Pontificia Universidad Católica del Ecuador

Integrantes: David Hidalgo, Daniel Valenzuela, Mateo Paredes

Materia: Ingeniería de software

Carrera: Ingeniería en Sistemas de información

Ensayo código de ética del ingeniero de software

Tiempo estimado (Unificación): 10 minutos

Principio 1. Sociedad

La primera regla del código de ética establece que los ingenieros de software deben actuar de forma congruente con el interés social. A mi parecer puede hacer referencia a que mi trabajo como ingeniero tiene efectos en la vida real de varias personas. Mi rol al diseñar y realizar sistemas no es solo satisfacer a mi cliente, si no crear algo que realmente pueda ayudar a la gente, siempre buscando el bien común, por lo que no debería hacer darle un visto bueno a software o proyectos que no sean seguros o que pueda afectar la calidad de vida de la gente. A mi parecer la confianza que las personas tienen en tu producto o tu palabra vale más que dinero, es por eso, por lo que debo siempre pensar en crear cosas que beneficien y ayuden a la sociedad.

Principio 2. Cliente y empresario.

La siguiente norma nos habla sobre la relación entre ingenieros y quienes los contratan. Si quiero hacer un buen trabajo tengo que ser honesto siempre, ser consiente sobre mis propios límites y no aceptar proyectos solo por el dinero. Siempre debo pensar en hacer las cosas de manera correcta, no debo dejar que el dinero deje que tome la decisión y entregue un producto malo. Como ya mencioné antes siempre debo pensar en ser de ayuda para la sociedad, encubrir algo que obviamente puede ser peligroso para todos seria participar en ello, y yo no quiero ser así. A pesar de para lo que me contraen, las herramientas que me den o lo que me van a pagar, no voy a poner en riesgo a los demás por ello, sobre todo si sé que está mal.

Principio 3. Producto.

Al leer el punto de Producto, entendí que un ingeniero de software tiene la responsabilidad de crear programas que cumplan con los más altos estándares de calidad. No se trata solo de hacer que el software funcione, sino de hacerlo bien, con ética y pensando en las personas que lo van a usar. También comprendí que debemos asegurarnos de que todo lo que desarrollamos esté bien documentado, probado y sin causar daños a otros. Este punto me hizo ver que el producto final refleja el compromiso y la profesionalidad del ingeniero, porque si uno entrega algo mal hecho, no solo afecta al cliente, sino también a la confianza que la sociedad tiene en nuestro trabajo. En conclusión, el principio de *Producto* enseña que la calidad y la responsabilidad siempre deben ir de la mano.

Principio 4. Juicio.

El punto de Juicio me hizo pensar en la importancia de mantener la integridad en cada decisión profesional. Ser ingeniero de software no es solo tener conocimientos técnicos, sino también saber tomar decisiones justas, objetivas y correctas. Entendí que tener buen juicio significa no dejarse influenciar por intereses externos o por la presión de otros, sino actuar de acuerdo con lo que uno sabe que está bien. También me pareció importante que se hable de la independencia, porque a veces pueden existir conflictos o tentaciones, y ahí es donde se demuestra el verdadero carácter profesional. Para mí, este principio muestra que el juicio ético es lo que mantiene la confianza en uno mismo y en la profesión.

Principio 5. Administración.

Los ingenieros de software en posiciones de liderazgo tienen responsabilidades amplificadas. No solo tengo que ser ético yo, sino que tengo que ser un ejemplo y asegurarme de que mi equipo también pueda serlo. También tengo que asignar tareas según las habilidades de mi equipo me hace pensar en cómo equilibrar el crecimiento profesional con la responsabilidad. Y siempre ayudar a los demás a crecer en conjunto, a no exigir algo a una persona que no sabe hacer, pero a su vez ayudándole a entender y crecer. A mi parecer también tengo que fomentar un buen trato y comunicación, necesito que puedan aprender de mí y yo aprender de ellos, tratarlos bien y recibir el mismo trato, de esta forma evitando enojos innecesarios.

Principio 6. Profesión.

Mis trabajos afectan mi profesión, esta regla menciona que cada ingeniero contribuye a la reputación colectiva de la profesión. Siempre tenemos que buscar mejorar la reputación de la misma, creando grupos donde lo correcto sea algo normal. La ingeniería de software no se trata solamente de programar para alguien, si no también incluye varios aspectos, entre ellos ser ético. Debo entregar no solo productos de calidad, si no en tiempos requeridos y siempre buscando el bien. Debo actuar de manera profesional para no solo mejorar mi imagen si no la de la profesión.

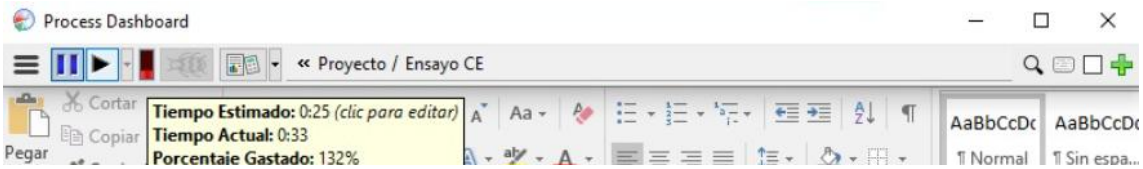
Principio 7. Colegas.

Mis colegas no son solo compañeros de trabajo; son parte de esta profesión que compartimos. Quiero motivarlos a seguir estos principios. Debo siempre trabajar a la par de mis compañeros, buscando opiniones y mejorar nuestro trabajo conjunto siempre, debemos aprender y enseñarnos entre nosotros, reconocer nuestros logros y crecer de manera conjunta. Siempre debemos ser objetivos y hablar con verdades, tenemos que mantener una buena amistad con nuestros compañeros siempre. El reconocimiento completo del trabajo ajeno combate el plagio y la apropiación indebida de ideas, problemas endémicos en muchos campos profesionales. La prohibición de intervenir injustamente en las carreras de colegas debe equilibrarse con la responsabilidad de cuestionar, de buena fe, la competencia de alguien cuando el interés del cliente o la sociedad lo requiere.

Principio 8. Personal.

El último principio enfatiza que ser ingeniero de software no es un estado alcanzado mediante un título, sino un compromiso de desarrollo continuo a lo largo de toda la vida profesional. Lo que es mejor práctica hoy puede ser obsoleto mañana. Los ingenieros tienen la responsabilidad de mantenerse actualizados, no solo por su propio beneficio profesional, sino porque trabajar con conocimientos obsoletos pone en riesgo a usuarios y clientes. Mejorar la habilidad para crear software seguro, confiable y de calidad va más allá del conocimiento teórico: requiere práctica deliberada y reflexión sobre experiencias pasadas. Mejorar la habilidad para crear software seguro, confiable y de calidad va más allá del conocimiento teórico: requiere práctica deliberada y reflexión sobre experiencias pasadas. La calidad no es accidental; es el resultado de esfuerzo consciente y sistemático.

Tiempo Mateo Paredes



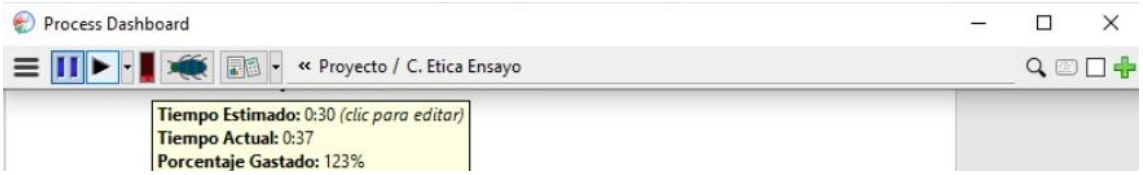
Variación: 33 - 25 = +8 minutos

Desviación: ((33 - 25) / 25) × 100 = 32% de retraso

Ratio: 33 / 25 = 1.32x el tiempo estimado

Propuesta de mejora: Implementar bloqueo temporal de redes sociales usando herramientas durante las tareas. Aplicar la técnica Comodoro con bloques de 25 minutos enfocados, permitiendo revisar notificaciones solo en los descansos de 5 minutos. También usar auriculares con cancelación de ruido o música para minimizar distracciones sonoras.

Tiempo David Hidalgo



Variación: 37 - 30 = +7 minutos

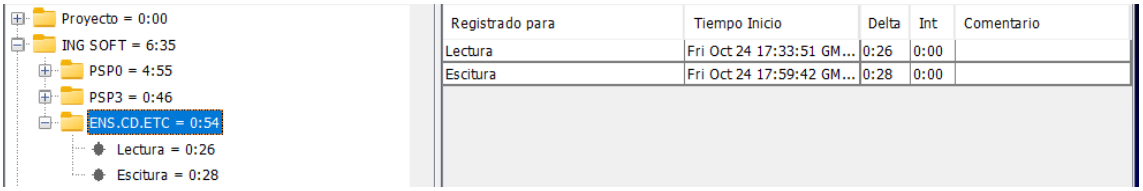
Desviación: ((37 - 30) / 30) × 100 = 23.3% de retraso

Ratio: 37 / 30 = 1.23x el tiempo estimado

Propuesta de mejora: Establecer bloques de "tiempo protegido" en el calendario y comunicar tu disponibilidad mediante indicadores visuales para reducir interrupciones. Negociar con el equipo horarios de menor interrupción para tareas que requieren concentración profunda. A su vez, crear un ritual de inicio de 2 minutos preparando el espacio.

Tiempo Daniel Valenzuela

Tiempo estimado de lectura: 20 min Tiempo estimado de escritura: 30 min



Para la lectura:

Variación = 26 - 20 = +6 min

Desviación = ((26 - 20) / 20) × 100 = 30%

Ratio = 26 / 20 = 1.3x

Para la escritura:

Variación = 28 - 30 = -2 min

Desviación = ((28 - 30) / 30) × 100 = -6.67%

Ratio = 28 / 30 = 0.93x

Propuesta de mejora: Organizar las tareas en bloques de tiempo específicos, utilizando la técnica Pomodoro (25 min de enfoque, 5 min de descanso). Priorizar la lectura comprensiva antes de redactar, y evitar multitareas para mantener la concentración y eficiencia.