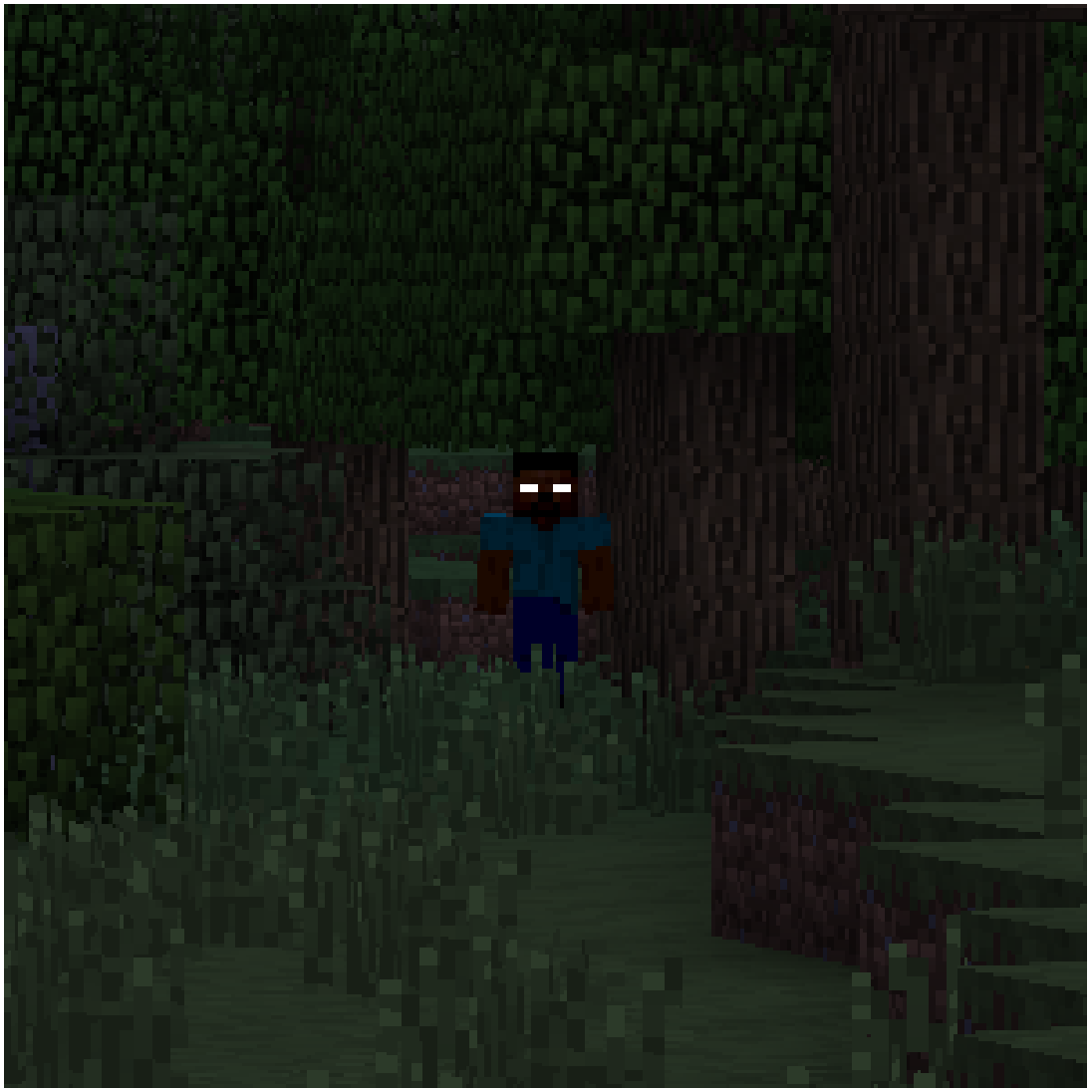
Metodologías de Desarrollo

Modelo iterativo incremental
Daniel Espinosa



13/11/2026

Entorno de Desarrollo

Rayssa Cristina Dantas
Erika Aldama Escaray

ÍNDICE

1. Introducción y Conceptos Clave

1.1 ¿Qué es?

Es una metodología de desarrollo de software. Su principal objetivo es planificar el proyecto en diversos bloques temporales que se pueden entender como mini-proyectos . Las iteraciones se repiten en un proceso de trabajo similar para tener un resultado completo sobre el producto final.

El iterativo hace referencia a “refinamiento”, consiste en repetir un ciclo de desarrollo para mejorar el sistema, profundizando y perfeccionando lo que ya existe.

El incremental hace referencia a “crecimiento”, que consiste en dividir el proyecto en pequeñas piezas funcionales que se van añadiendo al sistema progresivamente.

2. El Ciclo de Vida (El "Loop")

Podemos decir que el ciclo de vida es un “loop” porque literalmente, el proceso se repite muchas veces, como en bucle. Comparando con el modelo cascada, que es un desarrollo más lineal, el iterativo es un bucle infinito hasta que el cliente esté satisfecho.

Para definir las fases de esta metodología, tenemos:

1. Análisis de requisitos, que se eligen las funcionalidades específicas del proyecto.
2. Diseño, que se planea la arquitectura de las nuevas funciones.
3. Implementación, donde se escribe el código.
4. Pruebas o testing, que se verifica que las implementaciones nuevas no hayan roto lo anterior.
5. Evaluación o despliegue, se entrega una versión funcional, aunque incompleta, para que el cliente pueda dar un feedback.

En un caso práctico, se usan herramientas para realizar todos los pasos de una manera más organizada para gestionar los cambios constantes.

- Control de versiones (Git)

  El uso de Git es básicamente obligatorio teniendo en cuenta que el código cambia constantemente. Se suele utilizar un flujo de trabajo como Gitflow. Se crean ramas específicas para cada incremento y cuando el incremento está terminado y probado, se fusiona con la rama principal (main).

- Pruebas y testing (JUnit)

  Al añadir incrementos, existe el alto riesgo de que el nuevo rompa lo viejo. Hacen falta las pruebas unitarias para crear tests automatizados para cada nueva funcionalidad.

  Antes de cerrar una iteración, se ejecutan todos los tests hechos anteriormente para asegurar la estabilidad del sistema.

- Integración Continua (CI/CD)

  Herramientas como Jenkins o GitHub Actions permiten que a cada vez que un desarrollador haga un incremento, el servidor compile el proyecto y pase los tests automáticamente.

- Refactorización

  Este modelo puede generar el “código sucio”, que son básicamente parches sobre parches.

  Para evitarlo, se usan las herramientas de refactorización del IDE (IntelliJ, Eclipse, VS Code) para limpiar el código y mejorar la legibilidad sin cambiar su comportamiento externo.

3. Ventajas y Desventajas (Análisis Crítico)

Demuestra que entiendes las implicaciones de elegir este modelo.

- Ventajas:

  Gestión de riesgos y calidad del producto: Al hacer pruebas en cada incremento ayuda a detectar errores con mucha más facilidad y también ayuda a mejorar la calidad general.

  Feedback del usuario: El cliente ve avances desde el principio, no al final del proyecto, además de poder añadir nuevas ideas.

  Adaptabilidad: Si el mercado cambia a mitad del proyecto o se incorporan requisitos nuevos, el modelo puede adaptarse sin problema.

- Desventajas:

  Requiere una supervisión constante: Es necesario un alto nivel de gestión para coordinar iteraciones, retroalimentación y priorización.

  El cliente debe participar activamente: Sin el feedback continuo de los usuarios, la metodología pierde gran parte de sus ventajas.

  No siempre es el más eficiente: Cuando la idea y los requisitos están muy bien definidos y sin deseo de cambiarlos no es muy eficiente.

4. Utilidad del modelo iterativo incremental

La metodología iterativo-incremental es ideal para proyectos con requisitos cambiantes, necesidad de entregar valor rápido, alto riesgo técnico y clientes que participan constantemente.

Ya que este modelo está basado en la flexibilidad, el feedback continuo, la experimentación temprana y los cambios.

No es recomendable para proyectos con requisitos totalmente fijos, presupuesto o plazos muy rígidos, o con poca participación del cliente.

Este modelo brilla con diferencia en proyectos como: videojuegos con actualizaciones constantes (o la prueba alfa/beta las cuales buscan cambiar en torno al usuario), redes sociales, software empresarial y apps experimentales. Entre otros muchos.

5. Conclusión

Quisimos hacer la presentación de este modelo en específico ya que nos pareció uno de los más interesantes, ya que hay un feedback y cambio constante, lo que conlleva mejoras tanto de interfaz como del mismísimo proyecto en sí, por lo que puedes empezar con una idea muy vaga y terminar con algo totalmente diferente.

Somos conscientes de que este modelo es bastante costoso pero en los proyectos en los que se usa es necesario, pues están normalmente inclinados hacia el público.

FUENTES:

https://proyectosagiles.org/desarrollo-iterativo-incremental/
https://pmbc.es/que-es-desarrollo-iterativo-e-incremental/
https://devtia.com/post/desarrollo-iterativo-e-incremental
http://agilismoatwork.blogspot.com/2014/10/modelos-de-proceso-para-desarrollo-agil.html#:~:text=Proceso%20Iterativo%20e%20Incremental%20Tiene%20todas%20las,y%20revisar%20con%20el%20cliente%20cada%20versi%C3%B3n.