CONCEPTOS BÁSICOS

Tipos de software:

Software: conjunto de componentes lógicos los cuales pueden ser modificados y son necesarios para realizar tareas en específico.

·**De sistema:** (sistema operativo, drivers, controladores) interactúa directamente con el hardware es el software que interactúa con el hardware, es de bajo nivel

·**De aplicación:** (suite informática, navegador, edición de imagen) conjunto de programas que el usuario usa más habitualmente es el más usado

·**De desarrollo:** (editores, compiladores, intérpretes) son programas los cuales permiten crear y desarrollar varias aplicaciones y webs

---

·**Firmware:** es un programa el cual es el encargado de controlar todo el hardware de un micro dispositivo, es el software más cercano al hardware.

Ejemplo: cualquier tipo de aparato con minicontrolador.

·**Driver:** entre software y hardware, permite comunicar con firmware/hardware y recibe órdenes del sistema operativo.
Jerarquía de memorias.

Relación Hardware-Software

·**Disco Duro:** es la memoria secundaria y almacena de forma permanente los archivos tanto ejecutables como datos.

·**Memoria ram:** es la memoria principal y es una memoria volátil la cual almacena de forma temporal en binario archivos ejecutables y los archivos de datos necesarios.

·**CPU:** lee y ejecuta instrucciones de la RAM así como todos los datos necesarios, además de tener memoria caché.

·**E/S:** la entrada recoge información del exterior y la lleva al interior, la salida lleva información del exterior y la lleva al exterior.

Código fuente, objeto y ejecutable

·**Código fuente:** es el código del programa entendible por un ser humano para hacer un programa.

·**Código objeto:** es un archivo binario, se genera a partir del código fuente y no se puede ejecutar.

·**Código ejecutable:** es un archivo el cual si se ejecuta en el sistema operativo se completa su tarea

Solo para Lenguaje compilado

·**Lenguaje compilado:** simplifica el lenguaje para que varios dispositivos diferentes

·**Lenguajes interpretados:** no usa compilador, usan scripts.

-Script: son líneas de comando que se ejecutan de manera ordenada.

CICLO DE VIDA SOFTWARE:

Ingeniería del software: Disciplina que estudia los principios y metodologías para el desarrollo y mantenimiento de sistemas software.

Desarrollo de software:
·**Análisis:** se determina y analizar requisitos
·**Diseño:** implementa idea, donde se define la arquitectura
·**Codificación:** programar código
·**Pruebas:** se comprueba y se buscan errores en el programa
·**Mantenimiento:** se implementan parches y actualizaciones

Especificaciones de requisitos:
·Ser completa y sin omisiones
·Ser concisa y sin trivialidades
·Evitar ambigüedades. Usar lenguaje formal
·Evitar detalles de diseño e implementación
·Ser entendible por el cliente
·Separar requisitos funcionales y no funcionales
·Dividir y jerarquizar el modelo: dividir en partes el diseño planeado
·Fijar criterios de validación: poner límites donde no se puede sobrepasar el diseño específico.

Diseño:
· Descompone y organiza el sistema en elementos componentes que pueden llegar a ser desarrollado por separado.

·Especifica la interrelación y funcionalidad de los elementos componentes.

Actividades habituales:
·Diseño arquitectónico. Definir como solucionar el programa
·Diseño detallado: definir los problemas por bloques
·Diseño de datos: especifica los datos usado en la aplicación
·Diseño de interfaz de usuario: es lo que el usuario verá y interactuara

Codificación:
• Se escribe el código fuente de cada componente.
• Pueden utilizarse distintos lenguajes informáticos:
· **Lenguajes de programación:** C, C++, Java, Javascript, ...
·**Lenguajes de otro tipo:** HTML, XML, ...

Pruebas:
·Su objetivo es conseguir información en el funcionamiento del programa.

·Se debe de poner al programa en varias situaciones.

Mantenimiento:
·Se realizan cambios a lo largo de cambios, para ello hay que rehacer parte del trabajo realizado en las frases previas.

Tipos de mantenimiento:
·**Correctivo:** corrige defectos
·**Perfectivo:** mejorar funcionalidad
·**Evolutivo:** añade funcionalidades nuevas
·**Adaptativo:** adaptarse a nuevos entornos.
·**Preventivo:** se corrigen defectos antes de que se produzcan altercados.

Resultado tras cada frase:

·**Ingeniería de sistemas:** especificación de sistema

·**Análisis:** especificación de requisitos del software

·**Diseño arquitectónico:** documento de arquitectura del software

·**Diseño detallado:** especificación de módulos y funciones

·**Codificación:** código fuente

·**Pruebas de unidades:** módulos utilizables

·**Pruebas de integración:** sistemas utilizable

·**Pruebas de sistema:** sistema aceptado

·**Documentación:** documentación técnica de y de usuario

·**Mantenimiento:** informes de errores y control de cambios

Modelos del desarrollo del software

Modelos clásicos:

- Cascada
- Modelo en V

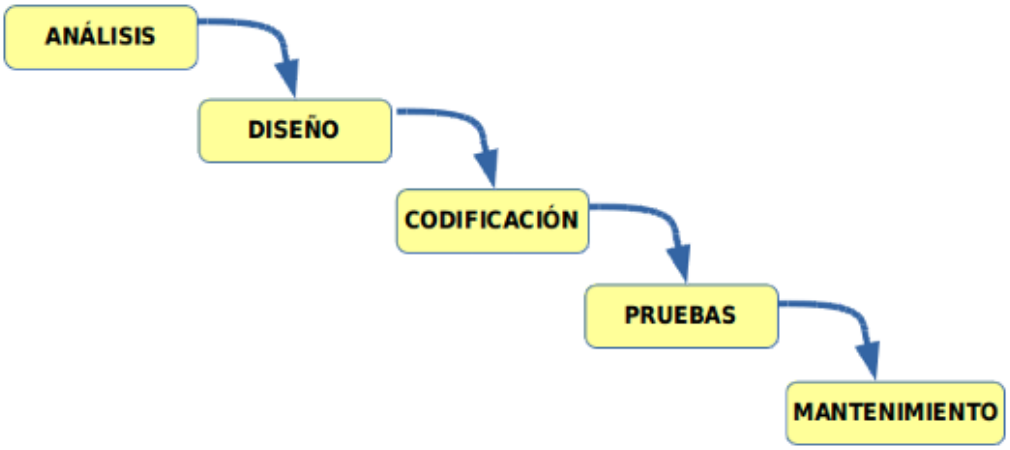
Modelo de construcción de prototipos

Modelo evolutivo o incrementales

- **Modelo en espiral** (iterativos: repetición de operación hasta que se cumpla la condición)
- **Metodologías ágiles** (adaptativos: diseñado para adaptarse y cambiar de forma sin intervención del programador)

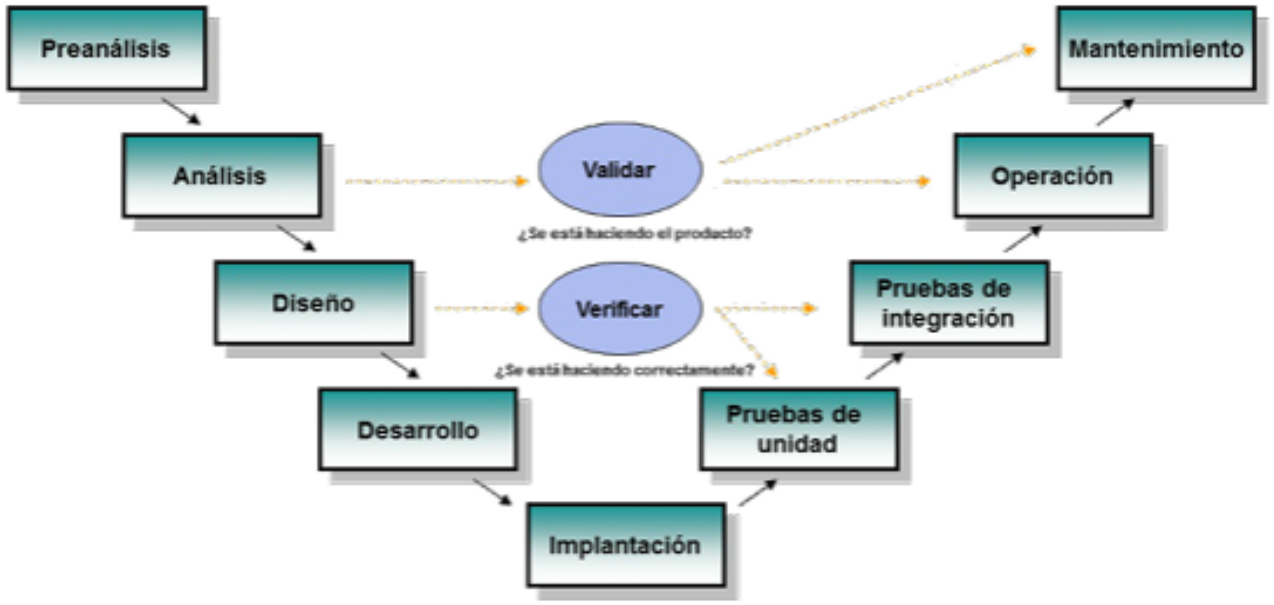
Modelo de cascada:

- El modelo más antiguo
- Identifica las fases principales del desarrollo software
- Cuando se termina una fase se comienza la siguiente
- Muy rígido y de mala adaptación
- Tiene variantes con mayor y menor cantidad de pasos



Modelo en V

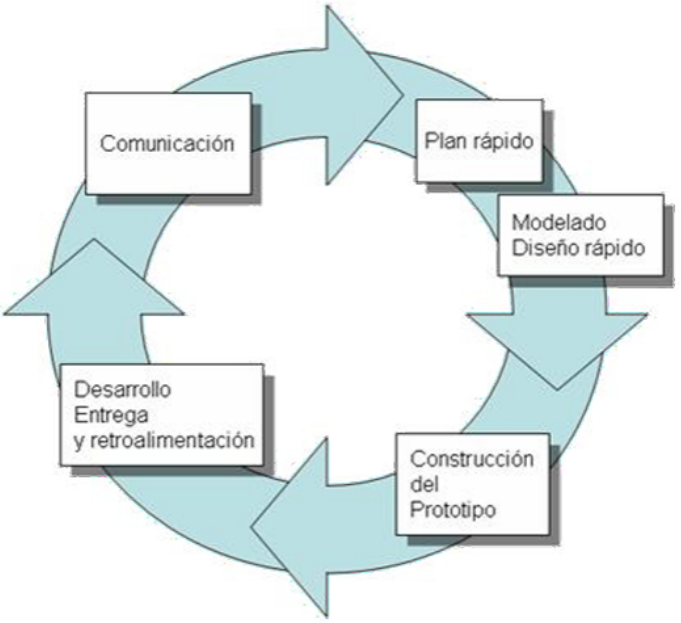
·Parecido al modelo en cascada
·Jerarquizada en distintos niveles
·Niveles superiores indican mayor abstracción
·Niveles inferiores indican mayor detalles
·El resultado de una fase es el inicio de la siguiente
·Existen diferentes variantes con mayores y menor cantidades de actividades



Prototipos

A Menudo los requisitos no están especificados claramente:

·por no existir experiencia previa
·por omisión o falta de concreción del usuario/cliente



Proceso:

·En la fase de análisis se crea un prototipo el cual es aprobado por el usuario/cliente para refinar el software.

·Se repite el proceso anterior todo lo que sea necesario.

Tipos de prototipos:

·Prototipos rápidos:

- ·Puede estar desarrollado usando lenguaje y/o herramientas
- ·Cuando se termina de usar de desecha

·Prototipos evolutivos

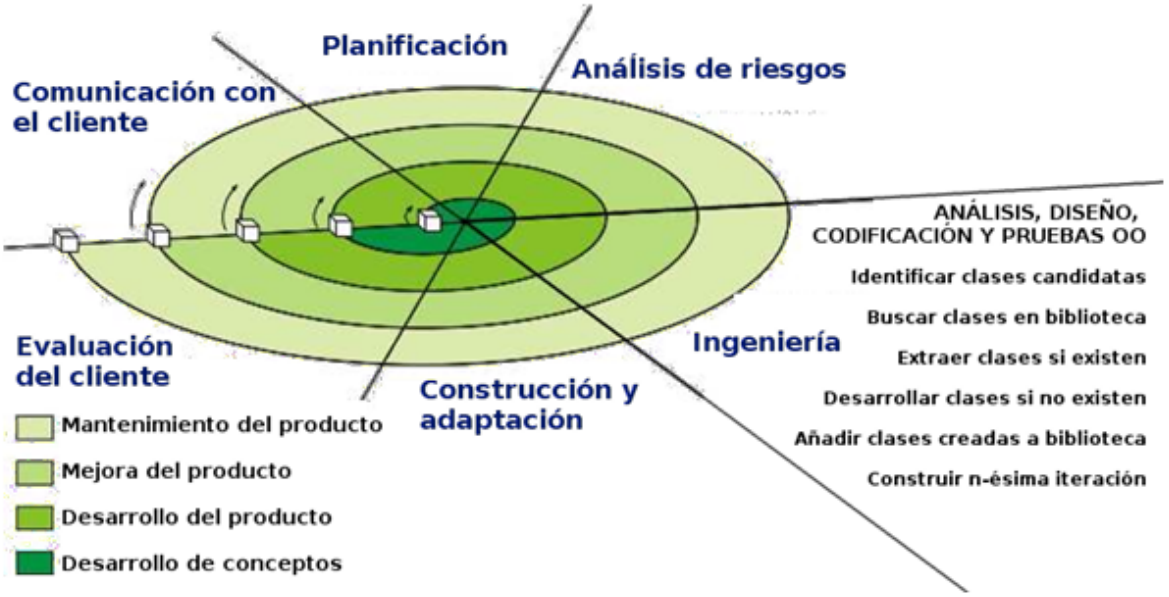
- ·Está diseñado con el mismo lenguaje y herramientas del proyecto
- ·El prototipo se usa como base para desarrollar el proyecto

Modelo en espiral:

·En la ingeniería corresponde a las fases de los modelos clásicos: análisis, diseño, codificación, etc.

·Se aplica la programación orientada a objetos

- ·En la actividad de ingeniería se da gran importancia a la realización de código.



Metodologías ágiles

·Son métodos de ingeniería del software el cual se toma como objetivo el desarrollo de iterativo e incremental.

·Los objetivos y soluciones van cambiando según el tiempo que necesite el proyecto.

·El trabajo es realizado mediante la colaboración de equipos organizados automáticamente y multidisciplinarios, inmersos en un proceso en común de toma de decisiones a corto plazo.

·Los métodos más reconocidos son:

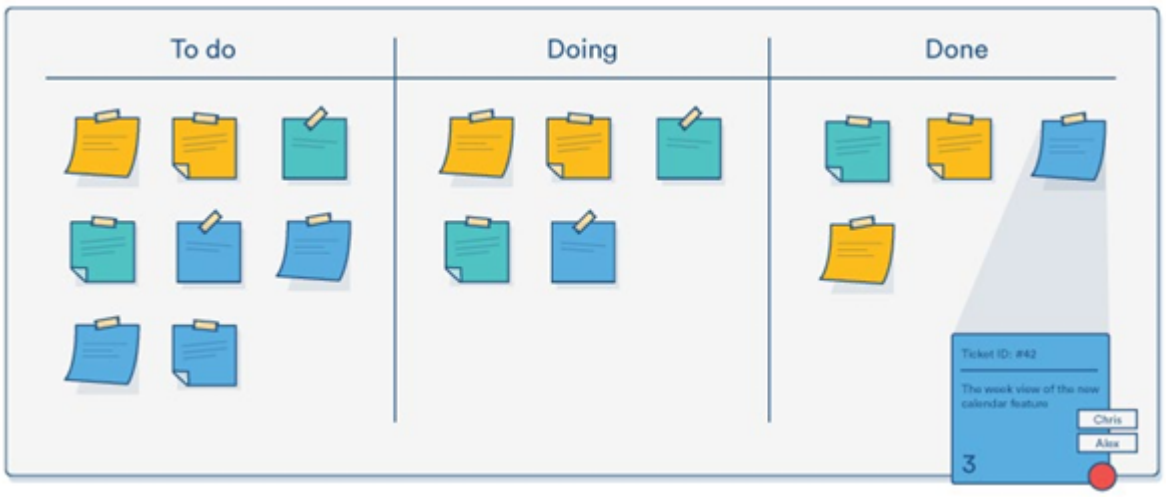
- **·Kanban**
- **·Scrum**
- **·XP(eXtreme Programming)**

Manifiesto por el Desarrollo Ágil

·**Individuos e interacciones** sobre herramientas y procesos
·**Software funcionando** sobre documentación extensiva
·**Colaboración con el cliente** sobre negociación contractual
·**Respuesta ante el cambio** sobre seguir un plan

Kanban

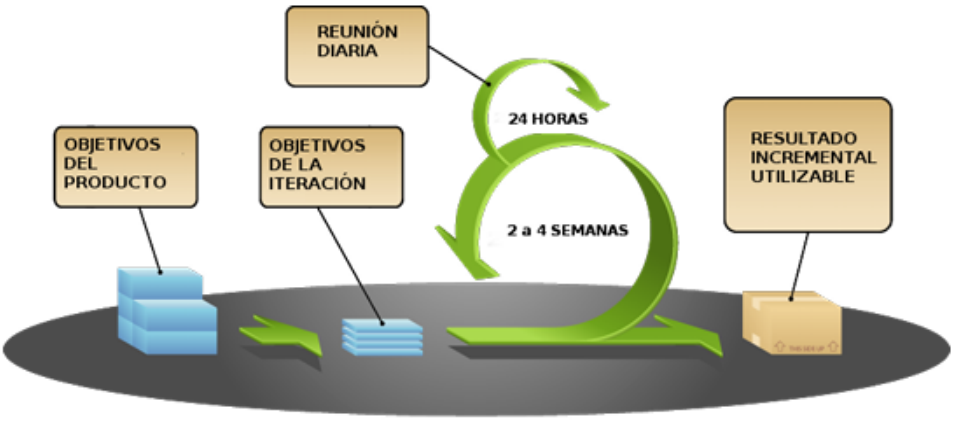
·También llamadas “Sistema de tarjetas”
·Desarrollado originalmente por Toyota
·Controla por demanda de la fabricación de los productos necesarios en la cantidad y tiempo necesario
·Se enfoca en entregar lo máximo utilizando recursos mínimos



Scrum

·Desarrolla el proyecto de manera progresiva.
·Se producen sprints regulares cada 2 a 4 semanas.
·Al principio de cada sprint se establecen objetivos.
·Cuando se acaba cada sprint el cliente recibe una entrega parcial utilizable.

·Se hacen reuniones diarias de como va el progreso del sprint.



Roles principales del Scrum:

·**Product Owner:** define los criterios de aceptación y se asegura de que se cumpla.
·**Scrum Master:** comprueba que se este siguiendo la metodología scrum, motiva y facilita el trabajo del equipo.
·**Team:** es el equipo el cual desarrolla de manera auto-organizada y multifuncional, puede ser hace normalmente entre 6 y 10 miembros.

Artefactos de Scrum
Product Backlog: lista ordenada con los requisitos del productos
Sprint Backlog: lista de requisitos sacados del backlog para su desarrollo durante el sprint
Incrementó: estado del producto después de cada sprint.

Eventos de Scrum

·**Sprint:** evento el cual contiene al resto de eventos. Dura como máximo 1 mes.

·**Sprint Planning:** reunión donde se planea el sprint. Dura un máximo de 8 horas.

·**Daily Scrum:** reunión diaria en la que se habla como va el sprint.

·**Scrum Review:** reunión final en la cual se evalúa el incremento obtenido. Tiene una duración máxima de 4 horas.

·**Scrum Retrospective:** reunión final en la que se evaluar la correcta evaluación. Tiene un máximo de 3 horas.

Valores de XP (Programación eXtrema)

·Simplicidad.
·Comunicación.
·Retroalimentación.
·Valentía o coraje.
·Respeto o humildad.

Características de XP (Programación eXtrema)

·Diseño sencillo.
·Pequeñas mejoras continuadas.
·Pruebas y refactorización.
·Integración continua.
·Programación por parejas.
·El cliente se integra en el equipo de desarrollo.
·Propiedad del código compartido.
·Estándares de codificación.
·40 horas semanales.

LENGUAJE DE PROGRAMACIÓN

Obtención de código ejecutable

·Para conseguir código binario ejecutable tenemos 2 opciones:
    ·Compilar: traducimos el código a un código entendible por máquinas.
    ·Interpretar: el código no es traducido y es ejecutado directamente.

Proceso de compilación/interpretación

·La compilación/interpretación de código se realiza en dos pasos:
    1.Análisis léxico: se mira si las palabras y los terminos estan correctos
    2.Análisis sintáctico: analiza si el orden es el correcto

·Un código correctamente escrito no significa que el resultado deseado

Lenguajes compilados

Algunos ejemplos son: C y C + +.
·Ventaja: es eficiente.
·Desventaja: cada vez que el código fuente es modificado es necesario compilar.

Lenguajes interpretados

Algunos ejemplos son: PHP, Javascript
·Ventaja: es interpretado de manera directa
·Desventaja: es menos eficiente

Java
·Es un híbrido de compilado y interpretado
·El código de java se compila y después obtiene un código binario el cual se llama bytecode
·Java funciona de tal manera que la máquina virtual de java traduce el código objeto a bytecode

Ventajas:
·Estructurado y orientado a objetos
·Es fácil de aprender
·Buena documentacion y base de usuarios

Desventajas:
·Menos eficiencia a comparación de los lenguajes compilados

Tipos:

Puede ser de 2 maneras:
·Declarativo: se da el resultado sin dar los pasos
·Imperativo: se da el resultado con los pasos
Tipos de lenguajes declarativos:

·Lógicos: usan reglas (Prolog)
·Funcionales: usan funciones (Lisp)
·Algebraicos: usan sentencias (SQL)

·Normalmente son interpretados

Tipos de lenguaje imperativos:

·Estructurados: C
·Orientados a objetos: Java
·Multiparadigma: C++, Javascript

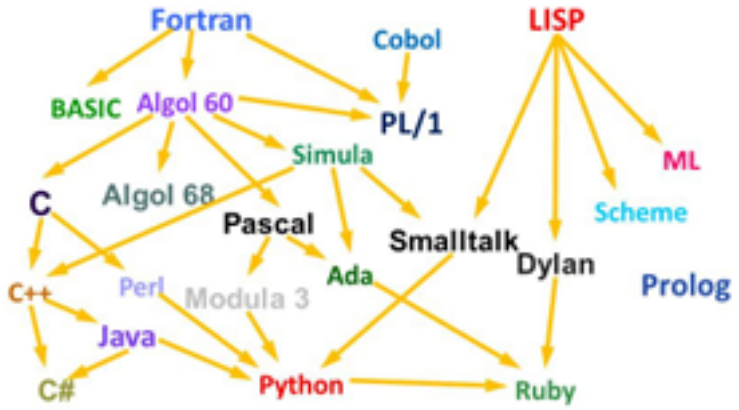
·Lenguaje estructurado= lenguaje objeto
·Muchos de ellos son compilados

Tipos de lenguajes por nivel:

·Bajo nivel: ensamblador
·Medio nivel: C
·Alto nivel: C++, Java

Evolución:
·Código binario
·Ensamblador
·Lenguajes estructurados
·Lenguajes orientados a objetos

Historia:



Criterios para la selección de un lenguaje

·Campo de aplicación
·Herramientas de desarrollo
·Base de usuarios
·Transportabilidad
·Experiencia previa
·Documentación disponible
·Reusabilidad
·Imposición del cliente