Universidad de las Fuerzas Armadas ESPE
Fundamentos de Software
NRC 1951

Integrantes: Gerald Astudillo, Henry Chalcualan, Lenin Erazo, Henry Suin.

Actividad: definición, bases y ejemplo de una metodología orientada a procesos y objetos

Metodología Estructurada de Edward Yourdon

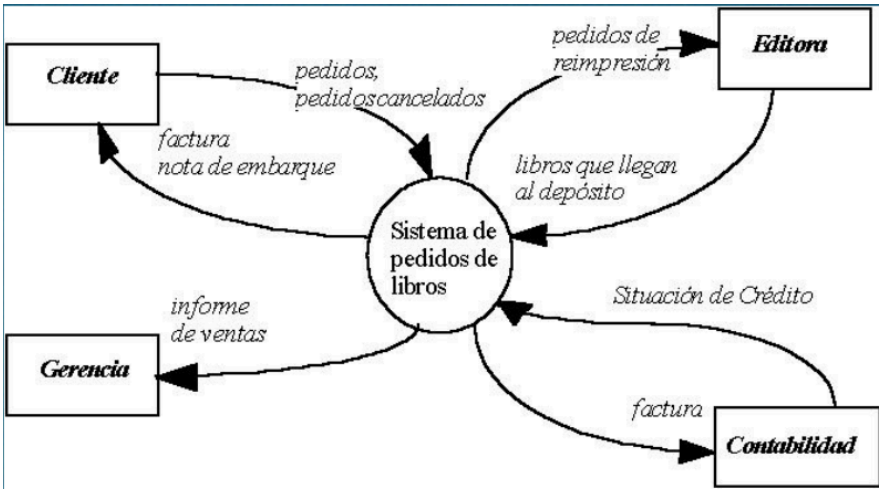
Definición:

La metodología estructurada de Edward Yourdon es un enfoque sistemático para el desarrollo de software, basado en el análisis y diseño estructurado. Su objetivo es descomponer sistemas complejos en módulos más manejables, utilizando diagramas y herramientas visuales para representar los procesos y datos del sistema.

Fundamentos/bases de la metodología:

- **Identificar el problema:** Para luego modelar el aspecto dinámico o estático del sistema.
- **Aspecto ambiental:** Implementar herramientas y tecnologías que optimicen la ejecución de los procesos.
- **Monitoreo y Medición:** Evaluar constantemente los procesos para identificar oportunidades de mejora.
- **Optimización Continua:** Aplicar mejoras con base en métricas y análisis de desempeño.
- **Enfoque en la Calidad:** Asegurar que cada fase del proceso cumpla con estándares de calidad establecidos.

Ejemplo:



Automatización del Proceso de Aprobación de Solicitudes de Compra en una Empresa

1. Identificación del Proceso

La empresa tiene un proceso manual para aprobar solicitudes de compra que genera retrasos y errores. Se decide automatizar el proceso con un sistema BPM para mejorar la eficiencia.

2. Modelado del Proceso

Se diseña un diagrama, donde se definen las siguientes etapas:

- **Solicitud de compra:** Un empleado llena un formulario en el sistema.
- **Revisión del supervisor:** Si el monto es menor a $1,000, se aprueba automáticamente. Si es mayor, se envía a Finanzas.
- **Aprobación de Finanzas:** Si el monto supera $10,000, se requiere aprobación de la gerencia.
- **Generación de orden de compra:** Si se aprueba, se genera una orden de compra en el sistema ERP.

3. Implementación del Proceso

- Se integran APIs para generar órdenes de compra automáticamente.
- Se implementa RPA con UiPath para automatizar la entrada de datos.

4. Ejecución y Monitoreo

- Se usará una configuración para ver métricas como tiempo promedio de aprobación y cantidad de solicitudes procesadas.
- Se generan alertas si una solicitud tarda más de 48 horas en aprobarse.

5. Optimización del Proceso

- Se analiza el rendimiento con minería de procesos para detectar cuellos de botella.
- Se agregan reglas para mejorar la toma de decisiones.

Metodologías Orientadas a Objetos: XP

Definición:

La programación extrema es una metodología ágil de gestión de proyectos que se centra en la velocidad y la simplicidad con ciclos de desarrollo cortos y con menos documentación.

Fundamentos/bases de la metodología:

Desarrollo Incremental: El software se desarrolla en pequeñas liberaciones frecuentes, con incrementos que aportan funcionalidad útil para el negocio desde el inicio.

Participación Activa del Cliente: El cliente está involucrado constantemente en el proceso, trabajando estrechamente con el equipo de desarrollo para definir y priorizar los requerimientos, y también para crear pruebas de aceptación.
Propiedad Colectiva del Código: Todos los miembros del equipo tienen responsabilidad sobre el código. Esto fomenta la colaboración y evita la creación de "islas de experiencia" dentro del equipo.
Programación en Pares: Los programadores trabajan en parejas, lo que permite la revisión continua del trabajo y el apoyo mutuo para asegurar la calidad del código.
Refactorización Continua: El código es constantemente mejorado y optimizado para evitar la degeneración de su estructura y mantenerlo simple y fácil de mantener.
Integración Continua: Las nuevas funcionalidades se integran al sistema de manera continua, y se ejecutan pruebas automáticamente para asegurar que todo siga funcionando correctamente.
Simplicidad: El diseño del sistema es simple y se enfoca solo en los requerimientos actuales, sin anticipar cambios futuros innecesarios.
Ritmo Sustentable: El equipo trabaja a un ritmo que puede mantenerse a largo plazo, evitando jornadas de trabajo excesivamente largas que puedan afectar la calidad del código.

Ejemplo:

Aplicación para gestión de tareas

Desarrollo Incremental

- En la primera iteración, el equipo crea una funcionalidad básica: los usuarios pueden crear y ver tareas.
- En la siguiente iteración, añaden la opción de editar y eliminar tareas.
- Cada semana lanzan una nueva versión con nuevas características.

Participación Activa del Cliente

- El cliente (el equipo que usará la aplicación) está involucrado desde el inicio. Al comienzo, se reúne con el equipo para definir qué funciones son más importantes (crear tareas, asignar fechas, etc.).
- Después, el cliente revisa las versiones frecuentes y valida que cada funcionalidad se ajusta a sus necesidades.

Propiedad Colectiva del Código

- Todo el equipo de desarrollo tiene acceso al código y puede modificarlo si es necesario. Si uno de los programadores encuentra un problema, cualquier otro puede solucionarlo, lo que mejora la colaboración.

Programación en Pares

- Dos desarrolladores trabajan juntos: uno escribe el código y el otro revisa y sugiere mejoras. Por ejemplo, uno podría estar escribiendo la funcionalidad para asignar tareas a los usuarios, mientras el otro revisa si el código está optimizado.

Refactorización Continua

- Después de cada iteración, el equipo revisa el código para simplificarlo. Por ejemplo, si al agregar una nueva funcionalidad el código se volvió complejo, se refactoriza para hacerlo más limpio y fácil de mantener.**Integración Continua**
- Cada vez que se termina una nueva funcionalidad, el código se integra al repositorio central y se ejecutan pruebas automáticas para verificar que todo siga funcionando correctamente. Por ejemplo, después de agregar la función para agregar fechas a las tareas, se ejecutan pruebas para asegurarse de que la funcionalidad anterior (crear tareas) siga funcionando.

Simplicidad

- El equipo de desarrollo solo crea lo necesario en cada etapa. Por ejemplo, la interfaz de usuario comienza simple: solo permite agregar tareas. No se añaden características innecesarias como recordatorios complejos hasta que el cliente lo pida.

Ritmo Sustentable

- El equipo trabaja en ciclos cortos y mantiene un ritmo de trabajo constante. No se sobrecargan, y se asegura de que cada miembro pueda mantener su productividad sin agotarse.