

Christian Aarón Campos Lucas e Ignacio Trejos Zelaya
Universidad CENFOTEC

Las computadoras son máquinas **programables**: ellas obedecen órdenes, que son instrucciones escritas en un lenguaje (o código) que ellas pueden interpretar. Lo que hagan las computadoras variará según las secuencias de instrucciones que estén ejecutando – eso es lo que llamamos **programas**. Todos los días interactuamos con artefactos que tienen computadoras invisibles (microprocesadores) cuyo proceder es logrado vía software, por ejemplo: teléfonos celulares, consolas de videojuegos, hornos de microondas, dispositivos en redes de comunicación, cajeros automáticos, sistemas de control de frenos en vehículos, robots en fábricas, pilotos automáticos en aviones, equipos de dosificación de medicamentos, máquinas de radioterapia en hospitales, y un largo etcétera.

Además de programas, el software abarca datos, modelos, documentación y procedimientos operativos mediante los cuales los sistemas basados en computadoras y microprocesadores son útiles para las personas, organizaciones, sociedades y el entorno. El software logra que los sistemas computacionales procesen, almacenen, recuperen, organicen y comuniquen los datos de nuestro interés. Las computadoras consumen energía y los dispositivos informáticos se construyen con componentes materiales que provienen de actividades extractivas (minería, perforaciones petroleras) y procesos de manufactura.

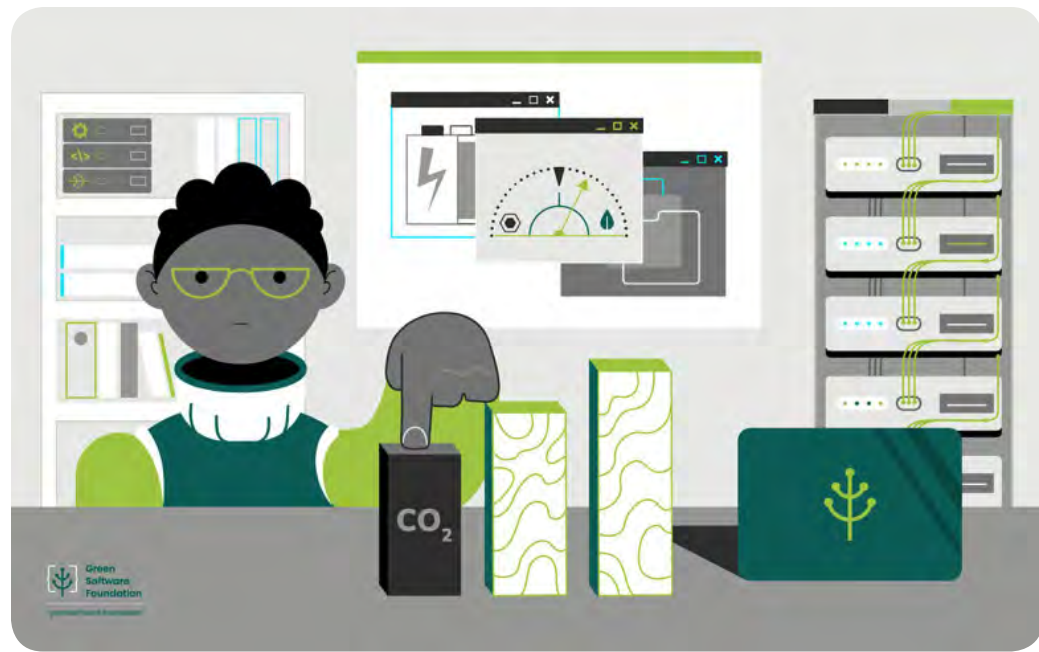
Los sistemas informáticos y de telecomunicaciones, y el software que los anima, impactan el ambiente, y lo hacen de diversas maneras.

La **Ingeniería del Software Verde** (en inglés, 'Green Software Engineering'), se enfoca en desarrollar y mantener software de manera que minimice el impacto nocivo sobre el ambiente.

La Ingeniería del software va más allá de la programación amateur. Los ingenieros de software escriben código para computadoras, pero esta es solo una faceta de su labor. Un programador no profesional se enfoca en crear programas para desarrollar aplicaciones de software, sin prestar mucha atención a la calidad, los costos – *financieros e incluso energéticos* – y los plazos.

El software es crecientemente importante para cada vez más actividades humanas. Muchos productos de software son extensos, complejos y críticos. Ingenieros e ingenieras de software trabajan de manera metódica para asegurar la calidad de sus resultados y desarrollan habilidades específicas que los hacen más eficientes en el desarrollo de software que otros profesionales en informática. Estudian teorías, modelos, técnicas, métodos y tecnologías para el desarrollo profesional del software, lo cual abarca todos los aspectos de su producción, así como los desafíos prácticos de desarrollarlo en equipos colaborativos y de implementar los productos para su uso. Los ingenieros y las ingenieras de software aprenden a crear software de alta calidad, de manera oportuna y económica, adoptando enfoques sistemáticos y organizados de la Ingeniería para realizar su trabajo, empleando técnicas y herramientas adecuadas según las necesidades del problema por resolver, las restricciones del desarrollo y los recursos disponibles.

En un artículo anterior decíamos que algo es **sostenible** cuando puede mantenerse o perdurar sin agotar los recursos necesarios para su existencia. La sostenibilidad consiste en satisfacer las necesida-



Ingeniería del Software Verde

des del presente sin poner en peligro la capacidad de las futuras generaciones para satisfacer las suyas. Este concepto se fundamenta en el equilibrio entre los aspectos económicos, sociales y ambientales de las actividades humanas.

Los estudiantes de Ingeniería del Software deben desarrollar consciencia y criterio propio sobre lo que es 'bueno', 'correcto', 'aceptable' o 'responsable' en su futura práctica profesional. Los profesionales en tecnología deben entender los impactos sociales y ambientales de su trabajo.

La informática sostenible, también llamada "tecnología verde", "informática verde" o "computación sostenible", se refiere a la práctica de diseñar, fabricar, usar y gestionar sistemas informáticos de forma que se minimice su impacto ambiental. Un objetivo principal es disminuir la huella ecológica de las tecnologías digitales. La informática sostenible abarca varias áreas clave: eficiencia energética, reciclaje de componentes, diseño sostenible, centros de datos eficientes, computación en la nube 'verde', desarrollo de software sostenible, educación y consciencia.

La informática sostenible intenta equilibrar el progreso tecnológico y económico con la responsabilidad ambiental y social, garantizando que la tecnología se desarrolle y utilice de forma que reduzca su impacto negativo sobre la sociedad y el ambiente, fomentando un futuro más sostenible.

En el resto de este artículo nos enfocaremos en la Ingeniería del Software con consideraciones de sostenibilidad ambiental.

Conceptos clave en Ingeniería del Software Verde

La **eficiencia energética del software** es un aspecto central de la informática sostenible. Se puede lograr vía dos enfoques principales: la optimización de código y la gestión de recursos. La **optimización de código** implica escribir programas que consuman menos recursos computacionales, lo que reduce el consumo de energía. Esto se logra evitando operaciones innecesarias y utilizando algoritmos más eficientes. Por otro lado, la **gestión de re-**



Minimizar el carbono

curso asegura que los componentes del sistema, como la memoria y el procesador, se utilicen de manera óptima, liberando recursos cuando ya no sean necesarios. Ambas prácticas son esenciales para desarrollar software que sea no solo efectivo, sino también eficiente y respetuoso con el ambiente.

La **arquitectura de software sostenible** se centra en dos enfoques principales: el diseño modular y el uso de virtualización y contenedores. El **diseño modular** conlleva crear software que se pueda actualizar y mejorar sin rehacer grandes partes del sistema, lo que facilita el mantenimiento y contribuye a la sostenibilidad. Por otro lado, la implementación de **tecnologías de virtualización y contenedores** optimiza la utilización de los recursos de servidores y reduce el consumo de energía, haciendo que los sistemas sean más eficientes y respetuosos con el ambiente.

La **gestión del ciclo de vida del software** abarca desde el desarrollo, su posterior mantenimiento y evolución, hasta su finalización (o retiro), asegurando prácticas sostenibles en cada etapa. Durante el **desarrollo, el mantenimiento y la evolución**, se implementan metodologías

ágiles y de mejora continua para promover la eficiencia y reducir el desperdicio. Al planificar el fin de vida del software, se busca minimizar los residuos electrónicos y facilitar el reciclaje del hardware, contribuyendo así a una mayor sostenibilidad y responsabilidad ambiental en la industria tecnológica. En consonancia con lo indicado en el párrafo precedente, como parte del mantenimiento y la evolución de los sistemas, puede modificarse estructuralmente los componentes de software para mejorar su modularidad, así como consolidar servidores mediante tecnologías de virtualización y contenedores.

Prácticas de Ingeniería del Software Verde

La **medición** y el **monitoreo** son esenciales para garantizar la eficiencia energética del software. Esto incluye el uso de **herramientas de monitoreo** específicas para medir el consumo de energía durante la ejecución de los programas, abarcando métricas como uso de procesador y de memoria, así como otras estadísticas de rendimiento. Además, se realizan **análisis de eficiencia** periódicos para identificar y corregir ineficiencias energéticas, lo que permite optimizar continuamente el software y reducir su impacto ambiental.

El **desarrollo sostenible** en el ámbito del software se centra en dos aspectos fundamentales. En primer lugar, se realizan **pruebas de eficiencia energética** como parte integral del proceso de desarrollo y pruebas de software, lo que permite evaluar y optimizar el consumo de energía de manera proactiva. Además, se adoptan prácticas de **programación conscientes del clima**, que evitan operaciones intensivas en energía siempre que sea posible, contribuyendo así a reducir el impacto ambiental de las aplicaciones y los sistemas de software.

La **educación y consciencia** tienen un papel decisivo en la promoción de la informática sostenible. Esto incluye la **formación de desarrolladores**, donde se educa a los estudiantes y a los profesionales en servicio sobre la importancia de la eficiencia energética y cómo pueden contribuir a ella mediante buenas prácticas de programación. Además, se **promueve activamente una cultura verde** dentro de las organizaciones, fomentando valores de sostenibilidad e incentivando la innovación en la ingeniería del software verde.

Tecnología y herramientas

Hay **herramientas de análisis y optimización** que pueden ayudarnos a mejorar la eficiencia energética del software. Por un lado, herramientas de 'perfilamiento' ofrecidas por proveedores de procesadores – como Intel VTune y ARM Streamline – permiten analizar detalladamente el consumo de energía y mejorar el rendimiento del software, al ayudar a identificar áreas de mejora y posibles puntos de optimización. Por otro lado, las herramientas de **gestión de energía en tiempo de ejecución** son capaces de ajustar dinámicamente la carga de trabajo y el consumo de energía del sistema en tiempo real, lo que permite una gestión más eficiente y adaptable de los recursos, contribuyendo así a una mayor sostenibilidad en el desarrollo y la ejecución de software.

La elección de **marcos de trabajo y bibliotecas** ('librerías') eficientes es fundamental para desarrollar software con un menor impacto ambiental. Optar por marcos livianos, reconocidos por su eficiencia y bajo consumo de recursos, permite reducir el uso de energía durante el desarrollo y la ejecución del software. Esas herramientas ofrecen funcionalidades necesarias sin agregar una carga excesiva al sistema, lo

que contribuye a una mayor sostenibilidad en el desarrollo de aplicaciones y sistemas de software.

La elección de **lenguajes de programación** también tiene un gran impacto: el software programado en lenguajes de programación que compilan a código nativo de las computadoras (Rust, C#, C++, Swift, Go) es más eficiente que los programas escritos en lenguajes que compilan a máquinas abstractas (Java, Kotlin, Scala), que a su vez son más eficientes en ejecución que los programas escritos en lenguajes interpretados (Python, JavaScript, Ruby).

En el contexto de la informática sostenible que nos motiva, se destacan otras **buenas prácticas** que ayudan a reducir el impacto ambiental y mejorar la eficiencia de los sistemas digitales. En primer lugar, la **optimización de algoritmos** contribuye a reducir la complejidad computacional, lo que lleva a un menor consumo de recursos y energía. La implementación de estructuras de datos eficientes permite minimizar el uso de memoria, optimizando así el rendimiento del sistema. Otra área importante es el **ahorro de energía en dispositivos móviles**, donde se enfatiza la optimización de aplicaciones para reducir el consumo de batería – por ejemplo, reduciendo la frecuencia de actualizaciones en segundo plano y el uso de GPS, lo que aumenta la duración de la batería y reduce la huella ambiental. Asimismo, el **uso eficiente de la red** minimiza el consumo de energía asociado con las transferencias de datos y las comunicaciones de red, lo que también mejora la eficiencia de la red en general. Estas prácticas ejemplifican el compromiso con la sostenibilidad en el diseño y uso de sistemas informáticos.

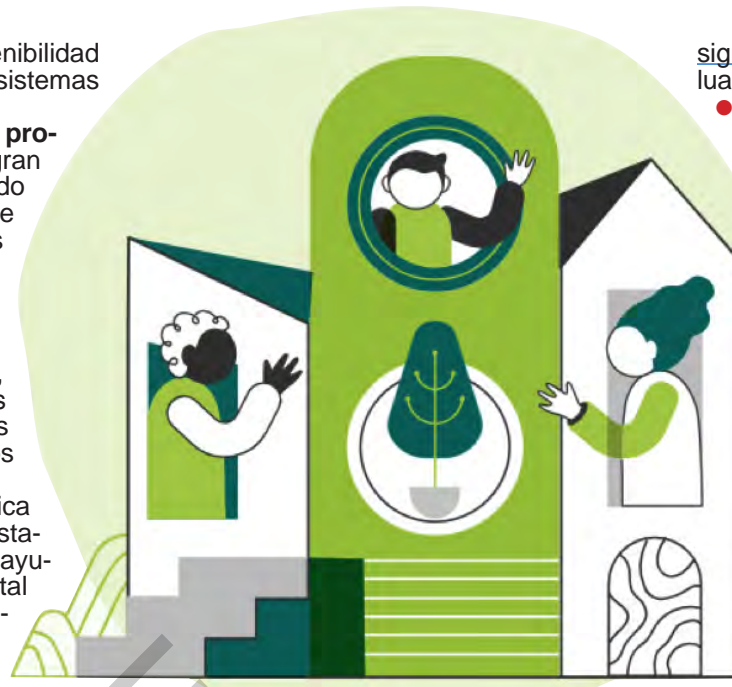
La Ingeniería del Software Verde es un campo emergente que requiere un enfoque consciente y proactivo para reducir el impacto ambiental del desarrollo y uso del software. La perspectiva 'verde' estimula la adopción de buenas prácticas que contribuyen a la sostenibilidad, así como a mejorar la eficiencia y el rendimiento del software. Esto se logra 'por diseño': mediante acciones intencionadas y conscientes de los profesionales en Ingeniería del Software.

Los **profesionales de la ingeniería del software** se enfrentan a una pregunta crucial: ¿Cuáles son algunas prácticas de desarrollo de software que deben realizarse para que una aplicación sea energéticamente sostenible? Basándonos en nuestra investigación, hemos identificado varias prácticas clave que pueden ayudar a reducir el consumo energético durante el desarrollo y la operación de aplicaciones de software. Así, podemos sugerir las siguientes iniciativas:

Estrategia de Monitoreo y Medición

Una estrategia robusta de monitoreo y medición es esencial. Existen diversas herramientas en el mercado que, mediante métricas, permiten identificar procesos, servicios o tecnologías que consumen más recursos de hardware y, por ende, más energía. Para ello es fundamental:

- Definir métricas de utilización de hardware: CPU (procesador), memoria, almacenamiento, interfaces de red, entre otros. Esto permitirá identificar cuáles recursos están consumiendo más energía, y establecer una base para realizar mediciones y comparaciones futuras.
- Establecer mecanismos de recolección e interpretación de métricas: Esto permite entender mejor el consumo energético de cada componente. Se podrían utilizar herramientas que generen gráficos visuales para revisar el comportamiento de los componentes a



Ser inclusivo y abierto a todos

- Implementar planes proactivos basados en los resultados: Mejorar continuamente los productos de software a partir de los datos obtenidos.

Diseño de Arquitectura Sostenible

El diseño de una arquitectura alineada con los requerimientos de la aplicación y los objetivos sostenibles (*Sustainability-Driven Design*) es crítico. Las organizaciones deben establecer metas verdes y trabajar de manera holística, incorporando principios de sostenibilidad en todas las etapas del desarrollo. Por ejemplo:

- Optimización de procesos de ingeniería de requerimientos: El levantado de requerimientos de forma efectiva puede ayudar a evitar re-trabajo que eventualmente podría acarrear consumos mayores de recursos en las fases operativas y de desarrollo.
- Selección de arquitecturas eficientes y tecnologías apropiadas: Decisiones durante la etapa de diseño, como la selección de un lenguaje de programación pueden impactar el consumo energético en los recursos de hardware. Lo mismo sucede con arquitecturas, entre diversos estilos de diseño arquitectónico algunos son más recomendables.
- Implementación y prueba de prácticas de desarrollo sostenibles: Durante la etapa de programación, integración y pruebas, se pueden seleccionar prácticas eficientes en rendimiento, que generen impactos positivos en el hardware: concurrencia y utilización de memorias de acceso rápido para evitar accesos innecesarios a la base de datos, entre otros. Es importante, establecer pruebas de rendimiento y escalabilidad que permitan asegurar el correcto funcionamiento del software, así como una utilización eficiente de los recursos. El sitio Web <https://sustainablewebde->



Participar con empatía

[sign.org/](https://www.websitecarbon.com/) facilita guías de diseño y evaluación de sitios Web sostenibles.

- Monitoreo continuo para asegurar el cumplimiento de los objetivos de sostenibilidad: Es necesario continuar midiendo el rendimiento del software cuanto a sostenibilidad, incluso después de que la aplicación sea puesta en producción, para asegurar la mejora continua. Por ejemplo, hay herramientas para evaluar la huella de carbono de los sitios Web, como: <https://www.websitecarbon.com/>

Documentación de Resultados y decisiones

Es fundamental documentar todos los hallazgos y decisiones tomadas durante el desarrollo. Como cada aplicación tiene aspectos únicos como tecnologías, requerimientos y prácticas específicas de la organización, la documentación ayuda a:

- Tomar decisiones futuras informadas: Al registrar las decisiones y los resultados obtenidos, se facilita el análisis y la comparación de enfoques pasados, lo que permite elegir las mejores prácticas en proyectos futuros.
- Capacitar a los miembros del equipo en prácticas sostenibles: La documentación detallada proporciona un recurso valioso para la formación continua de los miembros del equipo, asegurando que todos estén alineados con las estrategias sostenibles y puedan aplicar conocimientos previos a nuevos contextos.
- Evaluar y ajustar metas y estrategias: Documentar los resultados permite una evaluación continua de las estrategias implementadas, facilitando ajustes y mejoras basadas en datos concretos y experiencias previas, abriendo paso a nuevos hallazgos que pudieran ser implementados.

Inclusión y participación

Los **usuarios de las aplicaciones** también pueden contribuir en hacer un uso responsable del software (Web, móvil, local), por lo que es importante llevar a cabo iniciativas de concientización y capacitación dirigidas a los usuarios para fomentar prácticas que reduzcan el consumo energético y, a su vez, beneficien al ambiente. Algunas prácticas sencillas y efectivas incluyen:

- Uso responsable de los recursos del dispositivo: Cerrar aplicaciones no utilizadas para minimizar el consumo de CPU, memoria y batería.
- Configuración de ajustes de energía: Ajustar el brillo de la pantalla y activar

Agradecimientos

Christian Aarón Campos Lucas es Ingeniero de Software Sénior, con más de 7 años de experiencia desarrollando aplicaciones y sistemas de software para diferentes empresas, pre-cedido por trabajos en diversas áreas de la informática como software, redes, infraestructura y soporte técnico. Christian es graduado de Maestría en Ingeniería del Software con énfasis en Arquitectura y Diseño de la Universidad CENFOTEC, donde investigación titulada 'Recomendación de prácticas de desarrollo de software, en una arquitectura de microservicios, que sean ambientalmente sostenibles' mereció la distinción Suma cum laude. Christian un Bachiller en Ingeniería en Sistemas Computacionales por la Universidad Latina.

Ignacio Trejos Zelaya es Co-Fundador y Profesor Investigador en la Universidad CENFOTEC y Profesor Catedrático de Ingeniería en Computación en el Tecnológico de Costa Rica. Investiga sobre Educación en Informática, Lenguajes de programación e Ingeniería del software. Ignacio estudió Maestría y Doctorado en la Universidad de Oxford y es Ingeniero en Computación del TEC. Es Director de Investigación del Club de Investigación Tecnológica y Representante de Costa Rica en el Consejo Hispanoamericano de Pruebas de Software (HASTQB). Ignacio posee certificaciones profesionales en Ingeniería de Calidad del Software (ASQ) y Pruebas de Software (ISTQB).

Las imágenes ilustrativas provienen de la Green Software Foundation <https://greensoftware.foundation/>

los modos de suspensión por inactividad para conservar energía.

- Actualización regular del software: Mantener el software actualizado para aprovechar las mejoras de eficiencia energética que a menudo acompañan las nuevas versiones.

Implementar estas acciones no solo contribuye a la sostenibilidad ambiental, sino que también puede mejorar el rendimiento y la vida útil de los dispositivos utilizados. Los usuarios en ambientes laborales llevan las prácticas sostenibles a sus familias.

Conclusiones

El desarrollo de software verde es un campo que aún requiere considerable investigación, especialmente en lo que respecta a arquitecturas modernas y prácticas de desarrollo ampliamente aceptadas para la mayoría de las aplicaciones actuales. Nuestras recomendaciones pueden ser de ayuda para los profesionales del software como para los usuarios, permitiéndoles contribuir al cuidado del ambiente.

Es determinante continuar incentivando a las organizaciones a adoptar prácticas de desarrollo de software sostenible. Para ello, es necesario profundizar en investigaciones que evidencien el impacto económico, así como los aspectos de rendimiento y eficiencia del software y la extensión de la vida útil de los recursos de hardware. Estas investigaciones pueden proporcionar datos tangibles que demuestren los beneficios de las prácticas sostenibles, facilitando su adopción a gran escala.

Debemos reconocer que pequeños cambios pueden marcar una gran diferencia. El software está presente en todos los aspectos de nuestra vida diaria: si muchas personas adoptan prácticas sostenibles, el impacto acumulativo puede ser significativo en la meta común de mejorar el ambiente.

Implementar estas estrategias no solo ayuda a reducir el consumo energético y a minimizar el impacto ambiental, sino que también puede mejorar el rendimiento y la longevidad de los dispositivos, beneficiando tanto a los usuarios como a las organizaciones en términos económicos y operativos. Todo se traduce en la mejora de la calidad de vida de los humanos y contribuir a sostener la vida en nuestro planeta.

Recomendamos seguir la *Green Software Foundation*, que lidera el movimiento global por conseguir la adopción de prácticas sostenibles en el desarrollo de software: <https://greensoftware.foundation/>