TABLA. CARACTERÍSTICAS SOSTENIBLES DEL SOFTWARE

Características |Conceptos relacionados a sostenibilidad del software	**Descripción**	**Referencias**
Modificabilidad	Se refiere a la capacidad de adaptación y mejora continua para cumplir con los criterios de sostenibilidad ambiental, económica y social a lo largo del tiempo y su correcto funcionamiento.	[1]
Modularidad	Se refiere a la práctica de dividir un sistema de software en módulos independientes y cohesivos, lo que contribuye a su capacidad para adaptarse y evolucionar a lo largo del tiempo de manera sostenible.	[1] [15]
Usabilidad	Se refiere a la facilidad de ser comprensible, accesible y útil para sus usuarios, al tiempo que promueve prácticas sostenibles durante su desarrollo	[1]
Accesibilidad	Esto implica que pueda ser usado por todas las personas, independientemente de sus capacidades o características individuales, desde la sostenibilidad promueve la igualdad de oportunidades y la inclusión a largo plazo.	[1]
Soportabilidad	Se refiere a la capacidad de adaptarse y mantenerse relevante en una variedad de entornos tecnológicos, legales y ambientales a lo largo del tiempo, asegurando su utilidad y sostenibilidad a largo plazo.	[1]
Eficiencia	Se refiere a la economía de recursos, tiempo de cómputo y espacio de memoria	[1] [15][16]
Desempeño	Se refiere a la optimización operativa con un enfoque en el impacto ambiental y de largo del tiempo.	[1][16]
Portabilidad	Se refiere a la ejecución en diferentes entornos de manera eficiente y efectiva, promoviendo la utilización responsable de los recursos	[1]
Adaptabilidad	Se refiere a la capacidad para ajustarse y responder de manera efectiva a cambios en sus requisitos, entorno operativo, tecnologías relacionadas.	[1][16]
Escalabilidad	Se refiere al diseño del sistema que permita crecer de manera eficiente y responsable, manteniendo un equilibrio entre la capacidad de respuesta a la demanda y/o tamaño.	[1][16]
Flexibilidad	Permite asegurar que el sistema pueda evolucionar y adaptarse a los cambios de manera eficiente y efectiva, promoviendo así su longevidad.	[1][16]
Mantenibilidad	Capacidad para ser comprendido, adaptado, mejorado y mantenido de manera eficiente y efectiva a lo largo de su ciclo de vida	[1] [15][16]
Seguridad	Se refiere a la protección de los datos, los recursos y las operaciones contra amenazas y vulnerabilidades de manera continua y a largo plazo.	[1] [15]
Eficiencia energética, consumo energético	Grado en el que un producto de software consume energía mientras está activo.	[1] [15] [16][17]
Efecto e impacto ambiental	Influencias ambientales causadas por la producción y uso de tecnologías de la información.	[1] [15][16]
Vida útil del hardware-Residuos y obsolescencia (recursos computacionales y de infraestructura)	Se centra en maximizar la durabilidad, eficiencia y capacidad de adaptación del hardware utilizado, mientras se minimiza su impacto ambiental y se promueven prácticas de gestión responsable de recursos.	[1] [15]
Capacidad sin conexión	Se refiere a la capacidad para funcionar de manera efectiva incluso cuando no está conectado a una red o a internet, en el contexto de la sostenibilidad permite reducir la dependencia de recursos externos y la huella ambiental asociada.	[1]
Factibilidad	Describe como un proyecto y los procesos de desarrollo, mantenimiento y uso siguen los principios de sostenibilidad del software	[1]
Impacto y responsabilidad social	Busca garantizar que el desarrollo y uso de tecnología contribuyan positivamente al bienestar de las personas con prácticas éticas y justas.	[15]
Legalidad	Capacidad para asegurar que el desarrollo, distribución y uso cumplen con todas las leyes y regulaciones, también, promueve la confianza, la transparencia y la sostenibilidad en su ciclo de vida.	[15]
Educación	Se refiere a la necesidad de formar a los profesionales de software para que conozcan los requisitos y métodos de un software respetuoso con el medio ambiente.	[1]

TABLA II. PRÁCTICAS SOSTENIBLES EN SOFTWARE

Prácticas sostenibles	Descripción	Referencias
Sostenibilidad en el ciclo de vida del software		[1]
Gestión de requisitos	Se refiere a identificar, documentar, validar y gestionar los requisitos funcionales y no funcionales del software y que a lo largo de su ciclo de vida sea sostenible.	[19] [20]
Gestión y documentación ágil	Se refiere a la aplicación de los principios ágiles en la gestión y documentación de los procesos de desarrollo, entrega y operación de software dentro de un enfoque DevOps	[19] [20]
Diseño modular y escalable	Se refiere a la práctica de diseñar sistemas divididos en módulos independientes y cohesivos, capaces de crecer y adaptarse fácilmente a medida que las necesidades del negocio cambian.	[19] [20]
Aseguramiento de Calidad de software sostenible	Se refiere a la integración de principios y prácticas de calidad en todo el ciclo de vida del desarrollo de software, con el objetivo de garantizar la entrega de software de alta calidad de manera continua y sostenible.	[19] [20]
Optimización de recursos	Se refiere a la práctica de utilizar eficientemente los recursos disponibles en el proceso y operación del software. Esperando minimizar el desperdicio, reducir el tiempo de espera, y maximizar el valor entregado al cliente utilizando la menor cantidad de recursos posibles, como tiempo, dinero, hardware, energía	[19] [20]
Entrega continua	Se refiere al proceso automatizado y repetible de integrar, probar y desplegar cambios de forma continua, oportuna y segura.	[19] [20]
Integración continua	Consiste en fusionar de manera frecuente y automatizada cambios realizados a nivel del código fuente de un proyecto en un repositorio compartido.Esta práctica tiene implicaciones positivas para la sostenibilidad al reducir desperdicio.	[19] [20]
Control de versiones	Se refiere a la gestión y rastreo de cambios en el código fuente y otros artefactos de software a lo largo del tiempo utilizando herramientas específicas, para mantener un historial, revertir a versiones anteriores si es necesario, y/o coordinar el trabajo en paralelo.	[19] [20]
Gestión de la configuración	Se refiere al control de los elementos de configuración del software, como pueden ser el código fuente, la infraestructura, las dependencias y o todo aquello necesario para el desarrollo, la implementación y la operación del sistema.	[19] [20]
Desarrollo basado en troncos	Consiste en trabajar directamente sobre la rama principal (o tronco) del repositorio de código, en lugar de crear ramas separadas para el desarrollo de nuevas funcionalidades o correcciones de errores. Esta práctica promueve la integración continua y la entrega continua al facilitar una colaboración más fluida y una mayor visibilidad del progreso del trabajo en curso.	[19] [20]
Pruebas continuas y automatizadas	Se refiere a la ejecución automática de pruebas de software en cada etapa del proceso de desarrollo y entrega, desde la integración hasta la implementación y más allá.	[19] [20]
Desarrollo dirigido por hipótesis	Se centra en la experimentación y la validación de ideas a través de hipótesis claras y medibles.Se definen experimentos y luego se ajustan en función de los resultados obtenidos.	[16]
Gestión de datos de pruebas	Se refiere a la administración de los datos utilizados en las pruebas de software, esto implica asegurar que estos sean confiables, consistentes, relevantes y estén disponibles cuando sean necesarios.	[19] [20]
Despliegue continuo	Se refiere al proceso de implementación de cambios en el software de manera que estos cambios puedan ser entregados a ambientes de producción de forma rápida, frecuente y confiable.	[19] [20]
Sostenibilidad técnica	Se refiere a la capacidad de un sistema de software para mantener su integridad, calidad y eficacia a lo largo del tiempo, incluso a medida que se realizan cambios y se añaden nuevas funcionalidades	[1]

Gestión técnica o de arquitectura	Se refiere a las prácticas y procesos utilizados para gestionar la calidad técnica y la arquitectura del software a lo largo de su ciclo de vida. Esto incluye decisiones relacionadas con el diseño, la implementación, la refactorización, la automatización, la integración continua, la entrega continua y otras actividades propias de DevOps	[19] [20]
Arquitectura débilmente acoplada	Se refiere al diseño e implementación de manera que tienen pocas dependencias entre sí, limitando el impacto en la integración, la evolución y la mantenibilidad del sistema	[19] [20]
Infraestructura como código	Se refiere a la práctica de gestionar y provisionar infraestructura de TI utilizando archivos de código, esto implica escribir scripts o definiciones de configuración utilizando lenguajes de programación.	[19] [20]
Infraestructura de nube	Se refiere a la utilización de recursos informáticos, como servidores, almacenamiento y redes, que son proporcionados y gestionados a través de servicios de nube pública, privada o híbrida.	[19] [20]
Seguridad como Código	Se refiere a la configuración de seguridad, las políticas de acceso, la detección de vulnerabilidades y otras medidas de seguridad como código, lo que significa que se gestionan y aplican de manera automatizada y reproducible, utilizando herramientas.	[19] [20]
Seguridad por diseño	Se refiere a integrar la seguridad en todas las etapas del ciclo de vida del desarrollo de software, desde el diseño hasta la implementación y operación.	[19] [20]
Control del impacto ambiental	Se refiere a las acciones y medidas que se toman para mitigar y gestionar el impacto ambiental causado por el desarrollo y uso de software, en términos de consumo de recursos, emisiones de carbono, generación de residuos electrónicos, entre otros aspectos.	[1]
Medición del consumo energético del proceso	Se refiere a la evaluación y cuantificación del uso de energía asociado con el ciclo de vida completo de desarrollo de software, desde la fase de diseño y codificación hasta la implementación, operación y eventual retirada del sistema.	[1]
Medición del consumo energético del software	Se refiere al proceso de cuantificar la cantidad de energía que un sistema de software consume durante su funcionamiento. Esto es importante ya que tiene un impacto significativo en el medio ambiente y en los costos operativos de una organización.	[1]
Medición del consumo energético del hardware	Se refiere a la evaluación y cuantificación de la cantidad de energía que consume el hardware utilizado en los sistemas de software, su medición es importante dado el impacto ambiental.	[1]
Reciclaje y reuso de recursos de cómputo e infraestructura	Se refiere a la práctica de aprovechar eficientemente los recursos tecnológicos disponibles, tanto hardware como software, con el fin de minimizar el consumo de recursos y reducir el impacto ambiental.	[1]

TABLA III. PRÁCTICAS SOSTENIBLES EN LA DIMENSIÓN HUMANA PARA EL DESARROLLO DE SOFTWARE

Prácticas sostenibles	**Descripción**	**Referencias**
Responsabilidad social	Contribuye al beneficio del equipo, los usuarios y la Sociedad en general alrededor del software.	[15]
Integración de ética social	Aspectos de legalidad, transparencia en la recopilación y uso de datos, privacidad y los derechos de los usuarios.	[15]
Entornos inclusivos	Asegurar la representación equitativa de diversas identidades y perspectivas en los equipos de desarrollo.	[10][11]
Accesibilidad a los recursos y herramientas	Recursos de conectividad y energía y herramientas de software que sean accessibles a los desarrolladores.	[10][11]
Gestión del talento	Implementar políticas y procesos que reconozcan y recompensen el rendimiento excepcional, así como también el bienestar y la salud mental de los empleados.	[10][11]
Generación de empleo	Fomento en el desarrollo profesional y económico de los miembros del equipo.	[1] [10][11]
Necesidades en contexto	Diseño de soluciones basadas en las necesidades actuales de la comunidad	
Educación en prácticas sostenibles	Capacitaciones al equipo sobre prácticas sostenibles fomentando la conciencia sobre el impacto ambiental de las decisiones de desarrollo de software.	[1] [7] [8]
Gestión Lean	Crear valor para los clientes, el aseguramiento de la calidad, la creación del flujo, el liderazgo con humildad y respeto por cada individuo.	[19] [20]
Aprobación de cambios	Garantizar que los cambios propuestos en el software sean evaluados y aprobados antes de su implementación, lo que requiere comunicación efectiva por parte del equipo.	[19] [20]
Notificación proactiva de fallas	Mantener de forma permanente la socialización y/o documentación de aprendizajes del equipo de trabajo.	[19] [20]
Visualización del trabajo	Mantener de forma permanente la socialización y/o documentación de aprendizajes del equipo de trabajo.	[19] [20]
Límite de trabajo en proceso	Asegurar que los equipos no sean sobrecargados y expone los obstáculos del flujo de trabajo.	[19] [20]

Monitoreo y observabilidad continua	Observar y comprender el estado de sus sistemas mediante métricas o logs y depurar activamente su sistema explorando propiedades que se requieran.	[19] [20]
Retroalimentación y mejora continua	Mantener de forma permanente la socialización y/o documentación de aprendizajes del equipo de trabajo.	[19] [20]
Trabajo por lotes pequeños	Permite probar rápidamente hipótesis sobre una mejora en particular para que tenga el efecto que desea.	[19] [20]
Experimentación en equipo	Los equipos experimentan creando prototipos y probando ideas rápidamente a la medida.	[19] [20]
Gestión cultural	Cooperación entre los miembros de una organización, con normas para el trabajo, comunicación, dirección de problemas y calidad de soluciones.	[19] [20]
Cultura organizativa generativa	Fomenta la creatividad, la colaboración y la capacidad de adaptación del equipo experimentando con nuevas ideas y enfoques.	[19] [20]
Aprendizaje continuo	Es la capacitación constante que tienen los equipos para actualizar sus conocimientos en habilidades técnicas y blandas.	[19] [20]
Colaboración entre equipos	Forma de trabajo en la que los equipos del proyecto trabajan en conjunto.	[19] [20]
Medición del rendimiento	Se busca medir el rendimiento del proceso, el software (todas las características de la tabla I) y el equipo (la productividad y bienestar)	[19] [20]
Satisfacción laboral	Generar bienestar en los empleados en el desarrollo profesional, en el ambiente laboral y en su salud física, emocional y mental.	[19] [20]
Liderazgo transformacional	Inspirar y motivar al equipo para alcanzar niveles más altos de desempeño y lograr cambios significativos en la organización.	[19] [20]