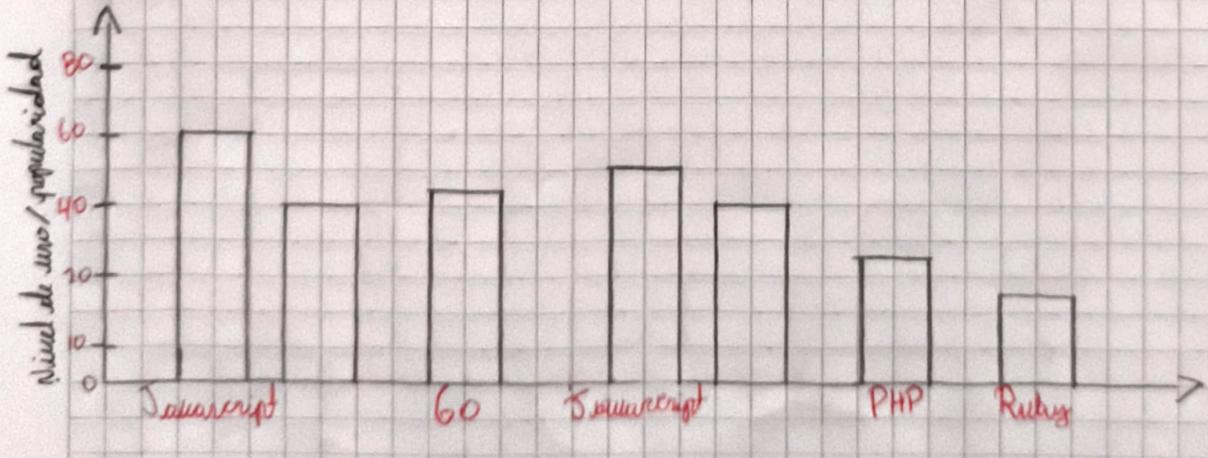


Artículo 1: Estado del Arte en el Desarrollo de Software

Este texto examina la evolución del desarrollo de software y las tendencias que actualmente definen la industria. Se enfoca en la importancia de lenguajes como JavaScript, TypeScript es popular en Frontend, mientras que en el Backend utilizan PHP, Ruby es el propio JavaScript, siendo ampliamente utilizado por grandes empresas como Google o Meta, no se limita únicamente a los lenguajes visto también incluyen frameworks y herramientas que facilitan el trabajo y permiten que los proyectos sean más ágiles y de mayor calidad. Algo que destaca es que no es suficiente con elegir el lenguaje más popular, sino que lo crucial es que hay que considerar la experiencia, la escalabilidad y la experiencia del usuario. Igualmente, se menciona la necesidad de la colaboración entre el Frontend y el Backend para conseguir soluciones integradas y duraderas, el artículo subraya que además de la competencia técnica, lo que realmente importa es la habilidad para adaptarse a las nuevas tendencias.

Gráfico



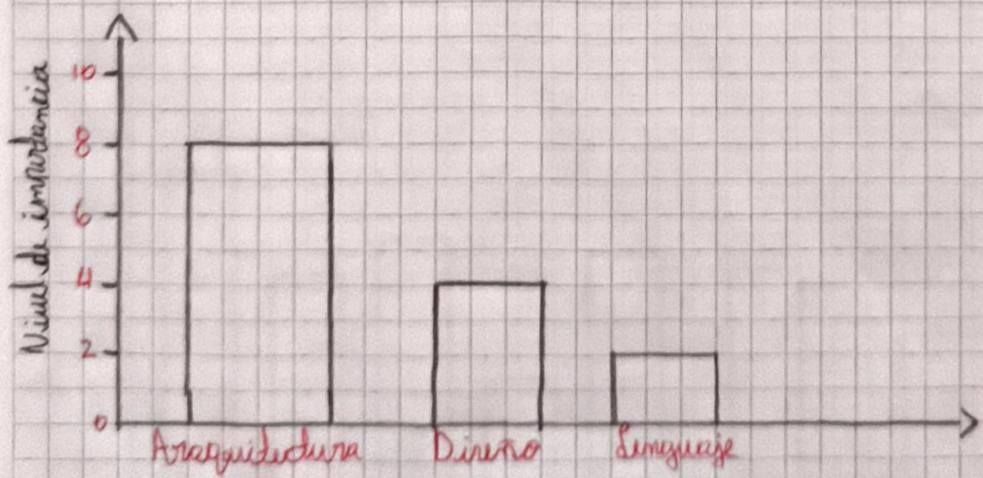
Reflexión:

Este artículo me recuerda que lo importante no es solo aprender un lenguaje, sino comprender su propósito y como se conecta con otros. Adaptarse es seguir aprendiendo y lo que nos mantiene relevantes en el mundo digital.

Artículo 2: Patrones arquitectónicos en el desarrollo de Software

Este artículo nos habla de la importancia de los patrones arquitectónicos en el desarrollo de software, destacando cómo aportan soluciones prácticas y probadas a problemas comunes en proyectos complejos. Explica bien qué los patrones no solo ayudan a organizar mejor el código, sino que también favorecen la seguridad, la escalabilidad y la rapidez en la entrega de resultados. A lo largo del texto se presentan diferentes tipos de patrones, como los de arquitectura, diseño y lenguaje, además de clasificaciones ampliamente reconocidas como POSA y PEA. También se detallan diversas formas en las que los patrones pueden combinarse, ya sea de manera independiente, como complementar un diseño para adaptarse a las necesidades específicas de cada proyecto. El artículo hace énfasis en la importancia de contar con la documentación clara y bien estructurada, ya sea para facilitar tanto la implementación como el mantenimiento a largo plazo.

Gráfica



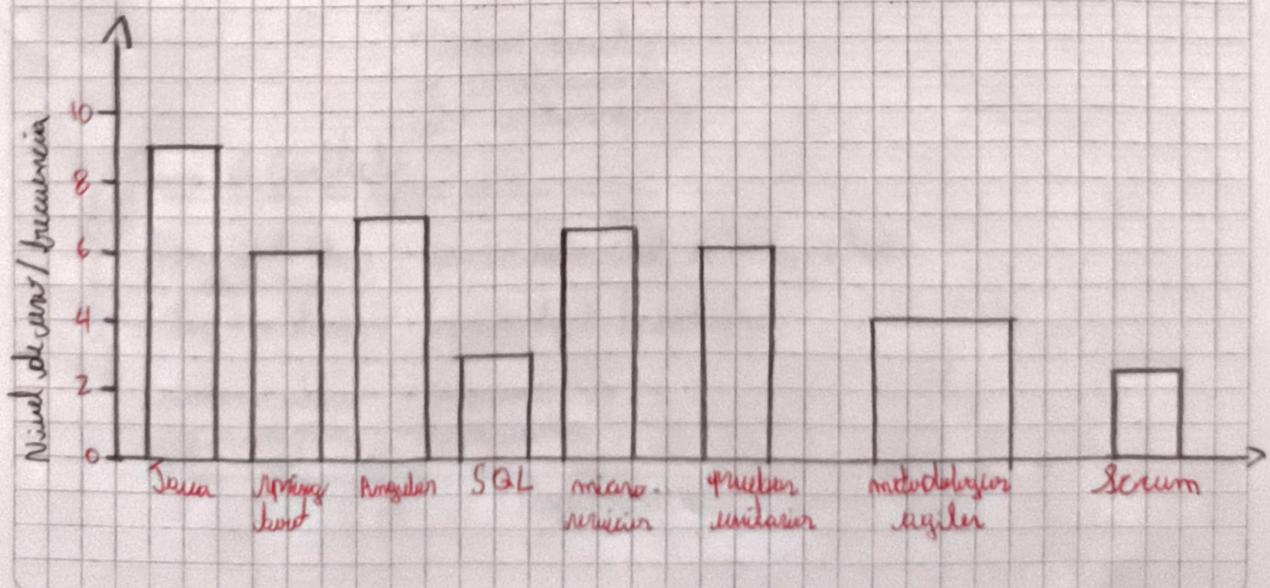
Reflexión:

Este capítulo me hizo ver que la arquitectura de software es como el plano de una ciudad, define como se conectan las partes y asegura que todo funcione de forma ordenada. Serán herramientas que nos ayuden a tomar decisiones más claras y a construir sistemas sólidos desde la base.

Artículo 3: Percepción de framework back-end en estudiantes y expertos.

Este artículo analiza cómo estudiantes y profesionales perciben el uso de frameworks en el desarrollo back-end, mostrando las diferencias entre la visión académica y la experiencia en el campo laboral. A partir de los encuestados y entrevistados, se encontró que los estudiantes valoran más la facilidad de aprendizaje, la versatilidad y el apoyo de la comunidad, mientras que los expertos priorizan aspectos como la seguridad, la eficiencia y la escalabilidad, fundamentales en proyectos empresariales. El artículo reseña una clara brecha entre lo que se enseña en la academia y lo que demanda la industria, lo que resalta la necesidad de promover una mayor integración entre ambos sectores. Se sugiere que los profesionales incorporen experiencias prácticas en los procesos de formación, promover la investigación aplicada y fortalecer la colaboración entre universidades y empresas, de este modo los futuros desarrolladores no solo adquieran conocimientos teóricos, sino una visión más clara con respecto al mundo real.

Gráfico:



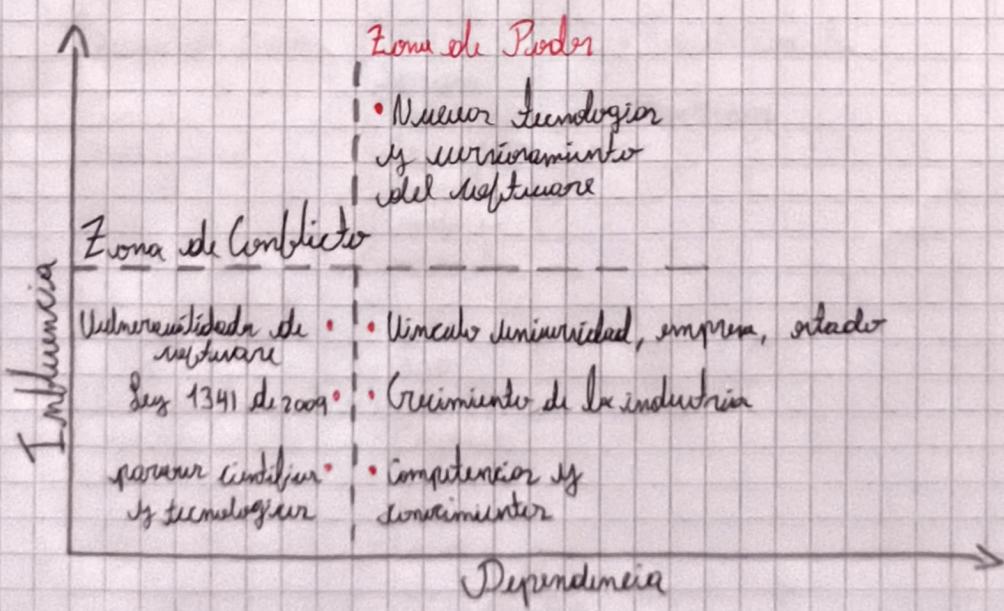
Reflexión:

Este capítulo demuestra que una práctica profesional bien planificada permite aplicar conocimientos técnicos en un entorno real. La combinación de tecnologías modernas y metodologías ágiles fortalece el aprendizaje al preparar para enfrentar reales del mundo laboral con mayor confianza.

Artículo 4 Análisis prospectivo de la industria del desarrollo de software en Colombia

El artículo sobreve un análisis prospectivo de la industria del software en Colombia, explorando sus principales retos y oportunidades de crecimiento. Se reconoce que la demanda de soluciones tecnológicas está en aumento y que el país tiene potencial para consolidarse como un actor relevante en el mercado global. Entre los factores positivos se encuentran la adopción de metodologías ágiles, la internacionalización de servicios y la necesidad de talento técnico calificado. Sin embargo, también se mencionan desafíos importantes como la brecha de habilidades, la competencia con mercados más avanzados y la adaptación a nuevas tecnologías como la IA, nube y ciberseguridad. El artículo también propone que la clave para el desarrollo del sector está en fortalecer la educación, incentivar la innovación y generar alianzas sólidas entre academia, empresas y Estado.

Grafica



Reflexión:

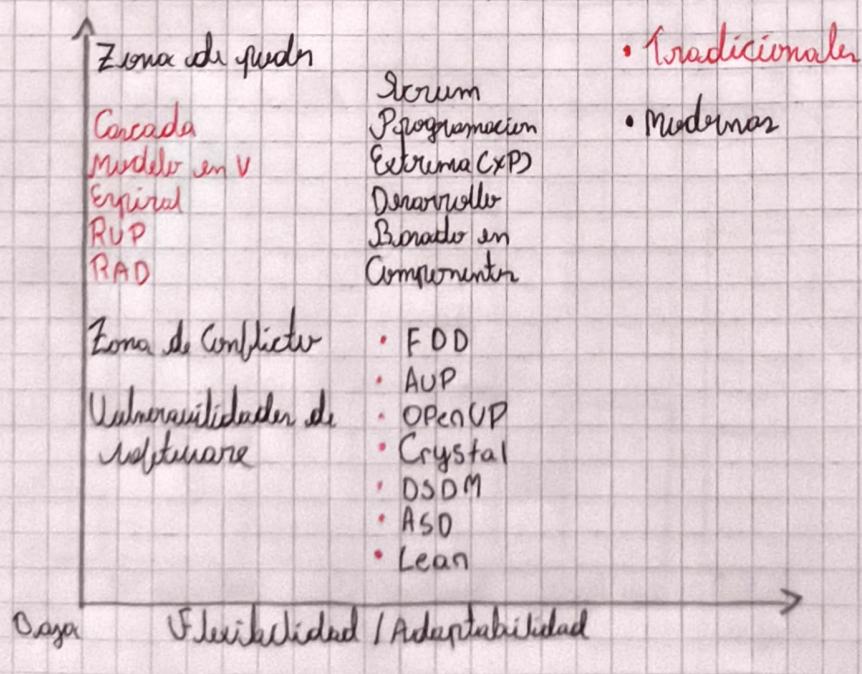
Este análisis muestra que el futuro de la industria de software en Colombia depende de innovar y actualizarse constantemente, mientras se apuestan a los retos como la ciberseguridad, la formación de talento y la sostenibilidad. La clave es anticiparse a los cambios y fortalecer alianzas que impulsen el crecimiento.

Artículo 5 Revisión comparativa de metodologías tradicionales y modernas en desarrollo de software

Este artículo realiza una comparación entre las metodologías, señalando sus ventajas y limitaciones en diferentes contextos. Se explican ejemplos claros como el modelo en cascada, caracterizando su rigidez y planificación secuencial, frente a metodologías ágiles como Scrum y Kanban, que se centran en la flexibilidad, la retroalimentación continua y la colaboración del equipo.

El artículo más dice que, aunque las metodologías tradicionales ofrecen mayor control y documentación, resultan menos efectivas en entornos cambiantes o con necesidades de entregas rápidas. En conclusión, las ágiles permiten responder mejor a la incertidumbre y facilitan la adaptación durante el ciclo de desarrollo. Sin embargo, se celebra que ninguna metodología es universalmente superior, ya que la elección depende del tipo de proyecto, el equipo de trabajo y los objetivos estratégicos.

Grafico:



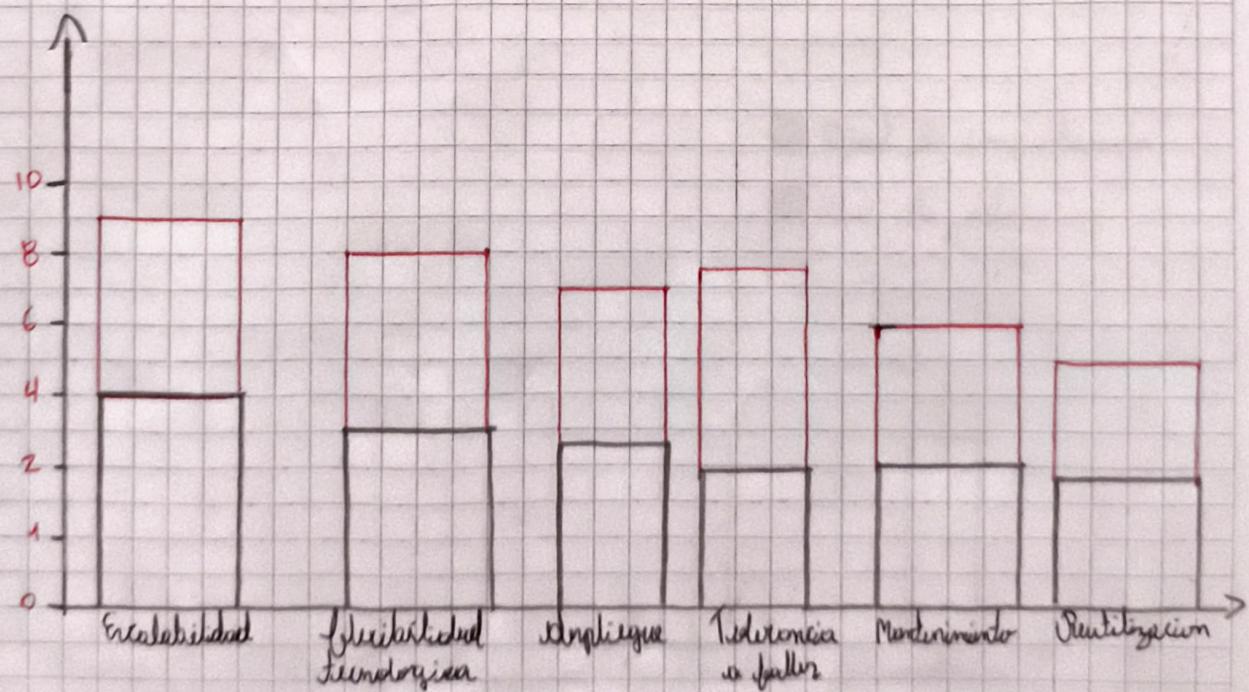
Reflexión:

Este artículo dice claro que no existe una metodología única para liderar los proyectos. Los tradicionales enfocan el control y estructura, mientras que los modernos priorizan adaptabilidad y un ritmo rápido. La clave está en elegir la que mejor se ajuste a las necesidades y al contexto del desarrollo.

Artículo 6: Estudio sobre metodologías de desarrollo y su impacto en la productividad

Este artículo analiza como diferentes metodologías de desarrollo emplean en la productividad de los equipos de software. Explica que los métodos tradicionales, como el modelo de cascada, permiten mayor control en proyectos bien definidos, pero presentan rigidez fuerte a su cambio. En contraste, las metodologías ágiles promueven la adaptabilidad, lo entrega continua y la retroalimentación frecuente. El artículo señala que la elección de la metodología depende tanto del tipo de proyecto como de la cultura del equipo, destacando que no existe un único enfoque universal. También se enfatiza que la productividad no debe medirse únicamente por la rapidez, sino también por el valor entregado al usuario final. En conclusión, el artículo plantea que la clave está en equilibradas estructuras y flexibilidad para alcanzar mejor resultados en el desarrollo de software.

Gráfica:



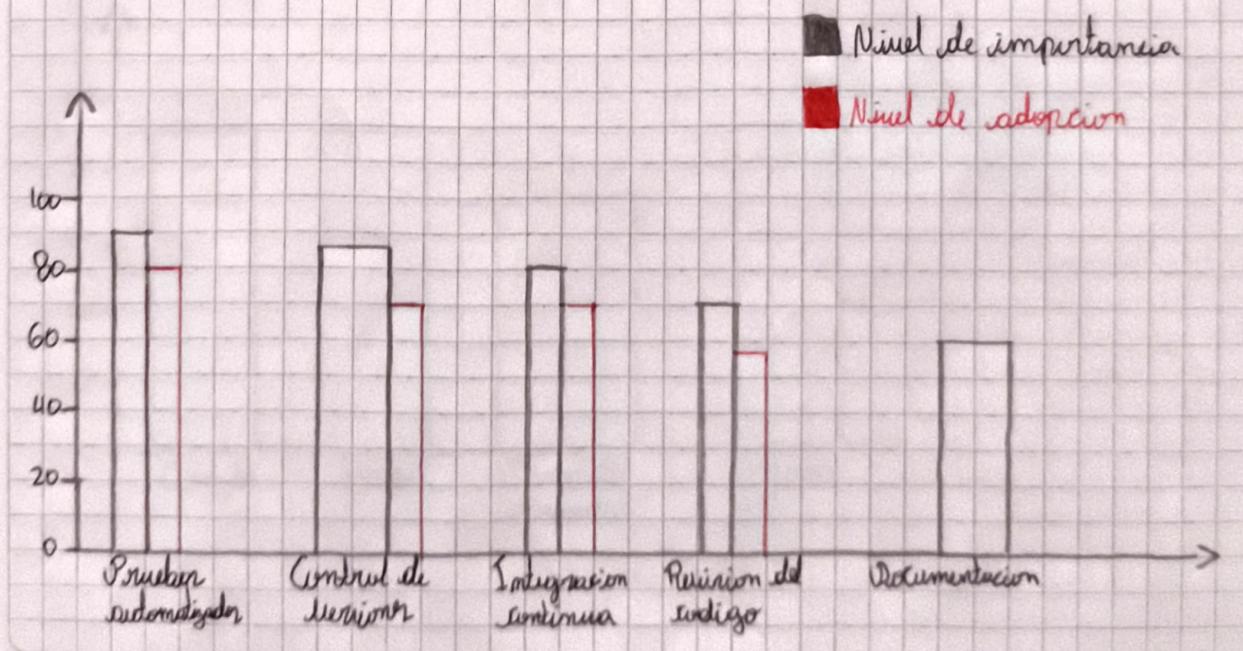
Páginas

Este capítulo muestra que la industria del software en Colombia tiene un gran potencial de crecimiento importante, impulsado por la demanda cada vez mayor en la nube, aplicaciones móviles y servicios de ciberseguridad. La clave para innovar y adaptarse a las tendencias globales para mantener la competitividad.

Artículo 7. Beneficios prácticos en la construcción de software

Este artículo reflexiona sobre el papel del software libre y de código abierto en la industria tecnológica. Explica como este alternativo se ha convertido en una opción viable frente a soluciones propietarias, gracias a su flexibilidad, menor costo y la posibilidad de personalizar las herramientas según las necesidades de cada organización. También se menciona que el software libre fomenta la innovación y el aprendizaje colaborativo, pues permite a los desarrolladores revisar, mejorar y compartir código. El artículo muestra que muchas empresas han adoptado este tipo de soluciones como parte de sus estrategias tecnológicas recomiendo que no solo surgen beneficios económicos, sino que también favorecen la independencia. Frente a proveedores. No obstante, se advierte que el uso de software libre implica desafíos, como la necesidad de personal capacitado y el compromiso de mantener la seguridad de las aplicaciones.

Gráfica:



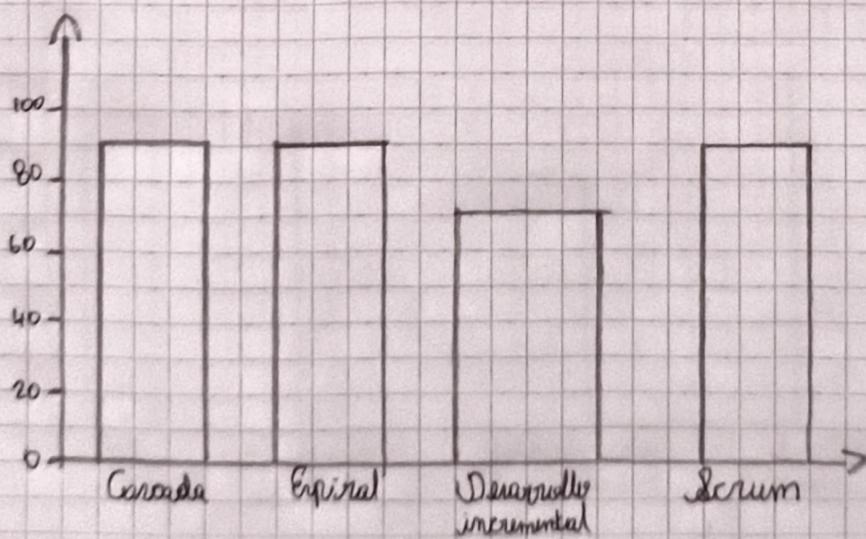
Reflexión:

Este capítulo resalta que los buenas prácticas no solo mejoran la calidad del software, sino que también facilitan el trabajo en equipo y reducen errores. Adentrarse de forma constante en clase para construir sistemas más confiables es sumamente deseable.

Artículo 8. Modelos de procesos de desarrollo aplicados a un proyecto de arquitectura de software

El artículo aborda el tema de la ética en el desarrollo de software, planteando como los profesionales de esta área tienen una responsabilidad que va más allá de lo técnico. Se explica que las decisiones en forma tal dirección y construcción de aplicaciones impactan directamente en la sociedad, lo que hace necesario actuar con principios éticos claros. Se mencionan aspectos como la primariedad de los datos, la seguridad de la información y la transparencia en el uso de las tecnologías. El artículo también resalta la importancia de que los desarrolladores sean conscientes del poder que tienen sus acciones y del impacto social que pueden generar, tanto positivo como negativo, también resalta que la ética que debe ser como una limitación, pero como agua para construir software más confiable y responsable.

Gráfica:



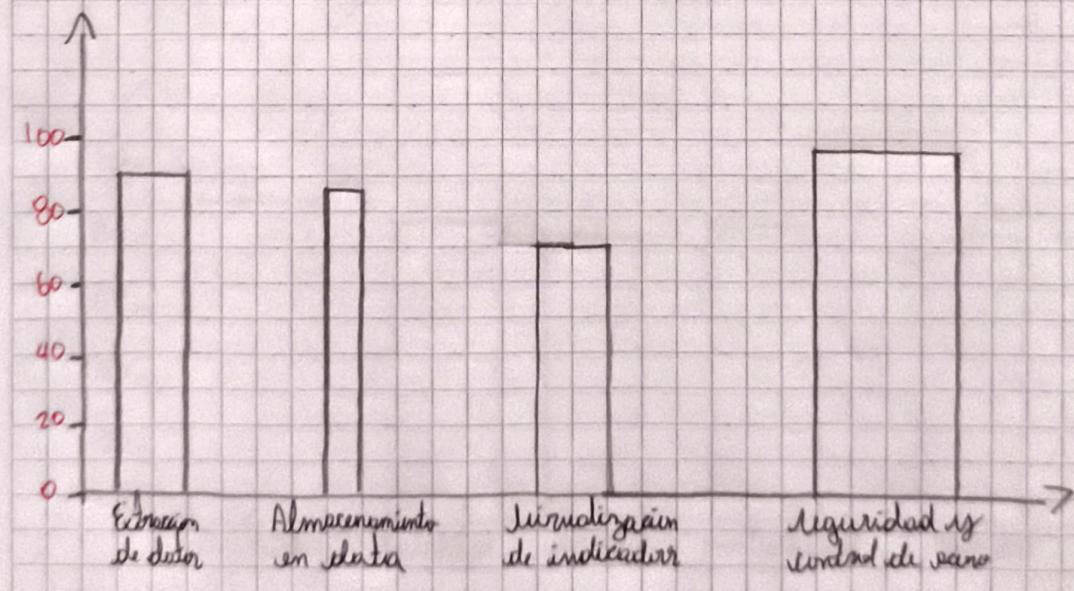
Reflexión:

Este capítulo me dice claro que no hay un modelo perfecto, depende del contexto, para definir arquitectura con requerimientos y alta necesidad de validación temprana, las prototipos ganan por su flexibilidad, la spiral gana cuando el margen opera mucho. Lo importante es elegir un enfoque, no una certeza.

Artículo 9 - Ética en el desarrollo de software

Este artículo se centra en la ética dentro del desarrollo de software, señalando que los profesionales de esta área tienen una responsabilidad social que va más allá de su trabajo. Se discuten temas como la privacidad de los datos, la transparencia en los algoritmos y el impacto social de las aplicaciones creadas. También se menciona que la ética debe guiar la toma de decisiones para garantizar que la tecnología no se utilice con fines perjudiciales. El artículo no habla sobre qué los desarrolladores deben ser conscientes del alcance de sus creaciones, ya que esto puede generar tanto beneficios como riesgos para la sociedad. La ética no se presenta como una limitación, sino como un marco necesario para orientar la innovación de forma responsable. En conclusión, el artículo plantea que un software verdaderamente ético no solo se mide por su funcionalidad, sino también por su compromiso con el bienestar social y la equidad.

Gráfica:



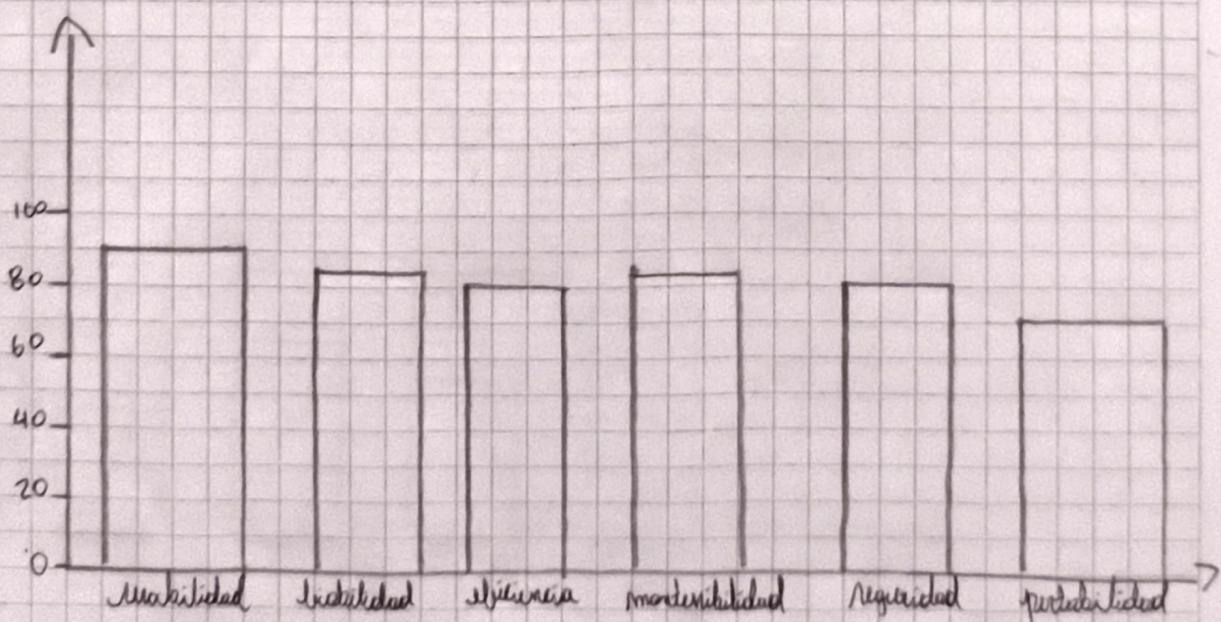
Definición:

Este capítulo muestra que un cuadro de mando integral bien diseñado no solo organiza la información, sino que la convierte en conocimiento útil para la toma de decisiones. La clave es crear sistemas éticamente correctos para nutrir las leyes y ademá servir.

Artículo 40 - Estándares para la calidad de software

Este artículo se centra en la calidad de software y los estándares que permiten garantizarla. Explica que, en un mercado cada vez más exigente, los usuarios demandan productos confiables que cumplan con sus necesidades y para ello han surgido modelos de evaluación como las normas ISO/IEC 9126, 14598 y 2500, así como CMMI y SPICE, estos estándares sirven como guías para medir aspectos como funcionalidad, fiabilidad, eficiencia, mantenibilidad y portabilidad. El artículo se enfoca en que los estándares no solo ayudan a validar software ya implementado, sino también a orientar proyectos de desarrollo, asegurando que los entregables cumplan con expectativas de calidad. Además, se detallan beneficios como mayor eficiencia en los procesos, productos más confiables y la posibilidad de certificar la calidad frente a los competidores.

Gráfica:



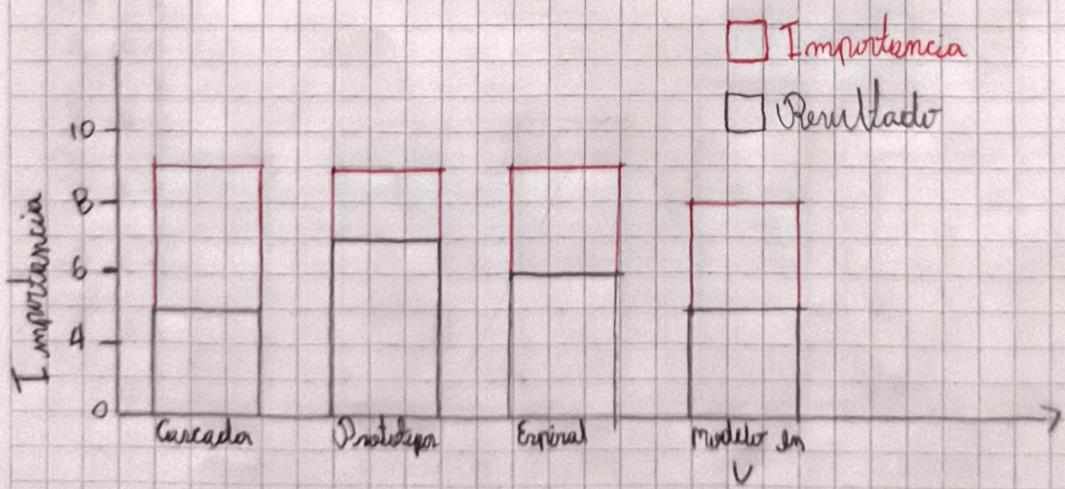
Reflexión

Este capítulo versa sobre los estándares de calidad del software con la idea para desarrollar productos confiables y eficientes. Aplicarlos de forma consistente asegura que el software cumple con las expectativas y necesidades de los usuarios.

Artículo 49 - Modelo de desarrollo de software aplicado al diseño de un prototipo de aplicación web para la gestión de correspondencia en conjunto residencial

El artículo propone un modelo de desarrollo de software aplicado al diseño de un prototipo web para administrar la correspondencia en conjunto residencial. Parte de la necesidad de mejorar la organización y trazabilidad de los documentos que ingresan, pues en muchos casos se pierden o no llegan a tiempo a su destinatario. El prototipo incluye módulos para registrar, clasificar, consultar y generar reportes sobre la correspondencia recibida. Se diseñaron perfiles de usuarios como admin, portero y vecindad cada uno con funciones específicas. En pruebas iniciales, se constató que la herramienta reduce tiempos de entrega y evita perdidas, manteniendo su utilidad como apoyo en la gestión comunitaria.

Gráfico:



Resumen:

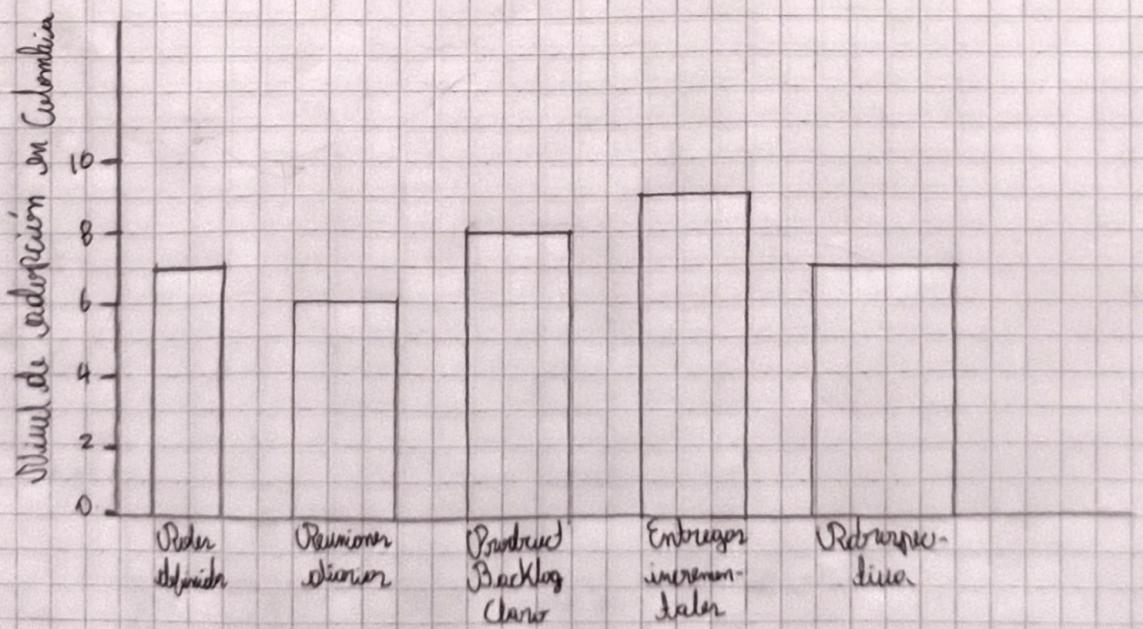
Este capitulo evidencia que, para un prototipo de aplicacion web de gestion de correspondencia, los modelos iterativos como prototipos usados ofrecen mayor flexibilidad y mejor resultado, mientras que modelos mas rigidos como Concreta pueden limitar la adaptacion a cambio.

Artículo 42 - Scrum en la Colombie

Este apartado analiza como se aplica Scrum en equipos colombianos, subiendo adaptaciones realizadas frente al modelo original. En mucha mayor medida que el product owner y Scrum master se combinan, los sprints se flexibilizan según la disponibilidad del cliente y las ceremonias suelen acortarse.

Aunque estas modificaciones permiten entregas rápidas y mejor comunicación, también generan riesgos como perdida de formalidad o acumulación de errores. Se concluye que lo clave eslo es mantener la ejecución de Scrum, con una definición clara de responsabilidades y criterios de calidad, pero evitar caer en prácticas arbitrarias poco efectivas.

Gráfico:



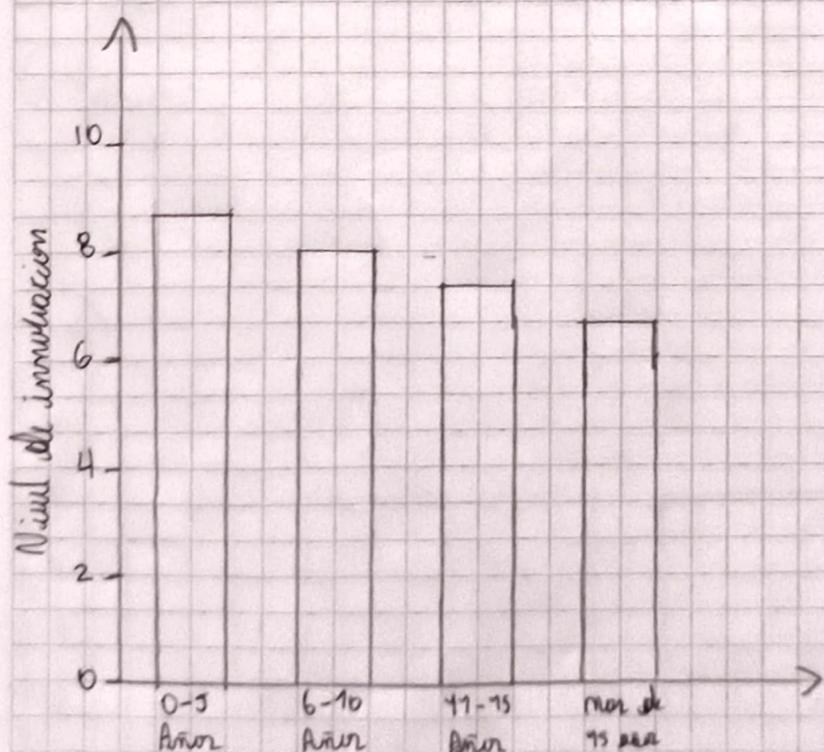
Resumen:

Este capítulo aborda sobre recum la colombiana mantiene la esencia del marco original, pero se adentra a la cultura y a los dinámicos de las empresas. Aunado la importación de las prácticas en alto, la adopción plena aún es un reto, lo que abre oportunidades para mejorar la disciplina y la consistencia en su aplicación.

Artículo 43 - Innovación vs edad de las empresas en un estudio en empresas argentinas de software

El artículo estudia la relación entre la edad de las empresas del software argentino y su capacidad de innovar. Los resultados muestran que las empresas jóvenes son más propensas a arrimar a nuevas tecnologías y formas de trabajo, mientras que las más maduras ofrecen mayor estabilidad y experiencia, aunque tienden a ser más conservadoras. El artículo señala que la innovación no solo depende de la edad, sino de factores como la cultura organizacional, la disponibilidad de recursos y la apertura al cambio. En conclusión, una combinación de experiencia y flexibilidad es lo que realmente favorece la innovación en el sector.

Gráfico:



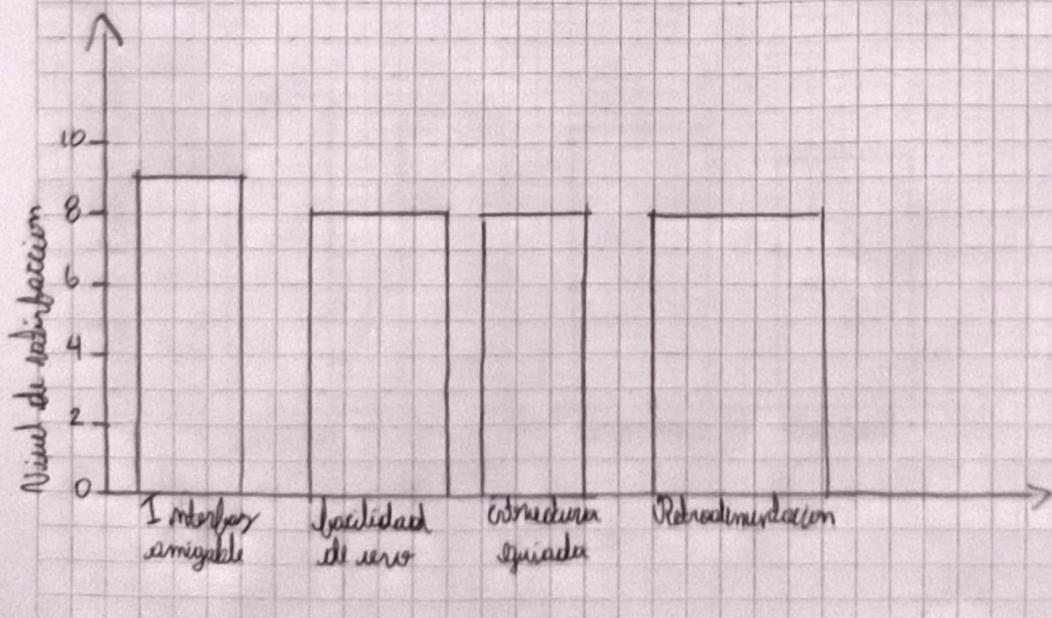
Reflexion

Este capitulo evidencia que los emprees mas jóvenes muestran mayor nivel de innovación y una mayor proporción de innovación en I+D.A. A medida que aumenta la antigüedad, la innovación tiende a disminuir, lo que sugiere la importancia de mantener una cultura innovadora a lo largo del tiempo para seguir siendo competitivo.

Artículo 94 - Software educativo para la formulación de proyectos de investigación

Este artículo describe el desarrollo de un software educativo pensado para guiar a estudiantes en la elaboración de proyectos de investigación. La herramienta organiza el proceso en módulos que abarcan desde el plantamiento del problema hasta la metodología, objetivos y cronograma. También ofrece plantillas, ejemplos y tutor de verificación para mejorar la coherencia del trabajo. En los primeros días estudiantes, se observó que el software ayudaba a estructurar ideas de trabajo de manera clara y ordenada, aunque no sustituyó la orientación del docente. En conclusión, se presenta como un recurso de apoyo que facilita el aprendizaje en la práctica en la formulación de proyectos.

Gráfica



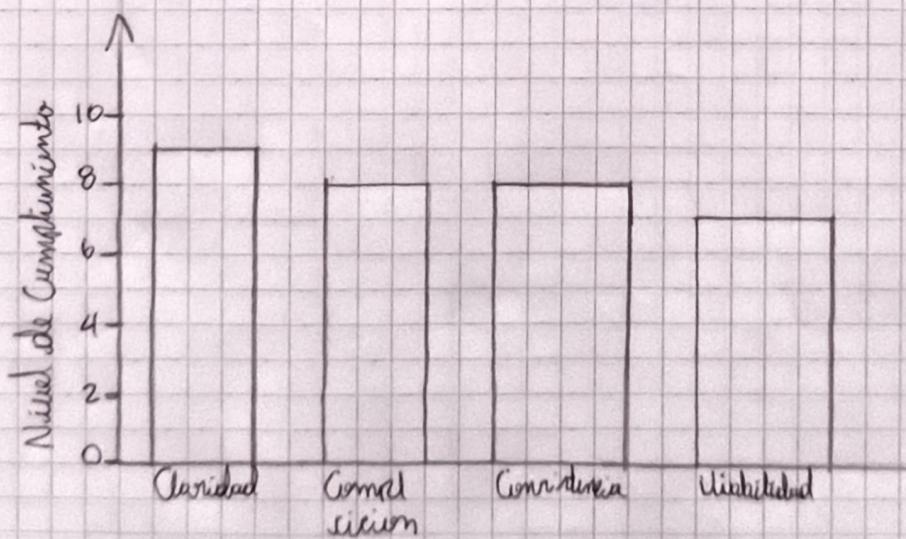
Conclusion

Este capitulo evidencia que un modelo educativo para formular proyectos de investigación debe priorizar una etapa verificable, facilidad de uso de una estructura que da oriente al usuario. La retroalimentación oportuna complementa el avance, mejorando la calidad y efectividad de los proyectos.

Artículo 45 - Calidad en los especificos de requerimientos de software replicado en metodologías ágiles.

El artículo nos habla de la importancia de definir las claridad de los requerimientos en proyectos que replican metodologías ágiles. Se destaca el papel de las historias de usuario, el criterio de aceptación es la satisfacción constante del cliente. Aunque la documentación es mínima, se enfatiza que debe ser clara y verificable para facilitar rebasejar. El artículo propone buenas prácticas como mantener trazabilidad ligera, usar historias en cada sprint y fomentar la comunicación con los usuarios. Concluye que la calidad de los requerimientos en ágil se basa en la precisión y la colaboración, más que extender documentación.

Gráfica:



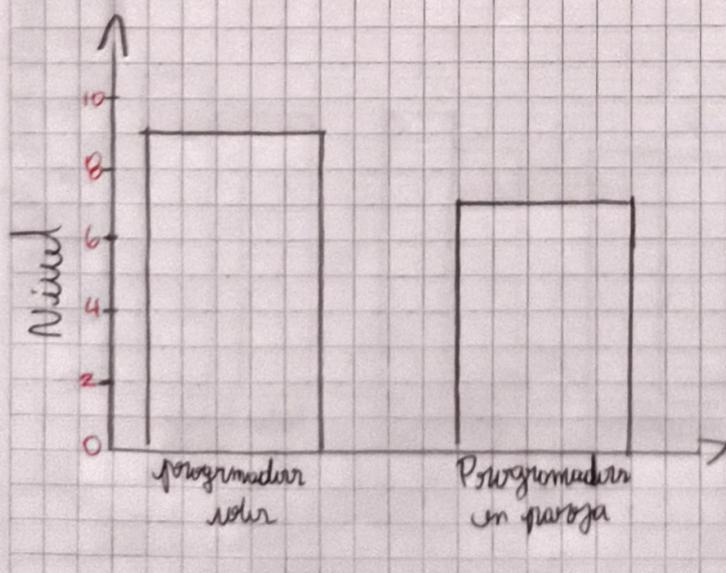
Reflexión:

Este capítulo resalta que, en metodologías ágiles, la claridad de los requerimientos es clave para el éxito del proyecto. Mantener claridad, consistencia, visibilidad y comunicación asegura que el equipo trabaje alineado y que el producto final cumpla con las expectativas del cliente.

Artículo 16 - Estudio de calidad vs eficiencia de un enfoque de desarrollo software orientado con programadores solo vs en pareja

Este estudio compara el desarrollo de programador trabajando bien a solas que lo hacen en parejas dentro de un enfoque secencial. Se evaluaron factores como tiempo de desarrollo, calidad del código y detección de errores. Los resultados muestran que el trabajo en pareja permite identificar fallas más rápidas y producir software más robusto, aunque el inicio puede requerir más tiempo. En contraste, los desarrolladores individuales avanzan rápido, pero tienden a dejar errores sin detectar. El artículo concluye que el trabajo en pareja es más eficiente en términos de calidad, aunque requiere mayor coordinación.

Grafico:



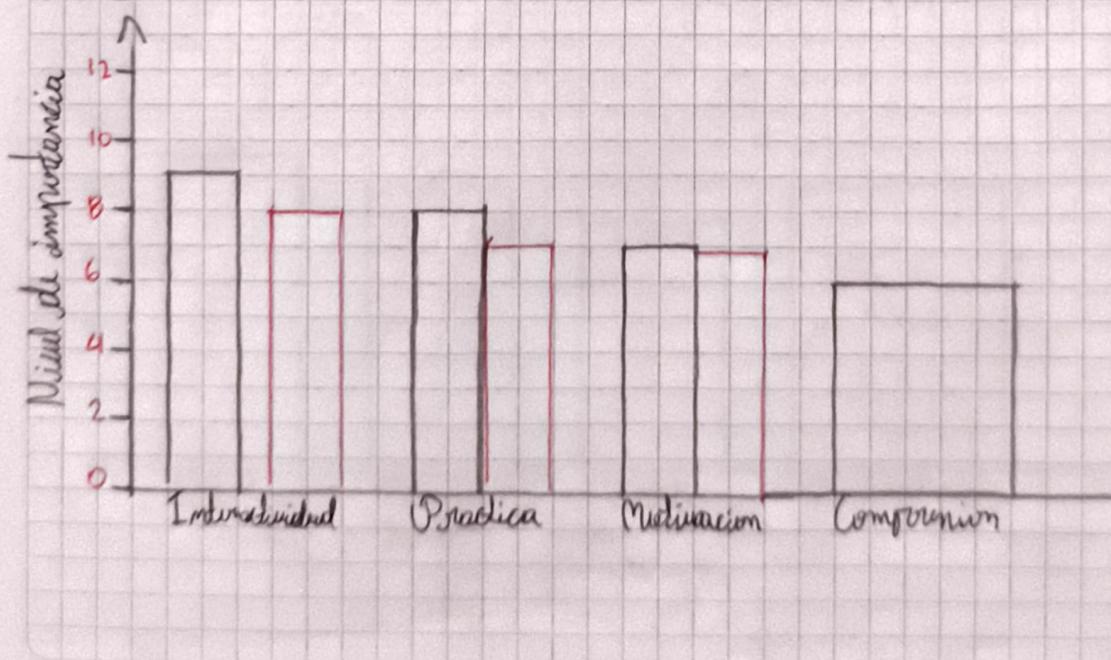
Resumen

Este capítulo muestra que la validad en el software sobrevalorado no genera una mayor calidad frente a un proceso ad-hoc, ya que puede mejorar la eficiencia al reducir complicaciones / ejecuciones innecesarias. Los diferencias entre roles de programadores no son significativas en calidad.

Artículo 13 - Software Educativo como método didáctico en la enseñanza

El artículo habla de software educativo diseñado para mejorar la enseñanza de los tablas de multiplicar en niños de primaria. Basado en un enfoque lúdico, incluye actividades dinámicas, retroalimentación inmediata y recurrir a imágenes y sonidos. La visualización con ilustraciones muestra que el software incrementa la motivación, facilita la comprensión y mejora el desempeño en las tareas posteriores. Se concluye que el uso de herramientas interactivas en el aula puede impulsar el aprendizaje, siempre con un acompañador que lo guíe todo durante.

Gráfico:



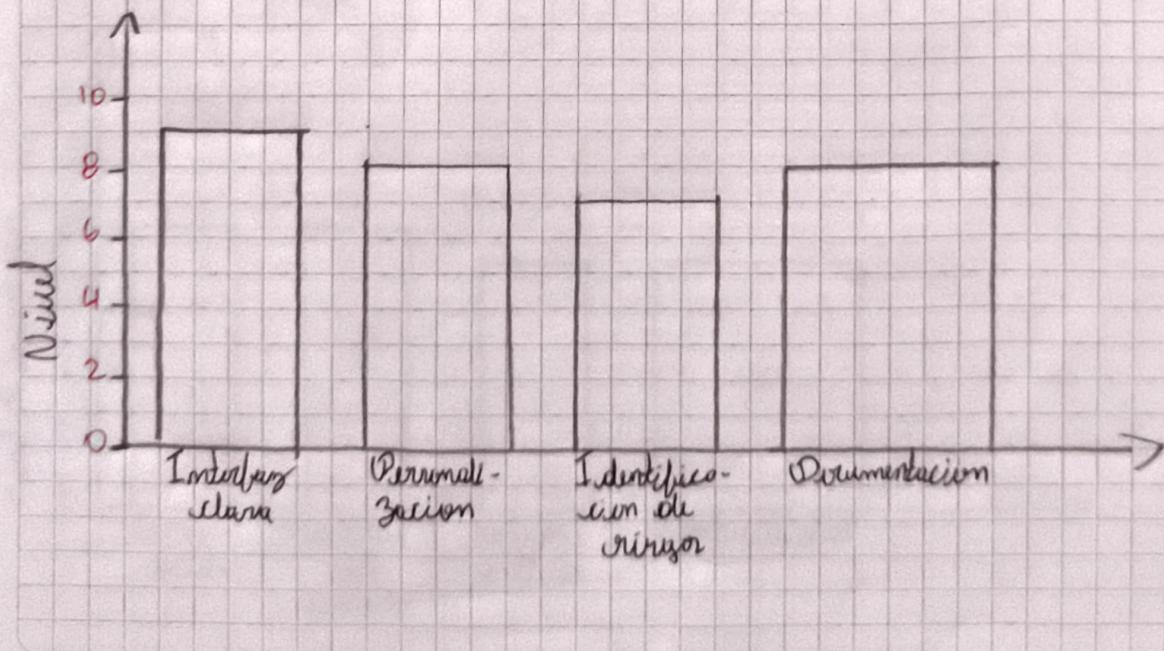
Reflexión:

Este capítulo muestra que el uso del software educativo para enseñar los hábitos de multiplicación potencia el aprendizaje al combinar interactividad, práctica constante y motivación. La comprensión mejora cuando la herramienta es vibrante y fácil de usar, reforzando el aprendizaje de forma rápida y efectiva.

Artículo 48 - Desarrollo de un software web para la generación de planes de gestión de riesgos de software

Este artículo propone una aplicación web que facilita la elaboración de planes de gestión de riesgos en proyectos de software. La herramienta permite registrar riesgos, calcular probabilidad e impacto y generar estrategias de mitigación. También ofrece reportes del seguimiento que resumen la toma de decisiones. En primeros aplicados a proyectos existentes, se evidenció que el sistema fortalece el proceso, mejora la trazabilidad y simplifica la documentación. Se concluye que este tipo de herramientas es útil especialmente en equipos con poca experiencia en gestión de riesgos.

Gráfica:



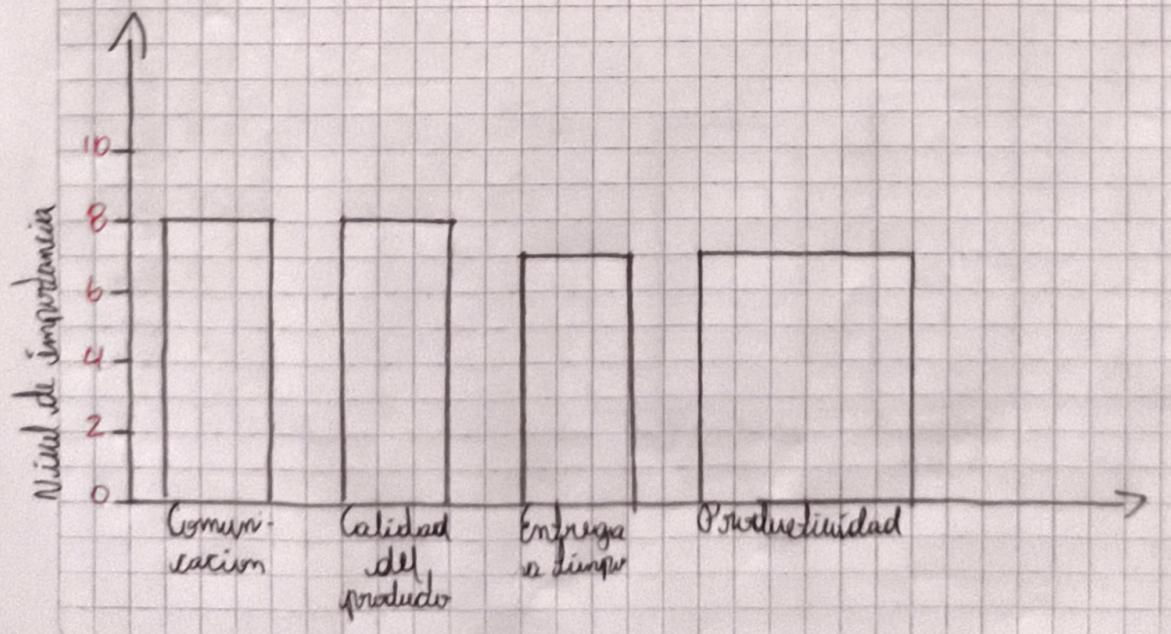
Reflexión:

Este capítulo muestra que un informe debe para la ejecución de riesgo debe priorizar una terminología clara, versiones de personalización y una identificación precisa de riesgos. La documentación completa y actualizada complementa el informe, asegurando que los planes de mitigación sean efectivos y fáciles de implementar.

Artículo 19 - Metodologías mixtas de Desarrollo: un caso de Aplicación medellín - Colombia 2016-2017

Este caso del estudio describe la implementación de metodologías mixtas en un equipo de desarrollo de Medellín. Antes de la aplicación, la empresa enfrentaba retrasos, problemas de comunicación y baja satisfacción de clientes. Con la aplicación de scrum, se mejoró la productividad, la transparencia y la alineación con los mercados del país. Aunque se presentaron retos como la cultura técnica no cambió de alcance, las prácticas de revisión continua y el diseño frecuente ayudaron a mitigarlos. En conclusión, la experiencia demuestra que la dirección y la adaptación del equipo fueron claves para lograr resultados positivos.

Gráfica:



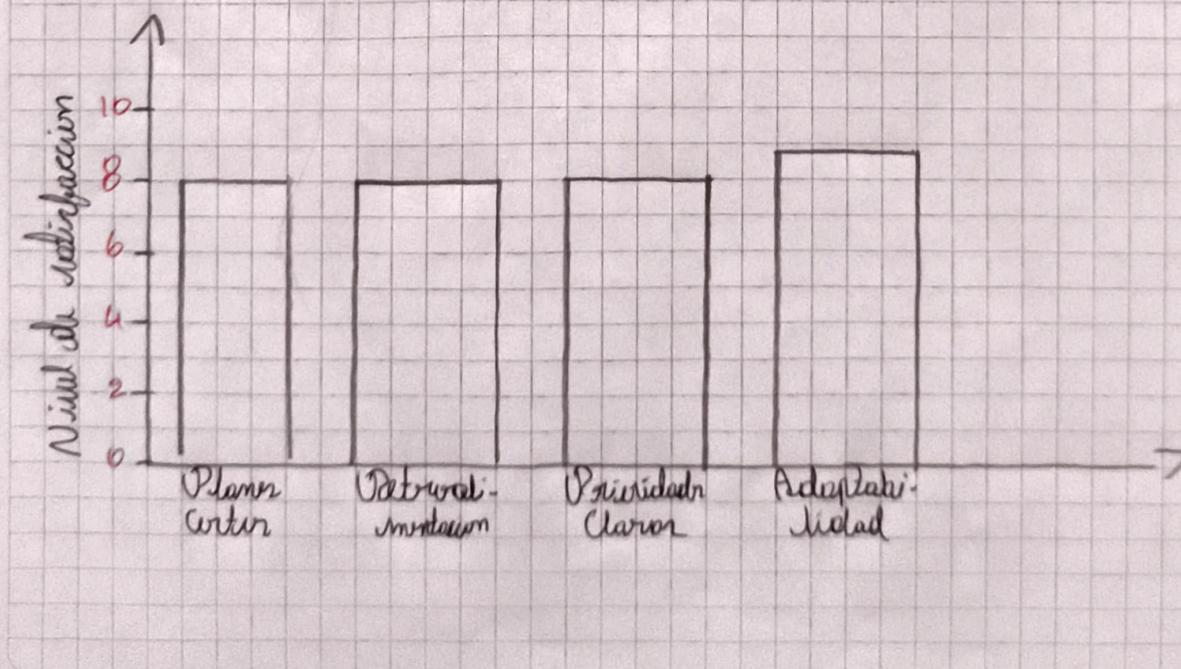
Rebbeck:

Este capitulo muestra que la aplicacion de metodologias agiles en medicina permite mejorar la comunicacion, la calidad del producto, la entrega a tiempo y la productividad. Aunque la importancia de sus factores es alta, el nivel de satisfaccion indica que aun hay espacio para optimizar la implementacion y consolidar la cultura agil en las organizaciones.

Artículo 20 - Metodología iterativa al desarrollo de Software para microempresas

El artículo propone una metodología iterativa adaptada a las condiciones de las microempresas, caracterizada por sucesos limitados y necesidad constante de adaptación. Se plantean ciclos cortos de desarrollo, entregas frecuentes y retroalimentación constante del cliente. La propuesta busca equilibrar simplicidad y flexibilidad, garantizando que los resultados son viables en un plazo reducido. Se concluye que esta metodología es una alternativa viable para microempresas que requieren soluciones tecnológicas rápidas y seguras.

Grafico:



Reflexión

Este capítulo evidencia la metodología iterativa en especialmente útil para microempresas, ya que permite adaptarse rápidamente a cambios, mantener prioridades claras al recibir retroalimentación constante. Esto asegura la eficiencia y asegura que el producto final responda mejor a las necesidades reales del cliente.