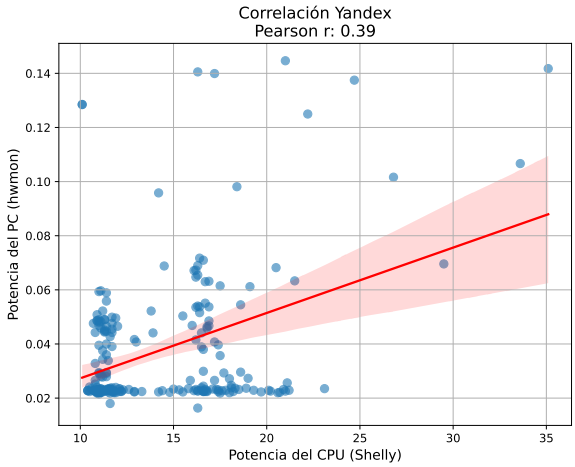

Correlación xHamster
Pearson r: 0.43
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

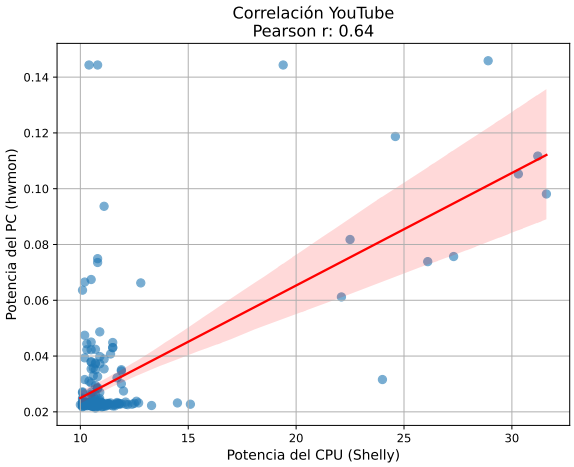

Correlación XNXX
Pearson r: 0.43
Potencia del PC (hwmon)
Potencia del CPU (Shelly)


Correlación XVideos
Pearson r: 0.45
Potencia del PC (hwmon)
Potencia del CPU (Shelly)


Correlación Yahoo! Japan
Pearson r: 0.23
Potencia del PC (hwmon)
Potencia del CPU (Shelly)


Correlación Yahoo!
Pearson r: 0.77
Potencia del PC (hwmon)
Potencia del CPU (Shelly)


Correlación Yandex
Pearson r: 0.39
Potencia del PC (hwmon)
Potencia del CPU (Shelly)


Correlación YouTube
Pearson r: 0.64
Potencia del PC (hwmon)
Potencia del CPU (Shelly)

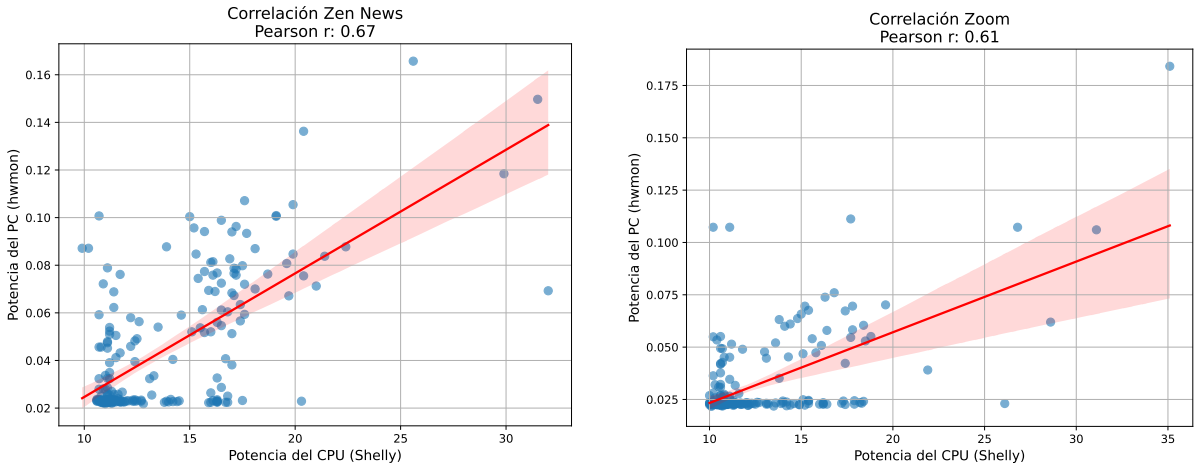


Figura C.1: Gráficas de correlación de uso de CPU vs. consumo total para cada sitio web

Bibliografía

[1] Allterco Robotics. Shelly plug s. https://shelly-api-docs.shelly.cloud/gen2/Devices/Gen3/ShellyPlugSG3/.

[2] G. Fettweis and E. Zimmermann. Ict energy consumption - trends and challenges. In *Proceedings of the 11th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, pages 1–4, 2008.

[3] GitLab Inc. Gitlab. https://about.gitlab.com/free-trial/devsecops/?utm_medium=cpc&utm_source=google&utm_campaign=brand_rlsa__global_exact&utm_content=free-trial&utm_term=gitlab&_bt=656315922370&_bk=gitlab&_bm=e&_bn=g&_bg=148481441276&gad_source=1&gad_campaignid=20029282011&gbraid=0AAAADcJCbde1h4eIQ08P7CilbZXc8fwK&gclid=CjwKCAjwyb3DBhBlEiwAqZLe5ONajtMmrchnjI1IxB8Yu-xpAuHP7G4GM7MXxzlNvKbtD0BwE.

[4] JetBrains s.r.o. Pycharm. el único ide de python que necesita. https://www.jetbrains.com/es-es/pycharm/#.

[5] LaTeX Project. Latex - a document preparation system. https://www.latex-project.org/.

[6] Linux Kernel Documentation. Fam15h power driver. https://docs.kernel.org/hwmon/fam15h_power.html.

[7] Linux Kernel Documentation. Hwmon kernel api. https://docs.kernel.org/hwmon/hwmon-kernel-api.html.

[8] Linux Kernel Documentation. Sysfs interface for hardware monitoring. https://docs.kernel.org/hwmon/sysfs-interface.html.

[9] Michael Waskom. Seaborn: Statistical data visualization. https://seaborn.pydata.org/.

[10] NumFOCUS, Inc. Pandas. https://pandas.pydata.org/.

[11] A. Pérez, J. García, and S. López. Energy-aware analysis framework for linux-based systems. *Journal of Systems and Software*, 150:437–449, 2019.

[12] Python Software Foundation. Python. https://www.python.org/.

[13] Python Software Foundation. Requests. https://pypi.org/project/requests/.

[14] Python Software Foundation. Subprocess management. https://docs.python.org/3/library/subprocess.html.

[15] Software Freedom Conservancy. Git. https://git-scm.com/.

[16] Y. Tene, A. Cohen, and Y. Naveh. Energy-efficient web design: Minimizing power consumption on mobile devices. *IEEE Access*, 9:89054–89066, 2021.

[17] The Matplotlib development team. Matplotlib: Visualization with python. https://matplotlib.org/.

[18] The SciPy community. Pearsonr. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pearsonr.html.

[19] Wholegrain Digital. Website carbon calculator. https://www.websitecarbon.com, 2023. Último acceso: mayo de 2025.

[20] Wikipedia. Anexo:sitios web más visitados. https://es.wikipedia.org/wiki/Anexo:Sitios_web_m%C3%A1s_visitados, 2025.

[21] A. Zaidman. An inconvenient truth in software engineering? the environmental impact of testing open source java projects. In *Proceedings of the 2024 IEEE/ACM International Conference on Automation of Software Test (AST)*, pages 214–218. ACM, 2024.