METODOLOGIAS ÁGILES

Febrero 2025

Docente
Julian Alexis Balanta Mera

Equipo:
Carlos Lozada
Oscar zuñiga
Junior Duvan Mesa
Luis Angel Caracas
Juan Camilo Gil

Asignatura
Gestión De Proyectos de Software

Universidad del Valle. Sede Regional Norte del Cauca
Facultad de Ingeniería
Tecnología en el Desarrollo de Software

Santander de Quilichao
2025

INTRODUCCIÓN

¿Para el desarrollo de software es necesario emplear alguna metodología que ayude a su construcción? o ¿es mejor desarrollar software basado en las propias habilidades? Lo primero que se debe considerar es que un proyecto se realiza de manera grupal. En este trabajo se muestran las metodologías que son mayormente usadas y han ido evolucionando hasta la actualidad, además se menciona de manera general las etapas del desarrollo del software, la intención no es volverse experto en estos temas con un conocimiento superficial, la idea principal es introducir y presentar como guía todo el proceso que envuelve la construcción de un sistema de información, de manera que teniendo claridad sobre los principales puntos del desarrollo de software (ciclo de vida) incluyendo su metodología, se pueda profundizar de manera personal aquel factor que cada quien considere apropiado, desde el análisis y todo el proceso de documentación, el diseño tanto de la base de datos como del sistema, hasta el desarrollo o construcción del software ya sea desde el Back-End o Front-End.

Metodologías Tradicionales

MODELO CASCADA:

El modelo cascada es un enfoque bastante tradicional para desarrollar software, donde cada fase del proyecto se realiza de forma secuencial. Es como un camino que sigues paso a paso: primero haces una cosa, luego pasas a la siguiente, y no puedes saltarte ninguna. Todo comienza con la recopilación de los requisitos, es decir, entender qué necesita el sistema.

Después, se diseña cómo será el software, cómo se organizan los componentes, y luego se pasa a escribir el código.

Una vez que el software está listo, se realizan pruebas para asegurarse de que todo funcione bien, y si todo está correcto, se pone en marcha. Pero el trabajo no termina ahí: con el tiempo, si surgen problemas o es necesario mejorar algo, se hace mejora o se estudia un plan mejor para satisfacer las necesidades.

¿Dónde lo utilizamos?

Es bastante útil cuando los requisitos del proyecto son claros desde el principio, ya que es muy estructurado y permite seguir un camino claro. Sin embargo, si durante el desarrollo aparecen cambios o nuevas ideas, puede ser difícil adaptarse, lo que hace menos flexible el trabajo.

Metodologías Ágiles

- Ejemplos de este método son: Scrum, Kanban, XP. Se puede decir que es una combinación de las metodologías tradicionales. La principal característica de este método es que son flexibles en cuanto a los requerimientos, es decir, que cuando no están claramente definidos o se considera que pueden cambiar, lo más recomendable es utilizar metodología ágil.
- Su ventaja es la flexibilidad en cuanto a los requerimientos.
- Su desventaja radica en que el equipo de desarrollo debe contar con sólidas capacidades de disciplina. Es entendible, debido a que la implementación de nuevos requerimientos es un trabajo arduo.

Scrum:

Una de las principales razones para hacer uso de la metodología scrum, es que agiliza el proceso de desarrollos complejos mediante entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. como afirma Canos s.f

"Está especialmente indicada para proyectos con un rápido cambio de requisitos". Lo que indica eficiencia en el desarrollo, al involucrar sólo parte del proyecto y no su conjunto.

El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. uno de los puntos claves importantes es la comunicación con el cliente, a quién se muestran avances finalizada cada iteración, la segunda característica
importante son las reuniones a lo largo proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo, lo que es fundamental para coordinar tareas y perspectivas de los avances. (Canós et.al)

Define un proceso empírico, iterativo e incremental de desarrollo que intenta obtener ventajas respecto a los procesos definidos (cascada, espiral, prototipos, etc.) mediante la aceptación de la naturaleza caótica del desarrollo de software, y la utilización de prácticas tendientes a manejar la impredictibilidad y el riesgo a niveles aceptables.

Scrum es un método iterativo e incremental que enfatiza prácticas y valores de project management por sobre las demás disciplinas del desarrollo. Al principio del proyecto se define el Product Backlog que contiene todos los requerimientos funcionales y no funcionales que deberá satisfacer el sistema a construir.

El Product Backlog se define durante reuniones de planeamiento con los Stakeholders. A partir de ahí se definen los sprints en los que se irá evolucionando la aplicación. Dentro de cada sprint el Scrum Master lleva a cabo la gestión de la interacción convocando diariamente al Scrum Daily Meeting que representa una reunión de avance diaria con el propósito de tener retroalimentación sobre las tareas y obstáculos que se presentan.

Kanban:

La metodología Kanban es una herramienta visual que permite gestionar las tareas y los elementos de un proyecto a través de una tabla. En ella, se define claramente el rol de cada miembro del equipo y las tareas que deben desarrollar a lo largo del proyecto, facilitando el seguimiento y la organización del trabajo.

Scrumban:

La metodología Scrumban es una metodología que combina las 2 metodologías anteriormente mencionadas y se utiliza para equipos ágiles que buscan un enfoque flexible y eficiente para gestionar sus proyectos.

En lugar de trabajar en spring fijos como en Scrum, Scrumban permite un flujo de trabajo más continuo, similar a Kanban, donde las tareas se inician solo cuando hay capacidad para hacerlo. Se usa un tablero kanban para visualizar el progreso de las tareas, desde "Por hacer" hasta "Hecho", lo que facilita ver en qué etapa está cada tarea.

Es ideal para equipos que necesitan adaptarse rápidamente a cambios y no tienen un cronograma rígido. Aunque no sigue los ciclos fijos de Scrum, Scrumban permite reuniones de revisión y retrospectivas para mejorar constantemente el proceso y optimizar el trabajo.

En resumen, Scrumban es perfecto para quienes quieren la estructura de Scrum pero con la flexibilidad y fluidez de Kanban.

PRINCE2

Es una metodología de gestión de proyectos en donde se planifica la estructura de cada fase de desarrollo del proyecto de manera anticipada con el propósito de que todos los implicados en el proyecto tenga todo claro antes de que comience la fase de desarrollo y los problemas que surgen se corregirán al ir finalizando el proyecto

PRINCE2 cuenta con 7 fases:

1. Puesta en marcha

Al principio se desarrolla un análisis de la viabilidad y necesidad del proyecto. En el caso de que los altos mandos consideren que es estrictamente necesario llevarlo a cabo, se crea un resumen más detallado del mismo.

2. Dirección del proyecto

El comité encargado de revisar dicho resumen, analiza cada uno de los pasos indispensables para la ejecución e inicio del proyecto. Se definen las responsabilidades y roles del equipo y del gestor del proyecto.

3. Inicio

El proyecto da inicio cuando uno de los gerentes del proyecto crea un plan o documento de inicio de proyecto. Este documento incluirá datos como el coste, alcance, riesgos, plazos, beneficios o resultados.

4. Control de una fase

El gestor del proyecto se encarga de crear cada una de las fases que deberá ser monitorizada y controlada de forma individual.

5. Gestión de la entrega de productos

En esta fase compararemos los progresos del proyecto con los objetivos iniciales del mismo.

6. Gestión de los límites de las fases

Una vez se han analizado los progresos de cada fase y se aprueban los resultados obtenidos, se aprobará el comienzo de la siguiente fase.

7. Cierre del proyecto

Una vez se revise y verifique que los entregables de cada una de las fases se encuentran dentro de los objetivos marcados, se acaban de atar los cabos sueltos restantes como pueden ser documentos o trámites y se cierra el proyecto.

Six sigma

El método Seis Sigma es una filosofía que apareció en los años ochenta gracias al ingeniero Mikel Harry, a través de la evaluación y análisis de la variación de los procesos en la empresa Motorola. Fue la primera empresa en implantar esta metodología como estrategia de mercado y de mejoramiento de la calidad. Debido a la globalización, las empresas del sector industrial y comercial empezaron a desarrollar técnicas para optimizar los procesos y

mejorar su competitividad y productividad. Esta metodología también se enfoca en la mejora continua.

Seis Sigma es una metodología compuesta por cinco fases: Definir, Medir, Analizar, Mejorar y Controlar. Representa el número de desviaciones estándar obtenidas a la salida del proceso. Su objetivo es aumentar la capacidad de los procesos, de tal forma que estos generen los mínimos defectos por millón de unidades producidas. Estos defectos deben ser imperceptibles para el cliente.

Esta metodología propone dos campos de aplicación: implementación de un proyecto existente o la creación de un nuevo proyecto, producto o servicio. Estos campos se centran en la reducción de defectos, fallos y no conformidades tratando de conseguir un valor cercano a 0.

Principios de seis sigma

1. Enfoque al cliente.
2. Centrado en los procesos
3. Metodología para la realización de proyectos.
4. Estructura organizacional.
5. Lucha contra la variación.

Fases de seis sigma

1. **Fase de definición:** se identifican los proyectos Seis Sigma que deben ser evaluados por la dirección para evitar la infrautilización de recursos, para así asignar la prioridad necesaria para cada proyecto.
2. **Fase de medición:** consiste en la caracterización del proceso identificando los requisitos clave de los clientes, las características clave del producto y los parámetros que afectan al funcionamiento del proceso y a las características clave. Es donde se define el sistema de medida y se mide la capacidad del proceso.
3. **Fase de análisis:** se analizan los datos actuales e históricos. Se desarrollan hipótesis sobre posibles relaciones causa-efecto mediante el uso de herramientas estadísticas.

4. **Fase de mejora:** se determina la relación causa-efecto para predecir, mejorar y optimizar el funcionamiento del proceso.
5. **Fase de control:** se diseñan y documentan los controles necesarios para asegurar que el sistema implantado se mantenga en el tiempo.

Programación Extrema (XP)

Se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. (Maida et.al)

Uno de los puntos fuertes de la metodología XP, es que considera como natural realizar cambios en los requisitos sobre la marcha, ya que pueden surgir nuevas novedades. por lo que consideran que es mejor adaptarse a esos cambios y cubrirlos con coraje que tratar de definir los requerimientos desde el inicio (Maida et.al)
Se considera oportuno incluir la metodología XP para el desarrollo de un proyecto cuando los requerimientos no presentan claridad o se encuentran propensos a cambiar. Maida concisa la idea de la siguiente manera: “XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico”.

La característica principal y fundamental es el trabajo en equipo donde Aguirre 2020 afirma que XP: "Se centra en potenciar las relaciones interpersonales como clave del éxito del desarrollo". Además se integra al cliente como parte del equipo.

ETAPAS DEL DESARROLLO DE SOFTWARE

Conocido como ciclo de vida, estas etapas son: definición de requerimientos, análisis, diseño, construcción.

FASE DE REQUERIMIENTOS

Las tareas principales que se realizan en esta fase son: planeación, extracción, análisis, especificación, validación. A continuación, se mencionará en qué consiste cada una de ellas:

Planeación: lo primero que se realiza es una recopilación de información, que por lo general es por medio de una entrevista con el cliente, para determinar el alcance, objetivo y propósito del software a desarrollar.

Extracción: en la recopilación de información se deben emplear usando técnicas para su correcta interpretación (UML, Casos de Uso, Diagrama de Procesos, etc.)

Análisis: una vez recogida la información se debe analizar. Las técnicas empleadas para ello son: matriz de requerimientos, matriz de trazabilidad, matriz CRUD.

Especificación: la especificación de los requerimientos del software es considerado un documento final de la fase de requerimientos, en el que se detallan los requerimientos Funcionales y los No Funcionales.

Validación: al tener los requerimientos establecidos y analizados, el líder del equipo de desarrollo se reúne con el cliente para validar los mismos. Una vez validada la información se procede a firmar el contrato, en el que se comprometen las dos partes, los desarrolladores a dar cabalidad a los requerimientos del sistema en el tiempo establecido y el cliente a no agregar más requerimientos de los presentes.

FASE DE ANÁLISIS

Después de firmar el contrato se inicia con la fase de análisis, en esta etapa se revisa la documentación que contiene la información de los requerimientos especialmente el diagrama UML para plantear la metodología que mejor se acomode al desarrollo del software.

Las tareas que se realizan en esta fase son: estudio del entorno tecnológico, elección de arquitectura de desarrollo, diagrama de análisis del sistema.

Estudio del entorno tecnológico: esta tarea consiste en realizar un inventario sobre el cómputo con el que se cuenta para realizar el desarrollo del software.

Elección de la arquitectura de desarrollo: según el documento del SENA Guía para el Desarrollo de Sistemas de Información al respecto dice:

> “Se debe elegir el ambiente (Web, Windows, Consola, Móvil); el sistema manejador de bases de datos (Robusto, Liviano, de Servidor, de Escritorio, Libre, Gratuito, Comercial); el lenguaje de programación (Estructurado, Orientado a Objetos, Libre, Gratuito, Comercial, Orientado a la Web, del lado del Cliente, del lado del Servidor). Pag.9

Ejemplo:

En un ambiente Windows, cada computador debe instalar el software para hacer uso del mismo. Los lenguajes de programación recomendados a usar son: .Net, Java Swing, Delphi entre otros.

En un ambiente web, donde los sistemas están alojados en servidores Web, los lenguajes de programación a usar son: PHP, JSP, ASP.NET, JavaScript, HTML5, CSS3…

En ambiente consola los lenguajes a utilizar son: C, C++, QBasic, Turbo Pascal, .NET, Java.

En ambiente móvil los lenguajes de programación a utilizar son: Java JME, .NET Compact Framework, Windows Phone, Android WML, HTML5.

Tener claridad en cuanto al ambiente de ejecución puede facilitar las tecnologías o herramientas correctas que se deben usar.

Diagrama de análisis del sistema

Son usados para representar de manera gráfica el alcance de un sistema de información. Existen diferentes tipos de diagramas como son:

Diagrama de contexto que permite conocer el alcance del sistema.

Diagrama de flujo que da una visión sobre el comportamiento del sistema.

Diagrama casos de uso que representan la funcionalidad del sistema.

Diagrama entidad relación que representa los datos que el sistema obtiene.

Diagrama de clases que representan aquellos datos y comportamientos que van a formar parte del sistema.

Diagrama de secuencia que representa la interacción entre diferentes funcionalidades del sistema.

Por lo general esta fase de análisis no recibe mayor importancia a pesar de ser vital. Por lo que se recomienda dedicar tiempo para presentar cada diagrama a detalle, ya que será fundamental para el diseño y desarrollo del software.

FASE DE DISEÑO

Aquí se debe realizar el diseño de la base de datos, con todo lo que ello conlleva, como las tablas, llaves principales, foráneas etc. para ello era importante haber realizado el diagrama entidad relación en la fase anterior.

El diseño de entradas y salidas es una especie de primer vistazo del software que permite la comunicación del usuario con el sistema. Se recomienda realizar este diseño inmediatamente terminado el diseño de la base de datos, ya que la interacción del usuario permite ver si la base de datos es alimentada con los datos de entrada y si este sistema muestra los datos almacenados en la base de datos. Todo ello para tener la aprobación del diseño.

También se deben diseñar los casos de uso y el diseño de clases, basándose en los diagramas de casos de uso y diagramas de clases establecidos en la fase de análisis.

Diseño de interfaz: involucra todo aspecto del sistema como el color, texto, logo, imágenes etc.

También se debe realizar el diseño del logueo, ya que tiene que ver con la seguridad del sistema.

FASE DE CONSTRUCCIÓN

En esta fase se realizan las tareas del Back-End y Front-End específicamente. Al codificar los módulos del sistema, existe una forma para que el software sea escalable además de poder reutilizar código para otros proyectos, y es el uso del Modelo Vista Controlador (MVC), ya que, si en un mismo documento se presenta tanto código HTML, CSS, PHP,

JavaScript, se hace muy robusto realizar mantenimiento o identificar los módulos del sistema. En cambio, sí se realiza bajo el MVC estos problemas se solventan.

Referencias:

https://d1wqtxts1xzle7.cloudfront.net/53222887/Metodologias_Agiles-libre.pdf?1495404476=&response-content-disposition=inline%3B+filename%3DUniversidad_Nacional_de_Trujillo.pdf&Expires=1740530993&Signature=GM89j9yV~NPzFhgNBpsSeuaIa-G~gZdJ20lCkhs2KEu5OMxPY6UXT~p0IJbYE46frVgHicBQ4Cu8H1627TRIWx6tiLs5kcvtpMoI4fAogcYBMJJxY~kr0vAmCnHCej4swglMfsNJEV2ljW9Hp-g0M7u8iV-qsKaw7MQUw90hLp05fRZoqghbxeaOT4YqeHIsJ08y6vnnRrd-lrcODukK1fciiji80w9hN8OZWAnwv21NRCFZ88LyzHdkjKnlZYyxPdmZ9~zLn6vtTad26TCKS1p8~PVaG-fCsyhnglLdWEic5GMqD9hiDLuADh67E6dtsIHR5sPznF1Dn-1c0fTlyQ__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

Amaro Calderon, S. D. (

Eduardo Navarro, V. G, A. P. (2017). Metodología e implementación de six sigma.

Maida, E. G., & Pacienzia, J. (2015). Metodologías de desarrollo de software.

Canós, J. H., Letelier, P., & Penadés, M. C. (2003). Metodologías ágiles en el desarrollo de software. *Universidad Politécnica de Valencia, Valencia*, 1-8.

Aguirre Barrera, J., & Aguirre Barrera, S. (2021). Metodologías para el desarrollo de Proyectos.

SENA. (2017). *EL PROCESO DEL SOFTWARE*. SISTEMA DE APRENDIZAJE NACIONAL SENA, BOYACÁ. Centro Industrial de Mantenimiento Integral - CIMI Regional Santander: CREATIVE COMMONS.

SENA. (S.F). *GUIA PARA EL DESARROLLO DE SISTEMAS DE INFORMACIÓN.* Centro Industrial de Mantenimiento Integral - CIMI Regional Santander: CREATIVE COMMONS.