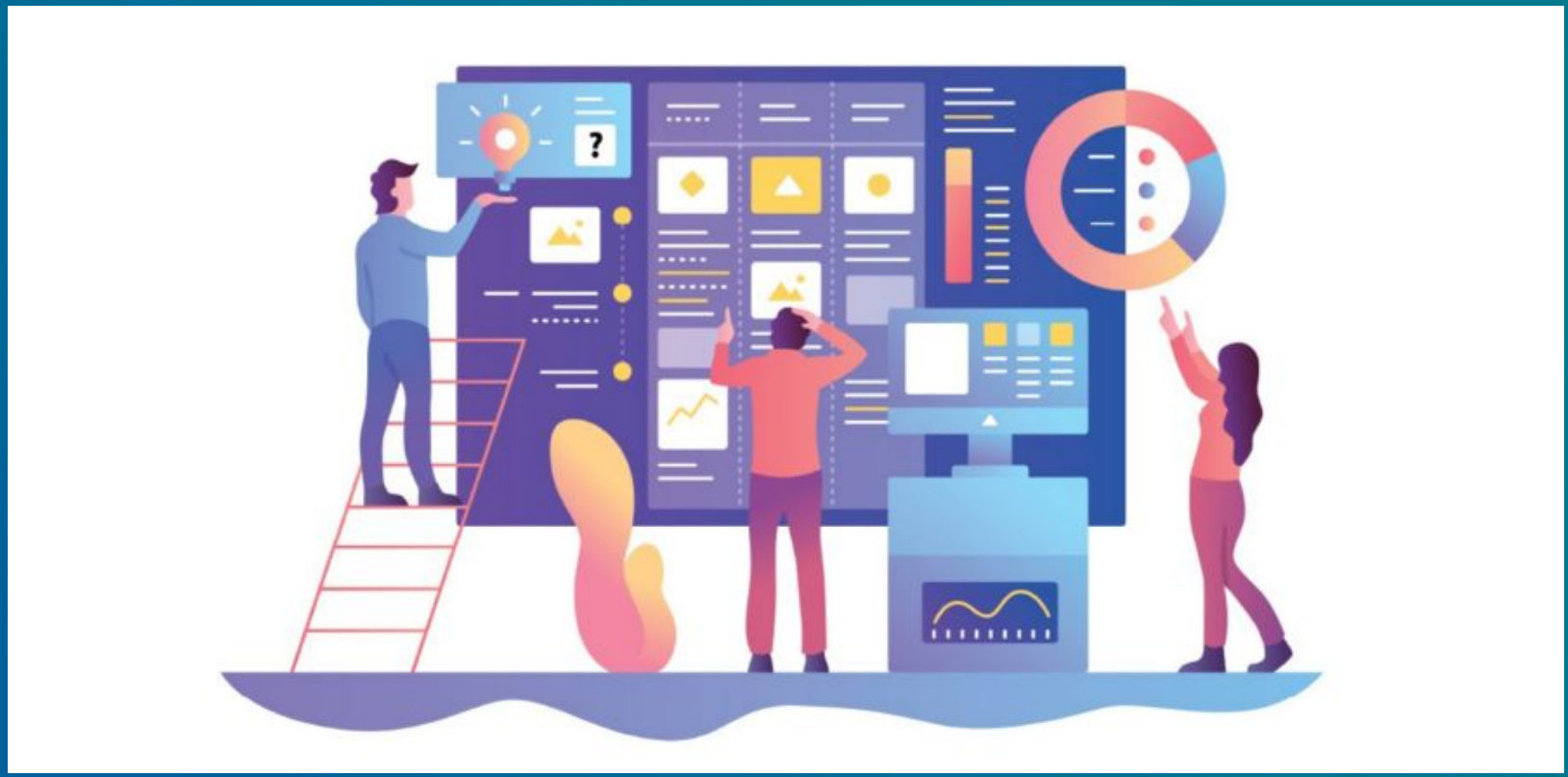
UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Metodología de Sistemas

I

Gustavo Julián Rivas

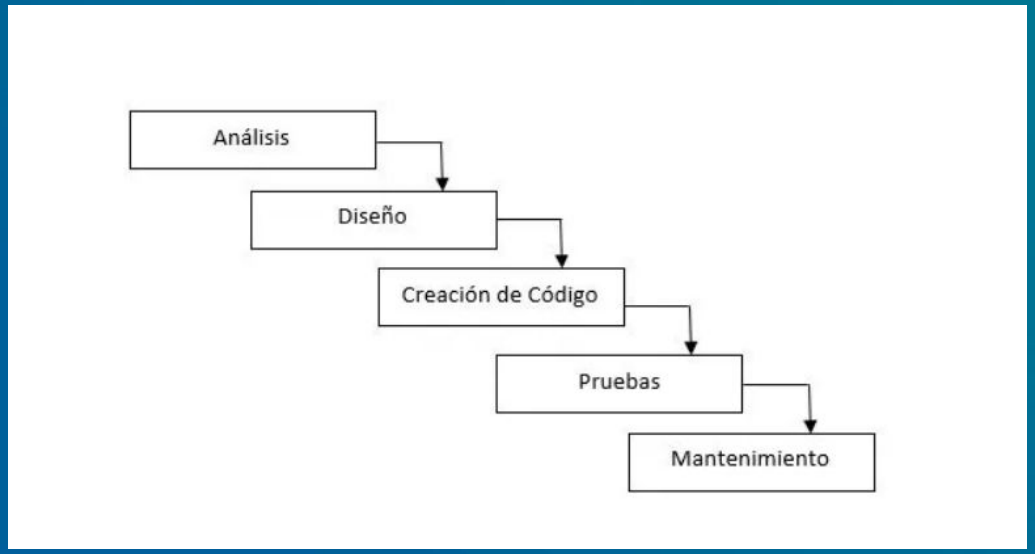


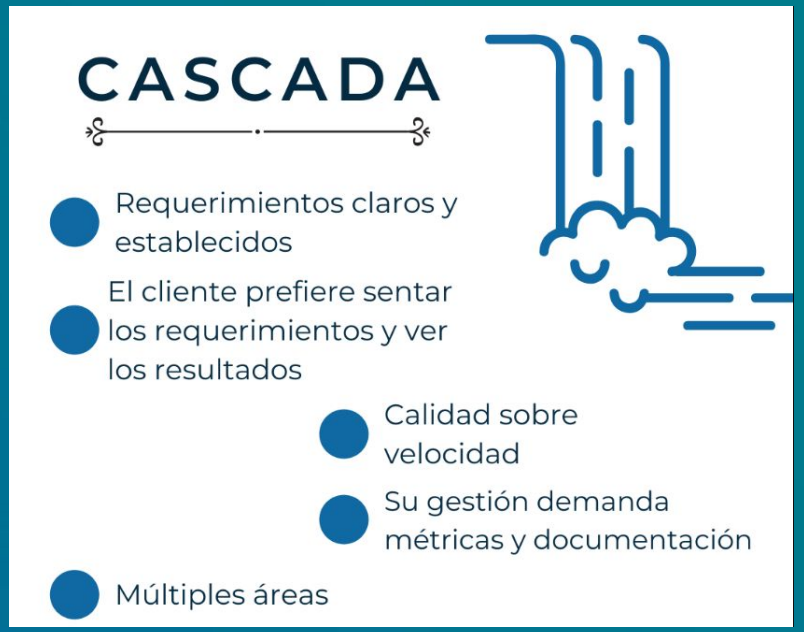
Introducción al Proceso de Desarrollo de Software

- Introducción a Waterfall (cascada) y Ágil
- Importancia de entender las etapas del desarrollo.

- **Introducción a Waterfall y Agile**

Qué es Waterfall y cómo funciona





Análisis: Es la fase en la cual se reúnen todos los requisitos que debe cumplir el software. En esta etapa es fundamental la presencia del cliente que documenta y repasa dichos requisitos. El ingeniero de software debe comprender el ámbito de la información del software así como la función, el rendimiento y las interfaces requeridas.

Diseño: Es una etapa dirigida hacia la estructura de datos, la arquitectura del software, las representaciones de la interfaz y el detalle procedimental (algoritmo). En forma general se hace un esbozo de lo solicitado y se documenta haciéndose parte del software.

Creación de código: Es la etapa en la cual se traduce el diseño para que sea comprensible por la máquina. Esta etapa va a depender estrechamente de lo detallado del diseño.

Pruebas: Esta etapa se centra en los procesos lógicos internos del software, asegurando que todas las sentencias se han comprobado, y en la detección de errores.

Mantenimiento: Debido a que el programa puede tener errores, puede no ser del completo agrado del cliente o puede necesitar, eventualmente acoplarse a los cambios en su entorno (sistema operativo o dispositivos periféricos) o a que el cliente, requiera ampliaciones funcionales o del rendimiento. Esto quiere decir que no se rehace el programa, sino que sobre la base de uno ya existente se realizan algunos cambios.

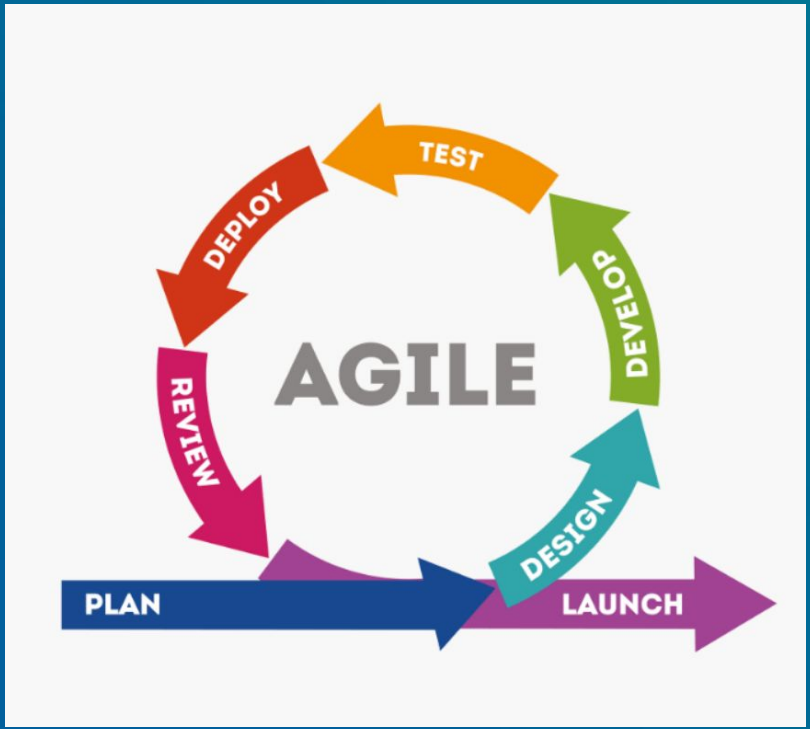
Ventajas.

- Ha sido muy utilizado y, por lo tanto, está ampliamente contrastado.
- El cliente tiene toda la información sobre el costo, el tamaño y la línea de tiempo del proyecto por adelantado. Ellos y el equipo de desarrollo también tienen una idea exacta sobre el resultado final del programa y trabajan para lograrlo desde el principio.
- Ayuda a detectar errores en las primeras etapas a bajo costo: El tiempo que se pasa en diseñar el producto en las primeras fases del proceso puede evitar problemas que serían más costosos cuando el proyecto ya estuviese en fase de desarrollo.
- Está dirigido por los tipos de documentos y resultados que deben obtenerse al final de cada etapa.
- La documentación es muy exhaustiva y si se une al equipo un nuevo desarrollador, podrá comprender el proyecto leyendo la documentación.
- Ideal para proyectos estables, donde los requisitos son claros y no van a cambiar a lo largo del proceso de desarrollo.

Desventajas.

- Es difícil que el cliente exponga explícitamente todos los requerimientos al principio, por lo cual, el producto final puede que no refleje todos los requisitos del usuario.
- No se va mostrando al cliente el producto a medida que se va desarrollando, si no que se ve el resultado una vez ha terminado todo el proceso. Esto provoca inseguridad por parte del cliente que quiere ir viendo los avances en el producto.
- El cliente necesitará ver una primera versión del software en funcionamiento. Entonces, cambiarán muchos requisitos y añadirán otros nuevos, lo que supondrá volver a realizar fases ya superadas y provocará un incremento del coste.
- El proceso de creación del software tarda mucho tiempo ya que debe pasar por el proceso de prueba y hasta que el software no esté completo no se opera. Esto es la base para que funcione bien.
- Los diseñadores pueden no tener en cuenta todas las dificultades que se encontrarán cuando estén diseñando un software, lo que conllevará a rediseñar el proyecto para solventar el problema.
- Para proyectos a largo plazo, este modelo puede suponer un problema al cambiar las necesidades del usuario a lo largo del tiempo. Si por ejemplo, tenemos un proyecto que va a durar 5 años, es muy probable que los requisitos necesiten adaptarse a los gustos y novedades del mercado.

Qué es Agile y cómo funciona



Metodología de Sistemas

La metodología Agile nace a principios del siglo XXI como necesidad de corregir determinadas prácticas ineficientes en el desarrollo de proyectos y se ha convertido en un modelo estandarizado que aplican grandes empresas nacionales e internacionales para optimizar sus recursos en la gestión de proyectos.

Aunque se desarrolló originariamente para el sector informático, concretamente en el campo de perfeccionamiento de softwares, gracias a los buenos resultados obtenidos, su aplicación se lleva a cabo en otros segmentos de negocio y proyectos de innovación.

La metodología Agile aplicada al desarrollo de proyectos permite:

- Tener una visión global del proyecto.
- Asignación de mini proyectos e hitos para el equipo.
- Incrementar el proyecto en fases.

Las metodologías ágiles no siempre son fáciles de implantar, porque en su aplicación hay que tener en cuenta el propio proyecto, su dificultad y los distintos actores que intervienen en él.

El método Agile se basa en 4 pilares fundamentales y 12 principios que se deben aplicar para obtener los resultados más eficientes.

Cabe destacar que la metodología Agile se centra más en personas que en procesos, además de en la adaptación, mutación y cambio para satisfacer al cliente.

Pilares esenciales de la metodología Agile

1. Valoración de las personas y las relaciones sociales por encima de herramientas y procesos.
2. Colaboración directa con el cliente para mantener una relación más participativa y cercana.
3. Priorización funcional del producto por encima de la acumulación y exceso de documentación.
4. Responder de forma ágil y efectiva a los imprevistos y cambios que puedan haberse desarrollado en el plan inicial.

La metodología Agile rompe con los esquemas de los proyectos planificados linealmente, formas de trabajo tradicionales poco productivas y demasiado prolongadas en el tiempo. De este modo se pueden ejecutar proyectos basados en entregas más rápidas y flexibles con la puesta en práctica de planificaciones rigurosas y exhaustivas, que tienen en cuenta las novedades y modificaciones que pueden surgir a lo largo del proyecto.

Los 12 principios de la metodología Agile

Para poder aplicar este método es imprescindible ceñirse a estos 12 principios fundamentales:

1. Perseguir la satisfacción del cliente e informarle periódicamente del estado del proyecto.
2. Los nuevos cambios y requisitos son bienvenidos y se valoran como modificaciones positivas.
3. La división del trabajo se realiza en fases temporales productivas divididas en semanas, quincenas, etc.
4. Posibilidad de medir el progreso.

Los 12 principios de la metodología Agile

5. La forma de ejecutar los proyectos debe garantizar en sí misma la continuidad del proyecto (desarrollo sostenible).
6. El equipo debe trabajar de forma coordinada y en conjunto, utilizando el método Scrum, como una práctica efectiva y esencial para la correcta organización y desarrollo del trabajo.
7. Las conversaciones entre los integrantes del equipo y/o el cliente deben llevarse a cabo en persona, para comunicar de forma eficaz los mensajes.
8. Es necesario infundir motivación y confianza a los miembros que forman parte del proyecto para obtener procesos exitosos.

Los 12 principios de la metodología Agile

9. Excelencia técnica y buen diseño. En la metodología Agile, la calidad del trabajo y la presentación forman parte del conjunto.
10. Se impone la ley de la simplicidad. Las tareas deben ser lo más sencillas posible. En caso de no poder simplificarlas se tendrá que dividir en iteraciones para reducir su nivel de complejidad.
11. Equipos autogestionados. Aunque es necesario que exista una figura que monitorice los equipos de trabajo, éstos deben ser capaces de organizarse por sí mismos.
12. Adaptación a las circunstancias cambiantes. Es imprescindible que los profesionales que ejecuten los proyectos puedan adaptarse a las distintas circunstancias y modificaciones que puedan surgir durante el proceso.

Comparación Directa - Waterfall vs. Agile

Criterio	Waterfall	Agile
Estructura	Secuencial, rígida	Iterativa, flexible
Cambios de requisitos	Difíciles de implementar una vez comenzado el proyecto	Fácil de acomodar durante el ciclo de desarrollo
Documentación	Muy detallada al inicio	Documentación continua y ligera
Participación del cliente	Mínima durante el desarrollo; involucrado principalmente al inicio y fin	Continua; retroalimentación constante en cada sprint
Tiempo de entrega	Largo; todo el software se entrega al final	Corto; entregas incrementales y continuas
Costos de cambio	Altos si se realizan cambios después de la fase inicial	Bajos, ya que se esperan cambios y adaptaciones

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Metodología de Sistemas

Ejercicio Práctico - Escenarios de Proyecto

Fase 1 - Recolección de Requisitos

- Definición y propósito.
- Tipos de requisitos (funcionales y no funcionales).
- Herramientas utilizadas (e.g., entrevistas, encuestas, análisis de stakeholders).

Objetivos de la Recolección de Requisitos:

- Entender las necesidades del negocio.
- Identificar los objetivos y expectativas del cliente.
- Documentar todos los requisitos funcionales y no funcionales.
- Obtener consenso y aprobación de todos los stakeholders.

Tipos de Requisitos

- Requisitos Funcionales
- Requisitos No Funcionales
- Requisitos del Negocio
- Requisitos Técnicos

Técnicas para la Recolección de Requisitos

- Entrevistas
- Cuestionarios y Encuestas
- Workshops o Talleres
- Análisis de Documentos
- Observación
- Prototipos

Herramientas para la Recolección de Requisitos
¿Cuales conocen?

Roles Involucrados en la Recolección de Requisitos

- Business Analyst (BA)
- Product Owner (PO)
- Desarrolladores y Testers (Devs / Qas)
- Usuarios Finales

Buenas Prácticas para la Recolección de Requisitos

- Involucrar a todos los stakeholders desde el inicio
- Documentar claramente los requisitos
- Priorizar los requisitos
- Validar y obtener aprobación

Ejemplo Práctico de Recolección de Requisitos

Contexto: Desarrollo de una plataforma de e-commerce para una cadena de tiendas minoristas.

- Requisito Funcional: El sistema debe permitir a los clientes buscar productos por categoría, nombre, o precio.
- Requisito No Funcional: El sistema debe cargar la página de resultados de búsqueda en menos de 2 segundos.
- Requisito del Negocio: Aumentar las ventas en línea en un 20% en los próximos 12 meses.
- Requisito Técnico: Integración con el sistema de gestión de inventario existente.

UTN AVELLANEDA
Tecnicatura Universitaria en Programación

Metodología de Sistemas

Ejercicio Práctico - Recolección de requisitos

Fase 2 - Análisis y Diseño del Sistema

- Definición de arquitectura del sistema.
- Modelado de datos (UML, diagramas de flujo).
- Importancia de la documentación clara.

Introducción al Análisis y Diseño del Sistema

- Objetivos de la fase de análisis y diseño.
- Importancia en el contexto del SDLC.

Análisis del Sistema

- Análisis de Requisitos: Comprensión y refinamiento.
- Identificación de actores, casos de uso, y flujo de trabajo.
- Técnicas de modelado (diagramas UML, diagramas de flujo, etc.).
- Herramientas utilizadas para el análisis

Diseño del Sistema

- Diseño de Arquitectura del Software: Definición de la estructura del sistema.
- Diseño Detallado: Componentes del sistema, módulos, bases de datos, interfaces de usuario.
- Modelos de Diseño: MVC (Modelo-Vista-Controlador), Diseño Orientado a Objetos.
- Herramientas de diseño

Documentación en la Fase de Análisis y Diseño

- Importancia de la documentación.
- Tipos de documentación (documentos de especificación, diagramas, etc.).