Metodologías de Desarrollo

Metodología en cascada:

1. **Análisis de Requisitos.** Se documentan los objetivos del Software a desarrollar.
2. **Diseño del Sistema.** Diseña la estructura del sistema se especifica cada una de las partes que se planea desarrollar.
3. **Diseño del Programa.** Realizar los algoritmos correspondientes a utilizar durante la fase de programación.
4. **Codificación.** Programar el código del software en el lenguaje que se requiera.
5. **Ejecución de Pruebas.** Intencionalmente tratar de “dañar” el software con el objetivo de buscar posibles errores que necesiten corrección.
6. **Verificación.** Ejecución del software por parte del usuario final, si se realizó las pruebas de manera correcta no deberían de haber problemas.
7. **Mantenimiento.** Actualizar el software constantemente para solucionar errores o añadir otras funcionalidades.

Metodología de Prototipos:

1. **Planeación.** Debe ser simple ya que se trata de crear un prototipo.
2. **Modelado.** Al igual que la planeación debe ser rápida.
3. **Elaboración del Prototipo.** Establecer la estructura primaria del software que se va a presentar al cliente.
4. **Desarrollo.** Programar el código tomado en cuenta las especificaciones creadas durante el modelado y las necesidades dadas por el cliente.
5. **Entrega y Retroalimentación.** Entregar y dar instrucciones al usuario de cómo utilizar el programa.
6. **Comunicación con el Cliente.** Mantener contacto con el cliente para recibir informes de calidad o si desea agregar funciones al prototipo entregado.
7. **Entrega del Producto Final.** Luego de implementar todo lo que el cliente sugirió y hacer varias pruebas, entregar el software finalizado.

Metodología Incremental:

1. **Inicialización.** Ideas, requisitos y especificaciones que requiera el proyecto.
2. **Periodos de Iteración.** Realizar la primera iteración con las características iniciales o principales, se pueden hacer el número de iteraciones que sean necesarias para satisfacer los requerimientos establecidos por el usuario.
3. **Lista de Control.** En caso de que alguna iteración no funcione correctamente o el usuario ya no la requiera, se debe tener una lista de todas las iteraciones realizadas, desde la primera.

Metodología en Espiral:

1. **Determinar el Objetivo.** Los objetivos planeados inicialmente se irán actualizando con cada nueva iteración del método.
2. **Análisis de Riesgo.** Identificar posibles amenazas o problemas que puedan afectar el proyecto y tomar las medidas respectivas para evitarlas.
3. **Desarrollar, Validar y Probar.**
4. **Planificación.** Para realizar otra iteración en esta etapa se evalúa y analiza todo lo que hicimos en las fases anteriores y en base a esto decidir cómo avanzar con nuestro proyecto.

Metodología RAD (Rapid Application Development):

Basado en el uso de iteraciones y el manejo de prototipos la metodología RAD se diferencia en el uso de herramientas CASE que permiten acelerar el proceso. Se subdivide en pequeños proyectos que se irán optimizando individualmente, reduciendo riesgos y disminuyendo el tiempo en el que se completa el proyecto.

Metodologías Ágiles:

Scrum:

**Características:**

1. **Desarrollo Incremental.** El desarrollo ira incrementando sin importar el orden en el que se lleven los procesos.
2. **Calidad de las Personas.** El suceso del proyecto dependerá de las personas encargadas, la organización y el conocimiento de los equipos de trabajo.
3. **No es Secuencial ni en Cascada.** Se puede devolver a alguno de los procesos anteriores en los que sea necesario elaborar más, a esto se le llama solapamiento.
4. **La Comunicación es Fundamental.** Los equipos de trabajo necesitan tener una buena comunicación tanto interna como con los otros equipos.

**Pasos:**

1. **Product Backlog.** Es una lista de las funcionalidades del proyecto a desarrollar ordenadas de acuerdo a su prioridad, es elaborada por el Product Owner.
2. **Sprint Backlog.** Consiste en seleccionar algunos de los puntos escritos en el Product Backlog, los cuales procederán a ser realizados, marca el tiempo en que se llevara a cabo el Sprint.
3. **Sprint Planning Meeting.** Reunión para definir plazos y subdividir los procesos necesarios para el desarrollo del proyecto.
4. **Daily Scrum o Stand-up Meeting.** Reunión diaria durante el tiempo de Sprint para responder las preguntas: ¿qué hice ayer?, ¿qué voy a hacer hoy?, ¿qué ayuda necesito?; es liderada por el Scrum Master.
5. **Sprint Review.** Reseña de lo que se realizó durante el Sprint.
6. **Sprint Retrospective.** analizar los objetivos cumplidos, si se cometieron errores, visualizarlos y tratar de no cometerlos nuevamente.

Kanban:

**Principios:**

1. **Garantía de Calidad.** Promueve la calidad antes que la velocidad a pesar de ser una metodología ágil.
2. **Desperdicios.** Se evita todo aquello que se pueda considerar extra, superficial o innecesario optimizando tiempos y costos.
3. **Mejora Continua.** Dependiendo del objetivo final se pueden ir mejorando los procesos constantemente y se pueden aplicar a cualquier tipo de proyecto no solo a desarrollo de software.
4. **Es Flexible.** Al no tener una línea de trabajo esta metodología es más dinámica, nos permite resolver problemas que surjan de imprevisto y adelantar procesos.

**Pasos:**

1. **Definir el Flujo de Trabajo.** En un tablero establecer un número de tareas, fases o proyectos que se tengan disponibles e ir actualizándolos conforme sea necesario.
2. **Fases de Ciclo de Producción.** Subdividir los procesos en pequeños segmentos para agilizar el proyecto, establecer horas o fechas límites para cada uno de ellos.
3. **Stop Starting, Start Finishing.** Cada nueva tarea que se empieza se debe terminar antes de empezar otra.
4. **Tener un Control.** Gracias a las notas que se van colocando en el tablero se mantiene un buen flujo del trabajo lo que nos permite manejar varios proyectos simultáneamente.

XP:

**Valores:**

1. **Simplicidad.** El diseño se simplifica al máximo, si el código se puede simplificar se simplifica, evitar documentación extra
2. **Retroalimentación.** El cliente podrá corregir, quitar, agregar o excluir partes del código durante su desarrollo.
3. **Valentía.**
4. **Respeto.**

**Características:**

1. **Tipo de Desarrollo Iterativo e Incremental.** Mejoras continuas a base de iteraciones y desarrollo incremental como el modelo espiral.
2. **Pruebas Unitarias.** Se utiliza software de codificación para analizar el código y solucionar errores, antes de validarlo y darlo por bueno.
3. **Trabajo en Equipo.** Se trabaja en parejas.
4. **Alguien del Equipo Trabaja con el Cliente.** Una persona está encargada de reunirse de forma constante con el cliente para preguntarle sus opiniones de cada progreso.
5. **Corrección de Errores.**
6. **Restructuración del Código.** Se simplifica el código pero no sus funciones.
7. **El Código es de Todos.** Aunque se trabaje en grupos todos tienen derecho de revisar el código de otro y a proponer cambios o correcciones.
8. **Código Simple es la Clave.** El código debe de ser simple de lo contrario es posible que se tome mucho más tiempo realizar cambios al ser muy complejo.

MyISAM VS INNODB

|  |  |  |
| --- | --- | --- |
|  | MyISAM | INNO DB |
| Bloqueo de Tablas | O | O |
| Bloqueo de Registro | X | O |
| Soporta Transacciones | O | O |
| Integridad de Datos | X | O |
| Rápida Velocidad de Carga | O | X |
| Recuperación automática de Datos | X | O |
| Restricciones de Clave Externa | X | O |
| ACID | X | O |

Bibliografía:

<https://okhosting.com/blog/metodologias-del-desarrollo-de-software/>

<https://www.webreunidos.es/blog/myisam-vs-innodb/>

<https://blog.openalfa.com/diferencias-entre-innodb-y-myisam-en-mysql>