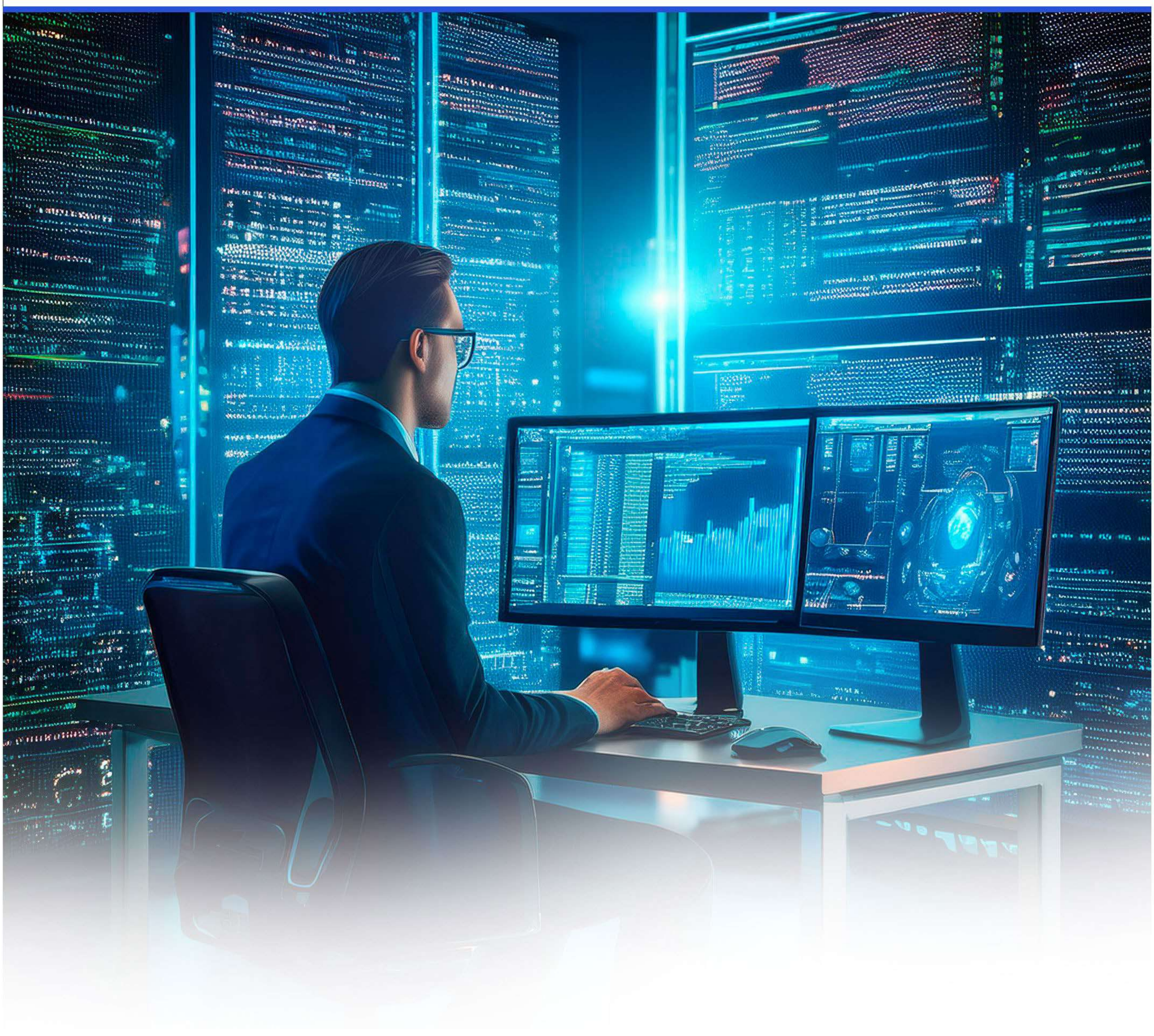


La evolución de las tecnologías de desarrollo de software

Autores:
Cristian Camilo Devia Montero
Albeiro Chicue Garcia
Erick Rodrigo Muñoz Giraldo

Resumen

El artículo aborda la evolución de las tecnologías de desarrollo de software desde sus inicios hasta la actualidad. Explica cómo la crisis del software impulsó el desarrollo de metodologías organizadas para mejorar la calidad y eficiencia del software. Entre estas destacan las metodologías tradicionales como el modelo en cascada, así como enfoques modernos como las metodologías ágiles (Scrum y XP). Estas últimas promueven la colaboración, adaptabilidad y la entrega gradual, lo que ha revolucionado la manera en que los equipos de desarrollo trabajan en proyectos. Asimismo, se menciona cómo la Ley de Moore ha impulsado avances significativos en hardware, posibilitando el desarrollo de software más complejo y potente.

El artículo subraya la importancia de la calidad del software y la gestión de riesgos a través de modelos iterativos y el desarrollo orientado a pruebas, asegurando productos funcionales y adaptables. En conjunto, la evolución del software ha transformado la comunicación, el entretenimiento y el trabajo, integrándose profundamente en la vida cotidiana y marcando un hito en el avance tecnológico.

Palabras claves:
Tecnología computacional, Informática, Hardware, Software, Evolución tecnológica, Ley de Moore, Ingenieros en Sistemas Computacionales, Licenciados en Informática, Metodologías de desarrollo de software (MDS), Calidad del software, Metodologías ágiles, Aplicaciones móviles, Telecomunicaciones, Desarrollo de software, Conecel S.A., Otecel S.A., CLARO, MOVISTAR, Multiusuario.

Abstract

The article addresses the evolution of software development technologies from their inception to the present. It explains how the software crisis drove the development of organized methodologies to improve software quality and efficiency. Among these are traditional methodologies like the waterfall model and modern approaches like agile methodologies (Scrum and XP). The latter emphasize collaboration, adaptability, and incremental delivery, revolutionizing how development teams work on projects. Additionally, Moore's Law is highlighted as a key driver of significant advancements in hardware, enabling the development of more complex and powerful software.

The article underscores the importance of software quality and risk management through iterative models and test-driven development, ensuring functional and adaptable products. Overall, the evolution of software has transformed communication, entertainment, and work, deeply integrating into daily life and marking a milestone in technological advancement.

Keywords:
Computational technology, Informatics, Hardware, Software, Technological evolution, Moore's Law, Computer Systems Engineers, Information Systems Graduates, Software Development Methodologies (SDM), Software quality, Agile methodologies, Mobile applications, Telecommunications, Software development, Conecel S.A., Otecel S.A., CLARO, MOVISTAR, Multi-user.

Introducción

En las últimas cinco décadas, la computación ha avanzado de manera impresionante tanto en el ámbito del hardware, los componentes físicos y tangibles de los sistemas informáticos, como en el del software, el conjunto de programas y procesos que operan estos sistemas. Los progresos en hardware son visibles en dispositivos como computadoras, teléfonos móviles, televisores y cámaras, cada vez más potentes y compactos. Un hito clave en esta evolución es la ley de Moore, postulada por Gordon Moore en 1965, que establece que la capacidad de los procesadores se duplica aproximadamente cada 18-24 meses. Esta predicción ha demostrado ser notablemente precisa y sigue siendo fundamental en el desarrollo de la industria.

A medida que el hardware ha avanzado rápidamente, la demanda de software ha crecido aún más. Cada dispositivo, ya sea una computadora, un teléfono móvil o una máquina especializada, necesita una diversidad de programas para funcionar. Esta necesidad se extiende a numerosos sectores, desde la administración y la educación hasta la medicina y las artes. La demanda de software no solo abarca aplicaciones generales, sino también soluciones adaptadas a necesidades específicas de diversos profesionales como ingenieros, médicos, contadores y abogados.

Desarrollar software de calidad es una tarea compleja que debe cumplir con altos estándares de funcionalidad y facilidad de uso. Los ingenieros en sistemas y desarrolladores de software están encargados de crear aplicaciones que sean no solo eficientes, sino también intuitivas y accesibles para los usuarios. Desde los años 60, la complejidad de esta tarea ha llevado al desarrollo de metodologías de desarrollo de software (MDS), diseñadas para mejorar la eficiencia y la calidad del proceso de creación de software.

Las telecomunicaciones también han experimentado una evolución significativa. Desde el primer teléfono básico, desarrollado por Antonio Meucci en 1876, hasta la moderna tecnología móvil, el progreso ha sido constante. En Ecuador, la tecnología móvil se introdujo en 1993 con la implementación de servicios por parte de Conecel (hoy conocida como CLARO) y Otecel (hoy MOVISTAR). Estas compañías introdujeron la tecnología de segunda generación, lo que permitió una mayor capacidad de comunicación y el inicio de los mensajes de texto SMS.

El desarrollo simultáneo de la electrónica, el hardware y el software ha acortado los ciclos de innovación, permitiendo rápidos avances en la forma en que nos comunicamos y utilizamos la tecnología. La transición de la comunicación por cartas hasta las redes sociales y el internet refleja una verdadera revolución en nuestras capacidades de comunicación, haciendo que los avances previos se consideren obsoletos en un corto período de tiempo.

Objetivo

Las tecnologías de desarrollo de software tienen como objetivo principal facilitar la creación de programas informáticos de manera eficiente y efectiva. Esto se logra a través de métodos y herramientas diseñadas para:

Mejorar la Calidad del Software:

Garantizando que el software sea confiable, funcione correctamente y cumpla con las necesidades de los usuarios.

Aumentar la Productividad:

Ayudando a los equipos de desarrollo a trabajar de manera más rápida y organizada, lo que reduce tiempos y costos.

Adaptarse a los Cambios:

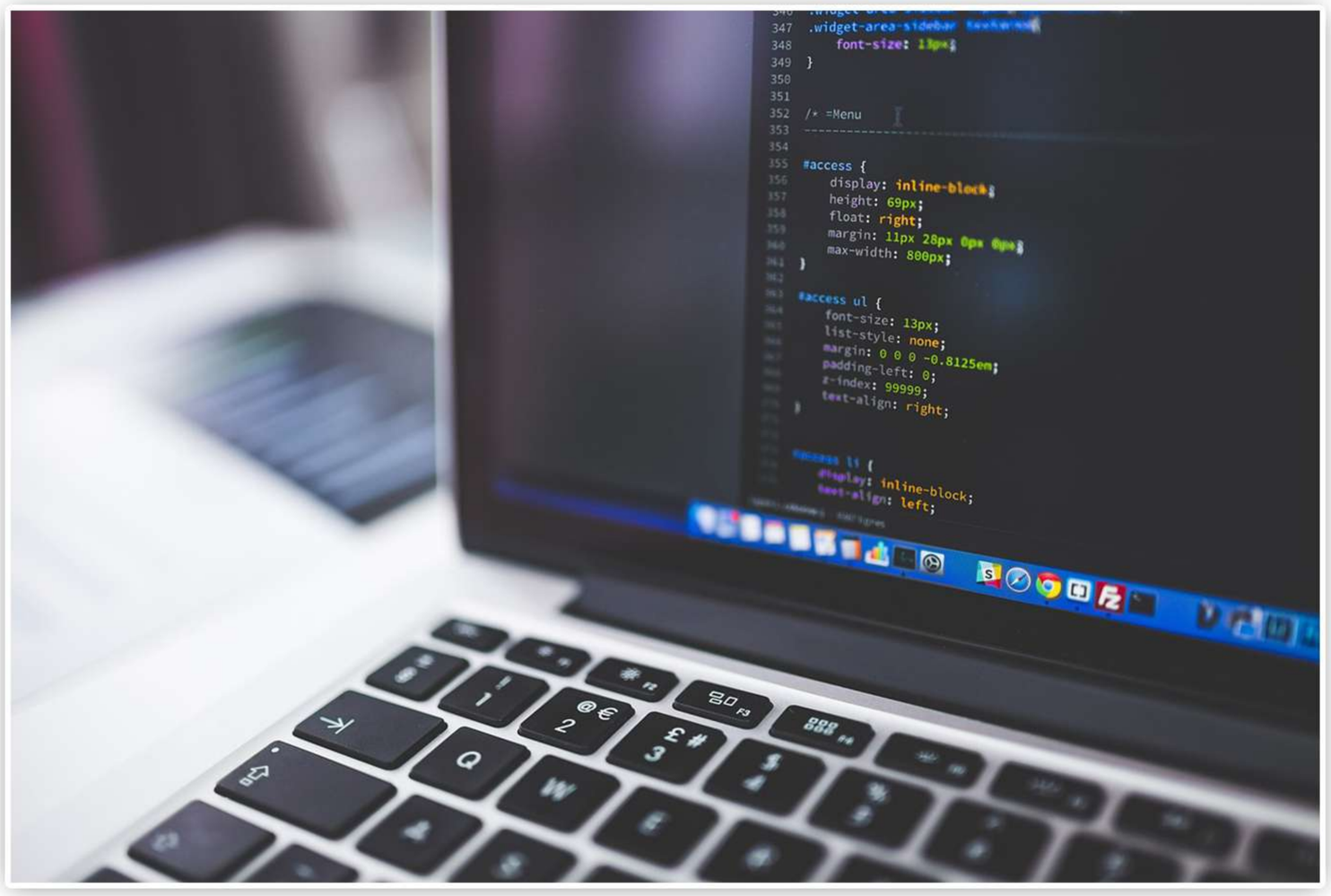
Permitiendo ajustes y actualizaciones durante el proceso de desarrollo para responder a nuevas necesidades o modificaciones de los requisitos.

Facilitar la Colaboración:

Promoviendo la comunicación entre los miembros del equipo y con los clientes para asegurar que todos estén alineados en cuanto a objetivos y avances del proyecto.

Gestionar Riesgos:

Identificando posibles problemas o dificultades antes de que ocurran y planificando cómo manejarlos eficazmente.



Metodología

La metodología propuesta para el desarrollo de aplicaciones para móviles se fundamenta en la experiencia de investigaciones previas en aplicaciones móviles, la evaluación del potencial de éxito para servicios de tercera generación denominada 6 M, la ingeniería de software educativo con modelado orientado por objetos (ISE-OO), y principalmente en los valores de las metodologías ágiles.

De las metodologías ágiles se heredan los conceptos inmersos en los cuatro postulados o manifiesto ágil (Beck et al., 2001).

- Desarrollar software que funciona más que conseguir buena documentación.
- La respuesta ante el cambio es más importante que el seguimiento de un plan.
- Colaboración con el cliente sobre negociación contractual.
- Individuos e interacciones sobre procesos y herramientas.

De la 6 M's se extrae la concepción de que las aplicaciones móviles deben garantizar el cumplimiento de las necesidades de los usuarios y al mismo tiempo generen ingresos. La 6 M's debe su nombre a los seis atributos que se miden para evaluar el éxito del servicio propuesto: Movement (Movimiento), Moment (Momento), Me (Yo), Multi-user

Tipos de metodología

Metodología XP

Es una metodología ágil "centrada en fomentar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen entorno de trabajo. XP se basa en una retroalimentación continua entre el cliente y el equipo de desarrollo, una comunicación clara entre todos los antes mencionados, permitiendo simplicidad en las soluciones de software implementadas y valor para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico"

Metodología Scrum

Esta metodología ágil tiene características compartidas con XP en su enfoque de la colaboración entre el equipo de trabajo y el cliente, pero su propósito principal es la realización de las actividades de gestión de proyectos
"Scrum es un marco de trabajo de procesos que ha sido usado para gestionar el trabajo en productos complejos desde principios de los an□os 90. Scrum no es un proceso, una técnica o método definitivo. En lugar de eso, es un marco de trabajo dentro del cual se pueden emplear varios procesos y técnicas. Muestra la eficacia relativa de las técnicas de gestión de producto y las técnicas de trabajo de modo que podamos mejorar continuamente el producto, el equipo y el entorno de trabajo. El marco de trabajo Scrum consiste en los equipos Scrum y sus roles, eventos, artefactos y reglas asociadas. Cada componente dentro del marco de trabajo sirve a un propósito específico y es esencial para el éxito de Scrum y para su uso"

Metodologías de desarrollo de software tradicionales

Este enfoque está condicionado a un proceso lineal el cual requiere que para empezar una fase del desarrollo de software, debe haberse concluido una anterior, este es un modelo orientado a resultados en donde el usuario tendrá una participación al inicio y al fin del desarrollo del producto de software.
Su proceso, basado en definir antes que diseñar, diseñar antes que codificar, codificar antes que realizar pruebas y realizar pruebas antes de implementar; permite tener resultados una vez que el producto esté terminado, y si existieran errores, implica un rediseño y con esto costos y tiempo de desarrollo.

Metodología RUP

De acuerdo con Noordeloos et al. (2012), la metodología RUP (por sus siglas en inglés o Proceso de Desarrollo Unificado) proporciona disciplinas en las cuales se encuentran artefactos con lo cual se podrá contar con guías para poder documentar e implementar de una manera fácil y eficiente con el propósito de asignar tareas y responsabilidades dentro de una organización del desarrollo, todo esto dentro de las respectivas fases con las cuales cuenta. "Su objetivo es asegurar que el producto final de software sea de alta calidad y que permita resolver las necesidades de los usuarios dentro de un presupuesto y tiempo establecidos"

Metodologías de desarrollo para aplicaciones móviles

En el sitio de RedHat (s.f) se expresa que "en esta era de tecnología móvil y constante actividad, las personas que trabajan en campo esperan que el nivel de conexión que les ofrece su teléfono inteligente sea el mismo que el de los compañeros que trabajan en un escritorio", por lo que objetivo de crear aplicaciones móviles para empresas es posibilitar dicha conectividad y, a su vez, cumplir con los requisitos de confiabilidad y seguridad de una compañía de alto nivel. "Las empresas buscan que sus aplicaciones estén disponibles en los dispositivos móviles sin que ello implique una cantidad excesiva de tiempo de desarrollo. Hay una variedad de estrategias de desarrollo para diseñar aplicaciones móviles, desde soluciones móviles y pre empaquetadas que no requieren código hasta soluciones completamente personalizadas y plataformas de desarrollo integradas en los dispositivos móviles"

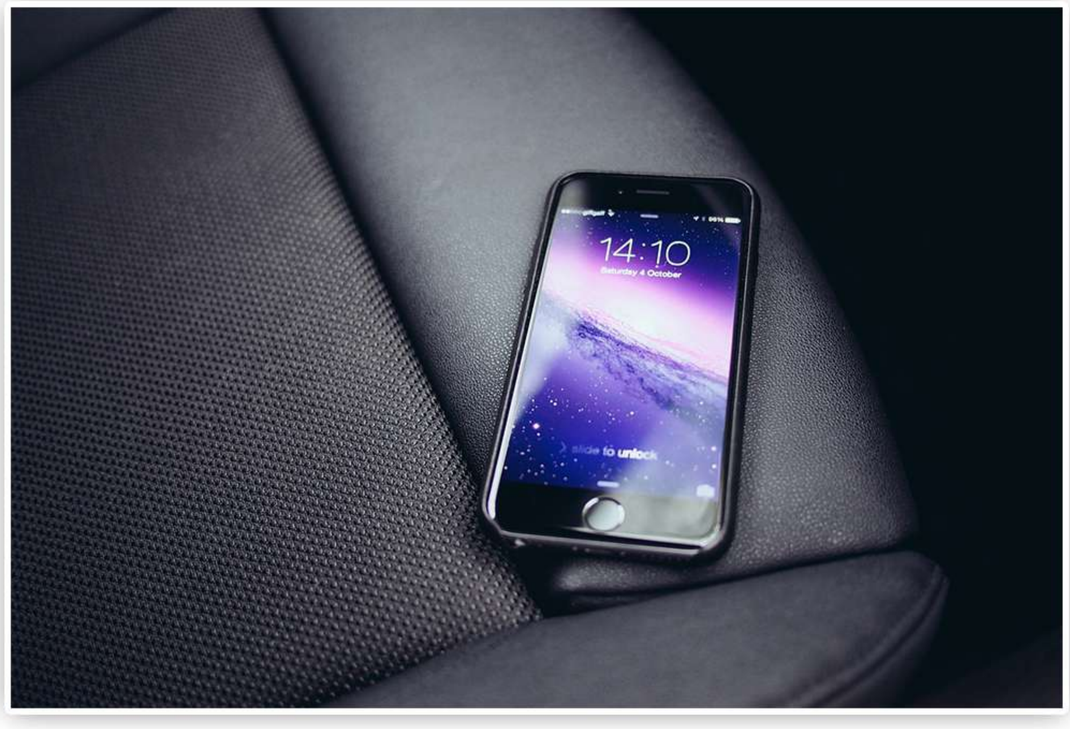
Metodología Mobile-D

El autor Sangama Oñate (2020) sostiene que el objetivo de esta es "conseguir ciclos de desarrollos muy rápidos en equipos muy pequeños (de no más de diez desarrolladores) trabajando en un mismo espacio físico. Según este método, trabajando de esa manera se deben conseguir productos totalmente funcionales en menos de diez semanas. Se trata de método basado en soluciones conocidas y consolidadas: Extreme Programming (XP), Crystal Methodologies y Rational Unified Process (RUP), XP para las prácticas de desarrollo, Crystal para escalar los métodos y RUP como base en el diseño del ciclo de vida"

Metodología Mobile espiral

Esta propuesta metodológica utiliza el modelo de desarrollo en espiral como base y permite incorporar procesos que evalúan la usabilidad y tomando lo especial de las metodologías ágiles, que es la participación del usuario en todos los procesos del ciclo de vida de diseño, y así garantizar un diseño centrado en el usuario. Este proceso permite a los desarrolladores de aplicaciones móviles detallar todos los criterios de usabilidad de la aplicación, tomando en cuenta dos pasos:

- El primero es identificar a los usuarios, las tareas y contextos en los que se utilizará la aplicación móvil,
- El segundo es dar prioridad a los atributos de usabilidad, identificar cuáles son los más relevantes, y definir para cada uno un conjunto de métricas para verificar el grado en que se cumplen en el producto



Desarrollo orientado a pruebas

Amaya (2015) sostiene que este condiciona la mentalidad de los desarrolladores dando una guía a través del desarrollo y se enfoca en la calidad del producto final de software. "El desarrollo mantiene un exhaustivo juego de pruebas por partes del programador, ninguna parte del código pasa a producción a no ser que pase sus pruebas asociadas, esta forma de desarrollar cambia drásticamente la mentalidad de cualquier equipo, generalmente agilizando los resultados y aumentando la calidad del producto de software", puntualiza. "El desarrollo orientado a las pruebas lleva más de 10 años en la industria del software, y su principal aplicación en la actualidad es en proyectos con metodologías ágiles, tal como XP, Agile o Scrum, pero se puede ver que es posible aplicarlo en otros proyectos, y no solo en nuevos" (Vaca et al., 2013).

Indicadores KPI

El ciclo de vida del software (por sus siglas en inglés SDLC, Systems Development Life Cycle) son etapas o fases que se debe pasar para desarrollar un software, la aplicación de un modelo dependerá de varios factores internos y externos y de cómo el grupo de desarrollo decidan implementar en la realización de los proyectos de software, podemos citar a varios autores que definen el SDLC según su nivel de aceptación (sin ningún orden en especial):

•Fase 1, análisis. Fase 2, diseño. Fase 3, codificación. Fase 4, prueba. Fase 5, mantenimiento, (Álvarez, 2007).
•Fase 1, requerimientos. Fase 2, análisis / diseño. Fase 3, construcción. Fase 4, pruebas. Fase 5, producción/mantenimiento, (Fábregas, 1991).
•Fase 1, investigación preliminar. Fase 2, determinación de requerimientos. Fase 3, desarrollo del software. Fase 4, prueba del sistema. Fase 5, implantación y evaluación (Senn et al., 1992)

Resultados

METODOLOGÍA	RESUMEN
XP	Es una metodología ágil centrada en fomentar las relaciones interpersonales como clave para el éxito en desarrollo de software.
Scrum	Su enfoque es la colaboración entre el equipo de trabajo y el cliente, pero su propósito principal es la realización de las actividades de gestión de proyectos.
RUP	Proporciona disciplinas en las cuales se encuentran artefactos con lo cual se podrá contar con guías para poder documentar e implementar de una manera fácil y eficiente.
Mobile-D	Consigue ciclos de desarrollos muy rápidos en equipos muy pequeños.
Mobile Espiral	Utiliza el modelo de desarrollo en espiral como base y permite incorporar procesos que evalúan la usabilidad tomando lo especial de las metodologías ágiles
Orientado a pruebas	El desarrollo mantiene un exhaustivo juego de pruebas por partes del programador

Preguntas y Respuestas sobre Metodologías de Desarrollo de Software

1. ¿Cuáles son las diferentes formas de desarrollar software?

Existen principalmente cuatro maneras de hacerlo:

Modelos Secuenciales: Como el Modelo en Cascada, donde se planea todo desde el inicio sin muchos cambios durante el proceso.

Modelos Iterativos y Evolutivos: Como el Incremental y los Prototipos, donde se va mejorando el software paso a paso según se aprende de cada versión.

Metodologías Ágiles: Como Scrum y Kanban, que se enfocan en adaptarse rápidamente a los cambios y entregar versiones útiles del software de manera continua.

Modelos de Minimización de Desarrollo: Donde se reutilizan partes del software existente para hacer nuevos programas más rápido y con menos esfuerzo.

2. ¿Qué significa "metodologías de desarrollo de software"?

Son formas organizadas y sistemáticas de planificar y hacer software. Cada metodología tiene sus propias reglas y métodos para manejar proyectos y asegurar que el software funcione bien.

3. ¿Quién hace el software?

Profesionales como ingenieros en sistemas, licenciados en informática y otros especialistas. Su trabajo es crear software que sea fácil de usar y haga lo que los usuarios necesitan.

4. ¿Por qué es importante elegir bien cómo hacer el software?

Elegir la forma correcta de hacer el software es clave para hacerlo eficientemente. Si se elige mal, se pueden perder tareas importantes o hacerlas en el orden incorrecto, lo que puede hacer el proyecto menos eficaz.

5. ¿Cómo funciona el modelo en cascada?

En este modelo, se planea todo el proyecto desde el principio y se sigue un paso a paso sin muchos cambios. Es bueno para proyectos donde los requisitos son claros y no cambian mucho.

6. ¿Qué son los modelos evolutivos y cómo se usan?

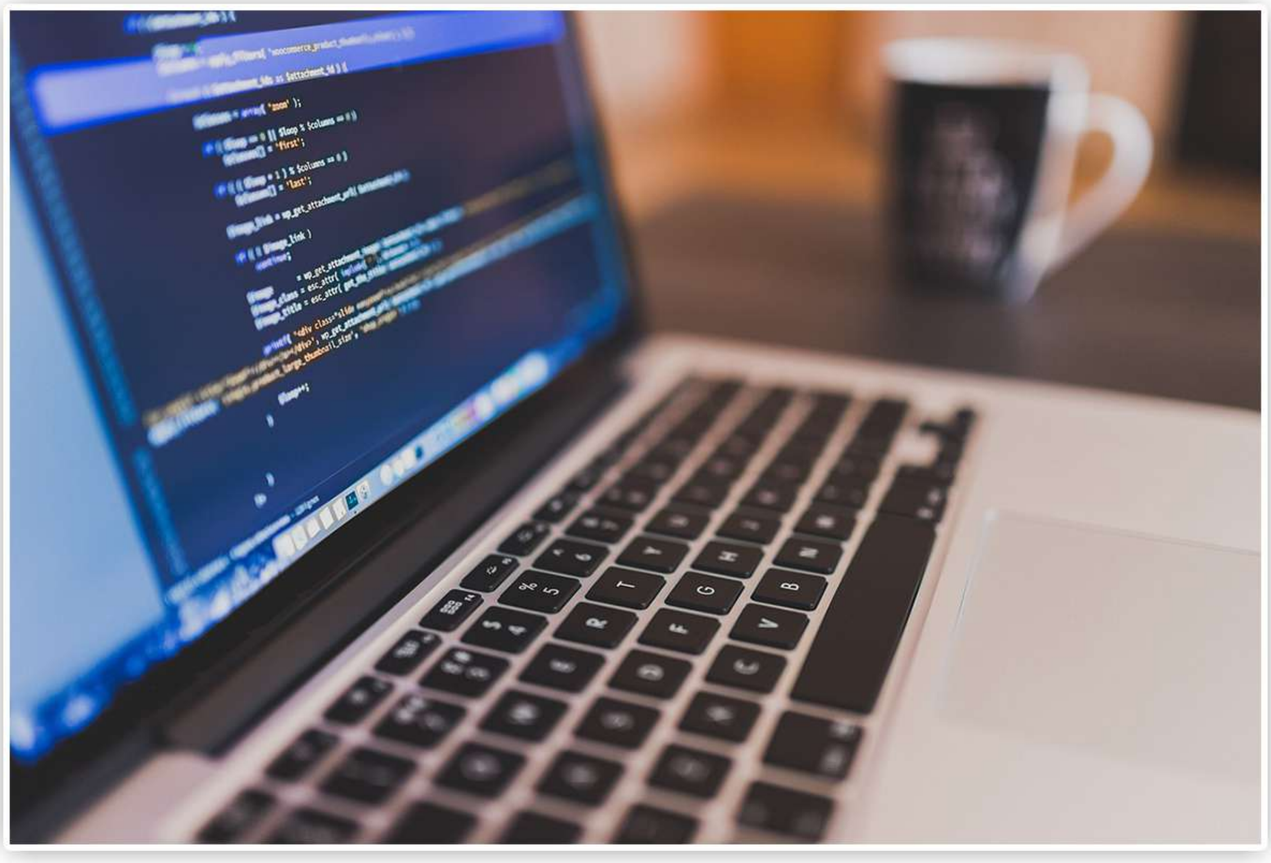
Estos modelos permiten ajustar y mejorar el software en cada ciclo de trabajo. Se revisan los requisitos y se hacen cambios según se avanza, pero puede ser complicado manejar grandes cambios si no se hace bien.

7. ¿En qué consisten los modelos de minimización de desarrollo?

Aquí se reutilizan partes de software existente para hacer nuevo software más rápido y con menos esfuerzo. Es útil cuando partes del software ya existen y se pueden usar de nuevo.

8. ¿Qué hacen las metodologías ágiles?

Estas metodologías se usan cuando los requisitos del proyecto pueden cambiar. Entregan versiones tempranas del software que cumplen con lo básico que el cliente necesita, y van mejorando según se van aprendiendo más cosas sobre lo que se necesita.



Conclusiones

En conclusión el desarrollo de software ha evolucionado enormemente a lo largo de las décadas recientes, impulsado por avances en hardware y nuevas formas de trabajar. Desde los primeros días de la informática hasta hoy, hemos pasado de programas simples a sistemas complejos que sostienen nuestra vida diaria digital. La introducción de metodologías como Scrum y Extreme Programming ha revolucionado cómo los equipos desarrollan software. Estas metodologías enfatizan la colaboración constante, la entrega gradual y la adaptación rápida a los cambios, permitiendo a las empresas ser más ágiles y responder mejor a las necesidades del mercado. Además, la ley de Moore, que predijo la mejora constante en la capacidad de procesamiento, ha hecho posible dispositivos más poderosos y aplicaciones más sofisticadas. Esto ha cambiado nuestra forma de comunicarnos, trabajar y entretenernos, todo impulsado por software que evoluciona a un ritmo vertiginoso.

Referencias Bibliográficas

La anterior información fue sacada a través de Google académico esta información fue sacada en mayor parte de archivos pdf a continuación los Links:

https://www.ecorfan.org/bolivia/researchjournals/Tecnologia_e_innovacion/vol2num5/Tecnologia_e_Innovacion_Vol2_Num5_6.pdf

https://www.eumed.net/rev/cccss/2016/04/5G.zip

https://www.redalyc.org/pdf/373/37326902005.pdf

https://www.researchgate.net/profile/Maryory-Urdaneta-2/publication/353826882_Sistema_de_seguimiento_de_requerimientos_eventos_e_incidentes_para_los_clientes_de_la_empresa_TELCONET_SA_en_la_ciudad_de_Quito/links/6113d39b1e95fe241ac5c3d5/Sistema-de-seguimiento-de-requerimientos-eventos-e-incidentes-para-los-clientes-de-la-empresa-TELCONET-SA-en-la-ciudad-de-Quito.pdf#page=251

https://www.researchgate.net/profile/Pere-Tumbas/publication/267711880_A_Comparative_Overview_of_the_Evolution_of_Software_Development_Models/links/546310e90cf2c0c6aec1c6eb/A-Comparative-Overview-of-the-Evolution-of-Software-Development-Models.pdf

https://ieeexplore.ieee.org/document/6717065

https://en.wikipedia.org/wiki/Software_development_process

https://www.ibm.com/topics/software-development