

<b>Tipos de software</b>	<b>2</b>
<b>Relación hardware-software</b>	<b>2</b>
<b>Códigos fuente, objeto y ejecutable</b>	<b>2</b>
<b>Desarrollo de software</b>	<b>2</b>
<b>Análisis</b>	<b>2</b>
<b>Diseño</b>	<b>2</b>
<b>Codificación</b>	<b>3</b>
<b>Pruebas</b>	<b>3</b>
<b>Documentación</b>	<b>3</b>
<b>Mantenimiento</b>	<b>3</b>
<b>Resultado de cada fase</b>	<b>3</b>
<b>Modelos de desarrollo de software</b>	<b>3</b>
<b>Modelo en cascada</b>	<b>4</b>
<b>Modelo en V</b>	<b>5</b>
<b>Prototipos</b>	<b>6</b>
<b>Modelo en espiral</b>	<b>7</b>
<b>Metodologías ágiles</b>	<b>7</b>
<b>Kanban</b>	<b>7</b>
<b>Scrum</b>	<b>8</b>
<b>XP (PROGRAMACIÓN EXTREMA)</b>	<b>9</b>
<b>LENGUAJE DE PROGRAMACIÓN</b>	<b>10</b>
<b>HISTORIA</b>	<b>10</b>
<b>CRITERIOS PARA LA SELECCIÓN DE UN LENGUAJE</b>	<b>11</b>

# Tipos de software

- De sistema (Sistema operativo, drivers, antivirus, firmware, etc...)
- De aplicación (Suite ofimática, Navegador, Edición de imagen, Multimedia, Diseño gráfico, etc...)
- De desarrollo (Editores, compiladores, intérpretes, etc...)

## Relación hardware-software

- Disco duro: almacena de forma permanente los archivos ejecutables y los archivos de datos.
- Memoria RAM: almacena de forma temporal el código binario de los archivos ejecutables y los archivos de datos necesarios.
- CPU: lee y ejecuta instrucciones almacenadas en memoria RAM, así como los datos necesarios.
- E/S: recoge nuevos datos desde la entrada, se muestran los resultados, se leen/guardan a disco, .

## Códigos fuente, objeto y ejecutable

- Código fuente: archivo de texto legible escrito en un lenguaje de programación
- Código objeto: (intermedio): archivo binario no ejecutable.
- Código ejecutable: archivo binario ejecutable

## Desarrollo de software

Fases principales:

- ANÁLISIS
- DISEÑO
- CODIFICACIÓN
- PRUEBAS
- DOCUMENTACIÓN
- MANTENIMIENTO

## Análisis

Se determina y define claramente las necesidades del cliente y se especifica los requisitos que debe cumplir el software a desarrollar.

## Diseño

- Se descompone y organiza el sistema en elementos componentes que pueden ser desarrollados por separado.

- Se especifica la interrelación y funcionalidad de los elementos componentes.
- Las actividades habituales son las siguientes:
  - Diseño arquitectónico
  - Diseño detallado
  - Diseño de datos
  - Diseño de interfaz

## Codificación

Se escribe el código fuente de cada componente. Con lenguajes informáticos.

## Pruebas

- El principal objetivo de las pruebas debe ser conseguir que el programa funcione incorrectamente y que se descubran defectos.
- Deberemos someter al programa al máximo número de situaciones diferentes.

## Documentación

- El principal objetivo de la documentación es explicar como esta realizado el código para que otra persona lo entienda y pueda continuarlo.
- Se especifica lo que realiza para función del programa.

## Mantenimiento

- Durante la explotación del sistema software es necesario realizar cambios ocasionales.
- Para ello hay que rehacer parte del trabajo realizado en las fases previas.
- Tipos de mantenimiento:
  - Correctivo: se corrigen defectos.
  - Perfectivo: se mejora la funcionalidad.
  - Evolutivo: se añade funcionalidades nuevas.
  - Adaptativo: se adapta a nuevos entornos.

## Resultado de cada fase

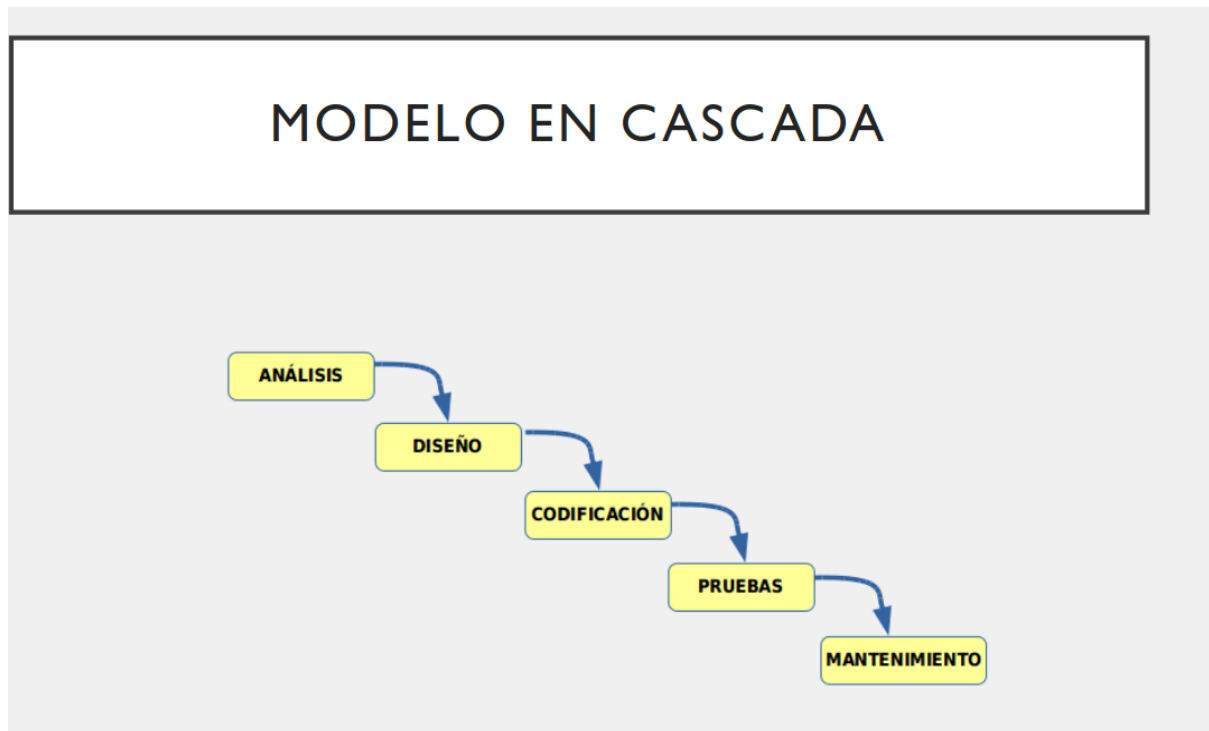
- ANÁLISIS: Especificación de requisitos del software
- DISEÑO arquitectónico: Documento de arquitectura del software
- DISEÑO detallado: Especificación de módulos y funciones
- CODIFICACIÓN: Código fuente

## Modelos de desarrollo de software

- Modelos clásicos (predictivos)

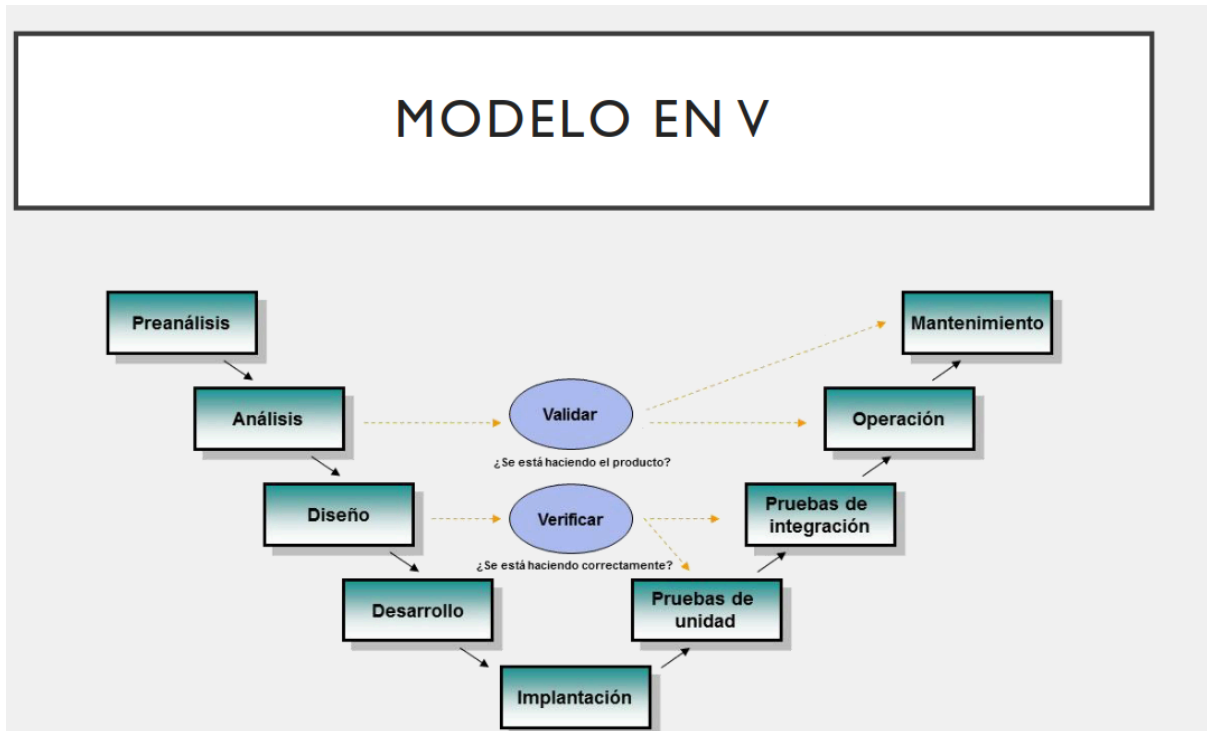
- Modelo en cascada
- Modelo en V
- Modelo de construcción de prototipos
- Modelos evolutivos o incrementales
- Modelo en espiral (iterativos)
- Metodologías ágiles (adaptativos)

## Modelo en cascada

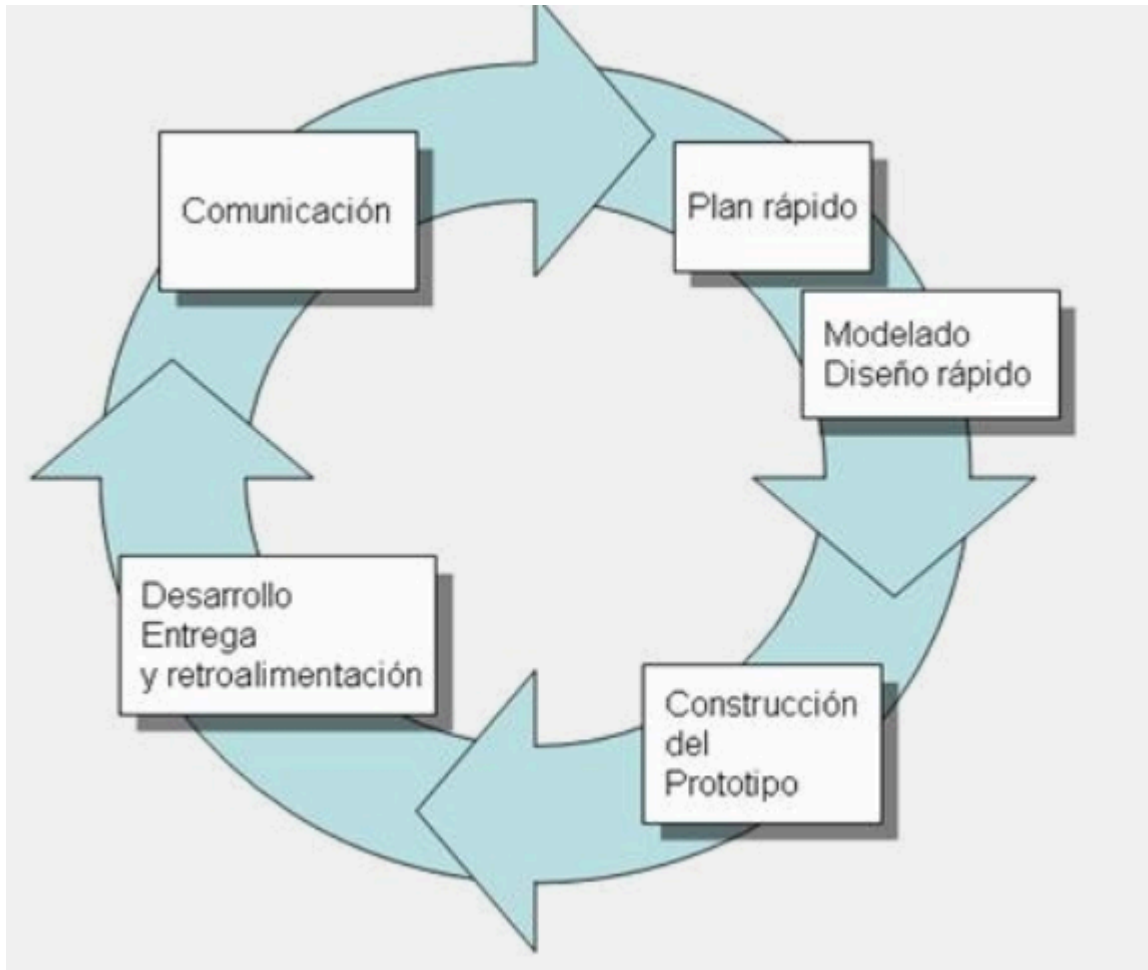


Consiste en ir avanzando después de terminar cada apartado

## Modelo en V



# Prototipos



- Tipos de prototipos:

- Prototipos rápidos

El prototipo puede estar desarrollado usando otro lenguaje y/o herramientas. Finalmente el prototipo se desecha.

- Prototipos evolutivos

El prototipo está diseñado en el mismo lenguaje y herramientas del proyecto. El prototipo se usa como base para desarrollar el proyecto.

Se responde a las tasas de los modelos clásicos.



# Metodologías ágiles

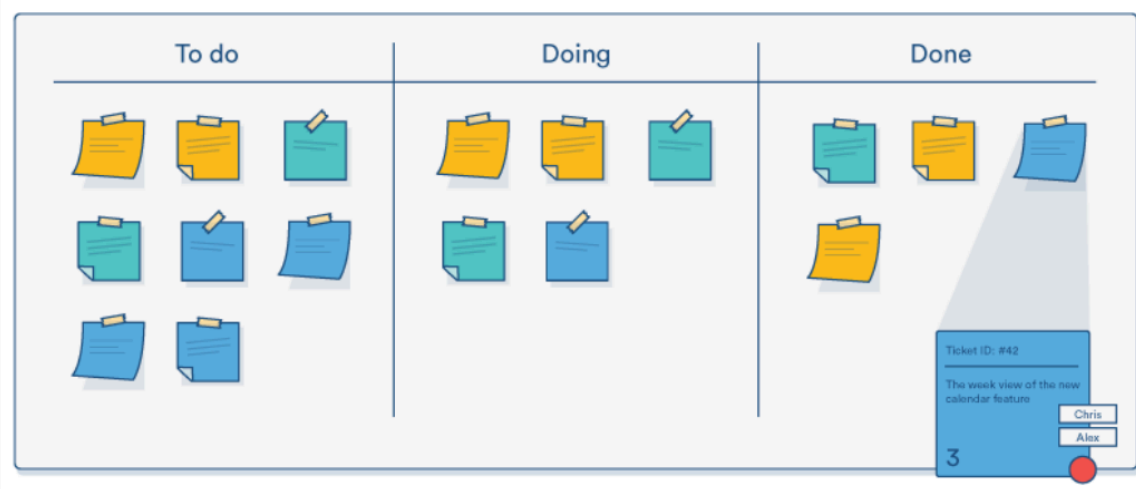
Las metodologías más conocidas son:

- Kanban
- Scrum
- XP (eXtreme Programming)

# Kanban

La **metodología Kanban** es un enfoque ágil para gestionar el trabajo de forma visual, usando tableros que muestran tareas en diferentes estados.

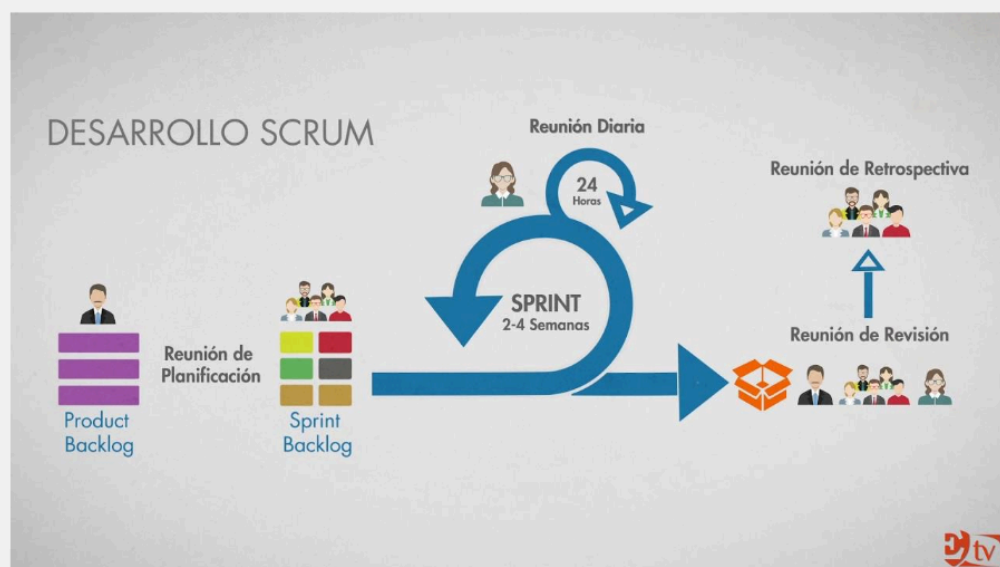
## Pizarra Kanban



## Scrum

La **metodología Scrum** es un marco ágil que organiza el trabajo en **iteraciones cortas llamadas sprints**.

Promueve la **colaboración del equipo**, la entrega frecuente de valor y la mejora continua.





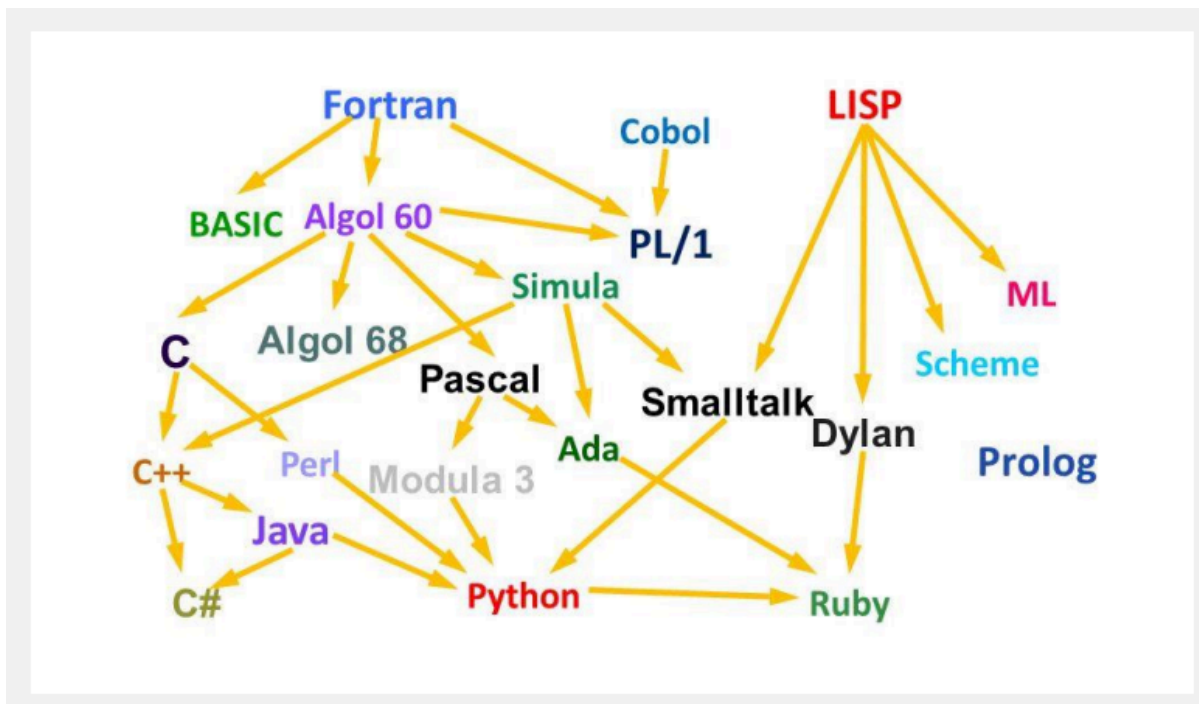
# XP (PROGRAMACIÓN EXTREMA)



# LENGUAJE DE PROGRAMACIÓN



## HISTORIA



# CRITERIOS PARA LA SELECCIÓN DE UN LENGUAJE

- Campo de aplicación
- Experiencia previa
- Herramientas de desarrollo
- Documentación disponible
- Reusabilidad
- Portabilidad
- Imposición del cliente