ROMPEMEGATON RESUMEN INGENIERÍA SOFTWARE 2

DIAGRAMAS

UML es el Lenguaje Unificado de Modelado, es un lenguaje grafico capaz de expresar requisitos, arquitectura y diseño del software, sirve para comunicarse entre desarrolladores, con los clientes y usar herramientas de generación automática de código. UML consiste en un conjunto integrado de diagramas definidos para ayudar a los desarrolladores de software y sistemas a realizar las tareas de análisis y diseño.

rubius**OMG** (Object Managment Group) es un consorcio internacional de estandarización de tecnologías orientadas a objetos, estandariza UML.

Abstracción: la técnica de hacer un modelo de tus ideas del mundo, por ejemplo: un mapa es un modelo del mundo. En UML se usa una abstracción del sistema ~~con el objetivo de~~ para que sea fácil de entender.

Tipos de diagramas:

- diagramas estructurales: muestran los elementos de construcción del sistema, características que no cambian con el tiempo.
- diagramas de comportamiento: muestra como el sistema responde a las peticiones o evoluciona con el tiempo.
- diagramas de interacción: ~~engloba a ciertos~~ son algunos diagramas de comportamiento que muestran el intercambio de mensajes ~~dentro~~ de un grupo de objetos que cooperan en un proyecto.

1) **Diagrama de casos de uso**: técnica ~~para~~ que especificar el comportamiento de un sistema, es una secuencia de interacciones entre un sistema y alguien/algo que usa alguno de sus servicios.

en UML no se tiene que entender a un sistema como un conjunto de objetos que interactúan, el su éxito de los casos de uso es porque propone que la mejor forma de entender el sistema es a partir de los servicios que ofrece, sin importar como los objetos interactúan dentro.

Puede usar los datos que reúne para identificar las funciones y la forma en que los roles interactúan con ellas, para una visión de alto nivel del sistema, es independiente del método de diseño y de la programación, o identificar los factores internos y externos.

Características: están expresados desde el punto de vista del actor, se documentan con texto informal, hace énfasis en la interacción, son iniciados por un único actor, están acotados al uso de una función determinada del sistema.

Componentes:

- actor: cualquier entidad que desempeñe un papel, es un stickman.
- caso de uso: función o acción dentro del sistema, es un óvalo con la función.
- relaciones: unen dos casos de uso, es una flecha con línea punteada.
- sistema: define el alcance del caso de uso, es un rectángulo.
- <<extend>> < - - - (señala al caso d uso principal, hacia abajo), <<include>> - - - > (señala al caso de uso obligatorio, a la derecha) herencia flecha con triangulo vacio.

2) **Diagrama de clases**: diagrama estático que usa un clasificador, es como se relacionan los componentes, pero no “qué y cómo” lo hacen.

- objeto: se puede describir un objeto, se puede colocar anónimos.
- atributos: son propiedades, datos que guarda la clase, visibilidad nombre : tipo.
- operaciones: funciones que el objeto va a hacer, métodos, visibilidad nombre(parámetros) : tipo.
- visibilidad: + publico, - privado, # protegido, ~ paquete.

- estereotipos: permiten definir nuestros propios elementos y roles, <<nombre>>.
- asociación: muestra relación entre clases, se lee tiene un, línea simple < > indican dirección.
- multiplicidad: cardinalidad, indica número de objetos en una asociación.
- clase asociada: se lleva a cabo por medio de una clase que tiene atributos propios, línea de puntos.
- agregación: es parte de/está hecho de, diamante en blanco, si se separan siguen existiendo.
- composición: si el contenedor se destruye entonces se destruye todo, la parte pertenece a un solo contenedor, diamante pintado.
- generalización: herencia, flecha con triangulito vacio.
- dependencia: relación débil y temporal entre dos clases o clase e interfaz, una requiere de otra para ofrecer un servicio, linea punteada con flecha al elemento del cual depende.

3) **Diagrama de objetos**: cómo se relacionan los objetos en un sistema en un estado concreto

4) **Diagrama de estados**: representa los diferentes estados por los que pasa un objeto a lo largo de su ciclo de vida, así como los eventos/condiciones que causan los cambios de un estado a otro

5) **Diagrama de secuencia**: representan la interacción entre objetos en orden cronológico

6) **Diagrama de actividad**: representa el flujo de trabajo o la secuencia de actividades en un proceso

7) **Diagrama de comunicación/~~colaboración~~**: muestra cómo los objetos colaboran entre sí mediante el intercambio de mensajes para llevar a cabo un proceso

→ Tips para diagramas de estados:
- identificar el objeto a modelar,
- definir todos los estados posibles,
- se determinan los eventos que cambian los estados,
- se dibuja el estado inicial, transiciones, etc., nombres claros y condiciones.

→ Tips para diagramas de actividad: (usarlo para procesos definidos y decisiones condicionales, flujo de un caso de uso completo, si un paso se repite se usa bucle, si varias actividades al mismo tiempo usar paralelo)
- identificar el proceso/flujo a modelar,
- se establece punto de inicio,
- se listan las acciones en pasos en orden,
- incluir decisiones con condiciones,
- añadir bifurcaciones y uniones si hay tareas en paralelo,
- se define un punto final.

RUP

El **Proceso Racional Unificado (RUP)** es un proceso de desarrollo de software que junto a UML constituye la metodología estándar más usada orientada a objetos, es un conjunto de metodologías adaptables al contexto y necesidades de cada organización, permite realizar actividades y artefactos de acuerdo con la elección del equipo y se puede adaptar para agilizar el proceso.

características:

- se centra en capturar los requisitos funcionales del sistema a través de casos de uso.
- promueve el diseño temprano de una arquitectura robusta como base del sistema.
- el desarrollo se hace en iteraciones (flujo de trabajo) e incrementos que permiten obtener versiones parciales del sistema, probarlas y mejorarlas.

arquitectura: conjunto de decisiones significativas acerca de la organización de un sistema software, selección de los elementos estructurales, el sistema, las interfaces entre ellos, su comportamiento, sus colaboraciones, y su composición.

se basa en **6 principios claves**:

1) adaptar el proceso: tiene que adaptarse a las necesidades del cliente.
2) equilibrar prioridades: encontrar un equilibrio que satisfaga los diversos requisitos de los participantes.
3) colaboración entre equipos: comunicación y coordinación fluida entre equipos.
4) demostrar valor iterativamente: en cada iteración se analizan opiniones, estabilidad y calidad, refinan la dirección y riesgos.
5) elevar el nivel de abstracción: motiva el uso de conceptos reutilizables, reduce la complejidad.
6) enfocarse en la calidad: asegurarla en cada aspecto de la producción, es parte del proceso.

se divide en **4 fases**:

1) inicio: define el alcance del proyecto y su viabilidad, se realiza en poco tiempo, qué vamos a hacer, planificar el proyecto.
2) elaboración: reducir riesgos y definir arquitectura, se elabora y orienta el plan de proyecto en detalle, cómo se va a hacer, adaptar el diseño.
3) construcción: construir el producto completo o en versión funcional, el resultado es un sistema ejecutable, se hacen las primeras pruebas, distribuirlo.
4) transición: entregar el sistema al usuario final, implementación y realización de capacitaciones.

artefactos: en cada fase se realiza una serie de entregables que ayudan a comprender el análisis y diseño del sistema, ejemplos: documentación, planes, códigos, informes, capacitaciones, etc.

roles:

- analistas: analistas del sistema, especificador de requisitos, etc.
- desarrolladores: diseñador, implementador, arquitecto de software, etc.
- gestores: jefes, ingeniero de procesos, gestor de pruebas, etc.
- apoyo: documentador, administrador, artista gráfico, etc.
- especialista en pruebas: tester, diseñador de pruebas, etc.

METODOLOGÍAS ÁGILES

Manifiesto de cuatro puntos se basa en valorar:

- a los individuos y su interacción por encima de procesos y herramientas;
- el software que funciona por encima de la documentación exhaustiva;
- la colaboración con el cliente por encima de la negociación;
- la respuesta al cambio por encima del seguimiento de un plan.

Las **metodologías ágiles** son una serie de estrategias que buscan la mejora continua empresarial, llamadas (Gestión de Proyectos Ágil); aplicado a un mercado tecnología se traduce en constante innovación y renovación, se implementan durante la ejecución de las actividades y se focalizan en el cliente.

Los **12 principios** del Manifiesto Ágil:

1) el cliente como objetivo: importante conocer opiniones que ayuden a mejorar el producto.
2) flexibilidad: procesos sujetos a cambios constantemente.
3) producción de prototipos: funciones presentadas individualmente para implementarle los cambios.
4) integración de los miembros del equipo: las relaciones entre el equipo y sus tareas deben tenerse en cuenta.
5) motivación: equipo motivado garantiza cumplimiento del objetivo.
6) comunicación: los miembros del proyecto deben conocer las decisiones y necesidades.
7) funcionalidad como indicador de progreso: producto que funcione correctamente y se adapte a las necesidades.
8) soporte: garantiza correcta funcionalidad de los mismos después de haber sido entregados.
9) excelencia: da calidad y efectividad en los resultados.
10) sencillez: permiten mayor nivel de control y análisis.
11) coordinación: interdependencia de las tareas implica coordinación de las mismas.
12) aprendizaje: cada etapa implica un periodo de reflexión con el fin de analizar qué funcionó y qué no, permite proponer soluciones y cambios.

pasos y etapas:

- análisis de los requisitos: se analizan demandas y necesidades del mercado, se proponen objetivos y enfoques.
- planificación de actividades: división de responsabilidades, tiempos límites, análisis.
- reunión del equipo: analizar plan diseñado, se proponen posibles cambios.
- producción: se empieza el desarrollo de sprints, implementan modificaciones.
- análisis, documentación y lanzamiento: se analizan y documentan las líneas de producción.

principales roles:

- agile coach: capacitación de capital humano y organización, garantiza calidad y excelencia de los medios de trabajo para que los miembros del equipo los usen bien.
- scrum master: garantizador de que se cumpla el esquema planificado de actividades y objetivos, guía al equipo.
- project manager: gestiona el proyecto en ejecución, conoce cada proceso y actividades para responder ante situaciones que se presenten, dirige al equipo.
- product owner: encargado del correcto desarrollo, está en contacto permanente con el equipo y el cliente y determina prioridades.

ventajas: flexibilidad para adaptarse a cambios, entrega continua de valor, agilidad en la producción, fomenta innovación y la mejora, incrementa la competitividad.
desventajas: requiere seguimiento constante, puede generar incertidumbre, cambio continuo puede afectar la estabilidad, depende del compromiso y madurez del equipo.

principal diferencia entre las metodologías tradicionales y las metodologías ágiles es que en las primeras el proceso es lineal y secuencial, mientras que en las segundas es proceso es iterativo. Enfoque predictivo vs adaptativo.

1) **método scrum**: es un framework de gestión de proyecto que ayuda a los equipos a estructurar y gestionar el trabajo con un conjunto de valores, principios y prácticas. Incluye reuniones, herramientas y funciones. El software está en constante evolución y cambio a través de Sprints.
sus roles son:
- scrum master: lidera y sirve a los equipos, promueve autoorganización y responsabilidad, ayuda a solucionar situaciones.
- development team: tiene habilidades para crear funcionalidades incrementales, miembros responsables, no hay roles ni subequipos. todos a labrar
- product owner: maximiza el valor del trabajo del equipo, conocedor del negocio, conoce métricas para medir y validar el ~~impacto~~ valor.

los sprints son periodos breves de tiempo fijo en el que el equipo trabaja para completar una cantidad de trabajo determinada
los protocolos son planificación del sprint, reunión rápida diaria, revisión del sprint, retrospectiva del sprint y refinamiento del sprint.
el backlog de un producto es una lista de trabajo ordenado por prioridades para el equipo de desarrollo que se obtiene de la hoja de ruta y sus requisitos

2) **método kanban**: ~~se basa en la continua mejora, se implementa a través de tableros~~, es un método visual de gestión que permite visualizar los flujos de trabajo y la carga; el proyecto se muestra en forma de tablero donde cada columna es una etapa del trabajo y las tareas individuales se muestran con tarjetas visuales.
diferencia scrum de kanban: son compatibles, pero scrum se centra en finalizar más trabajos y más rápido.
~~tiene 4 principios: empieza con lo que haces ahora, comprométete a buscar e implementar cambios progresivos y evolutivos, respeta los procesos, roles y responsabilidades actuales, impulsa el liderazgo en todos los niveles.~~
~~tiene 6 prácticas: visualizar el trabajo, limitar el trabajo en curso, gestionar el flujo de trabajo, implementar políticas de procesos explícitas, implementar ciclos de comentarios, mejorar colaborando y evolucionar experimentando.~~

3) **método programación extrema XP**: se centra en la velocidad y la simplicidad con ciclos de desarrollo cortos, tiene estructura rígida, pero con sprints altamente centrados e integraciones continuas. El diseño de XP permite a los desarrolladores responder a solicitudes de los clientes, adaptarse y realizar cambios en tiempo real. Es disciplinada, creativa y colaborativa.
el ciclo de vida fomenta la integración continua, ya que requiere que los miembros del equipo trabajen casi constantemente.
~~sus 5 valores son: simplicidad, comunicación, comentarios, valentía y respeto~~.

sus 5 reglas son: planificación, gestión, diseño, codificación y prueba.
sus 12 prácticas son: juego de planificación, pruebas de clientes, pequeñas entregas, diseño simple, programación en parejas, desarrollo guiado por pruebas, refactorización, propiedad colectiva, integración continua, ritmo de trabajo sostenible, metáfora y estándares de codificación.
diferencia scrum de xp: es más rígido porque tiene reglas y pautas estrictas que promueven intercambios constantes entre clientes y desarrolladores.

mini resumen:
tipos de metodologías ágiles:

- método scrum: está enfocada en la gestión, se caracteriza por la división de responsabilidades, su elemento fundamental es el Sprint.
- extreme programming XP: enfocada en parte técnica, hace uso de la iteración, establece relaciones perdurables entre el cliente y trabajadores y entre integrantes del proyecto.
- kanban: divide el desarrollo en etapas dispuestas para que destaquen las más importantes a terminar, se recurre a un tablero de tres columnas y no es necesario el uso de sprints y/o iteraciones, pero si tarjetas que representan importancia de tareas.