



# Programación Avanzada

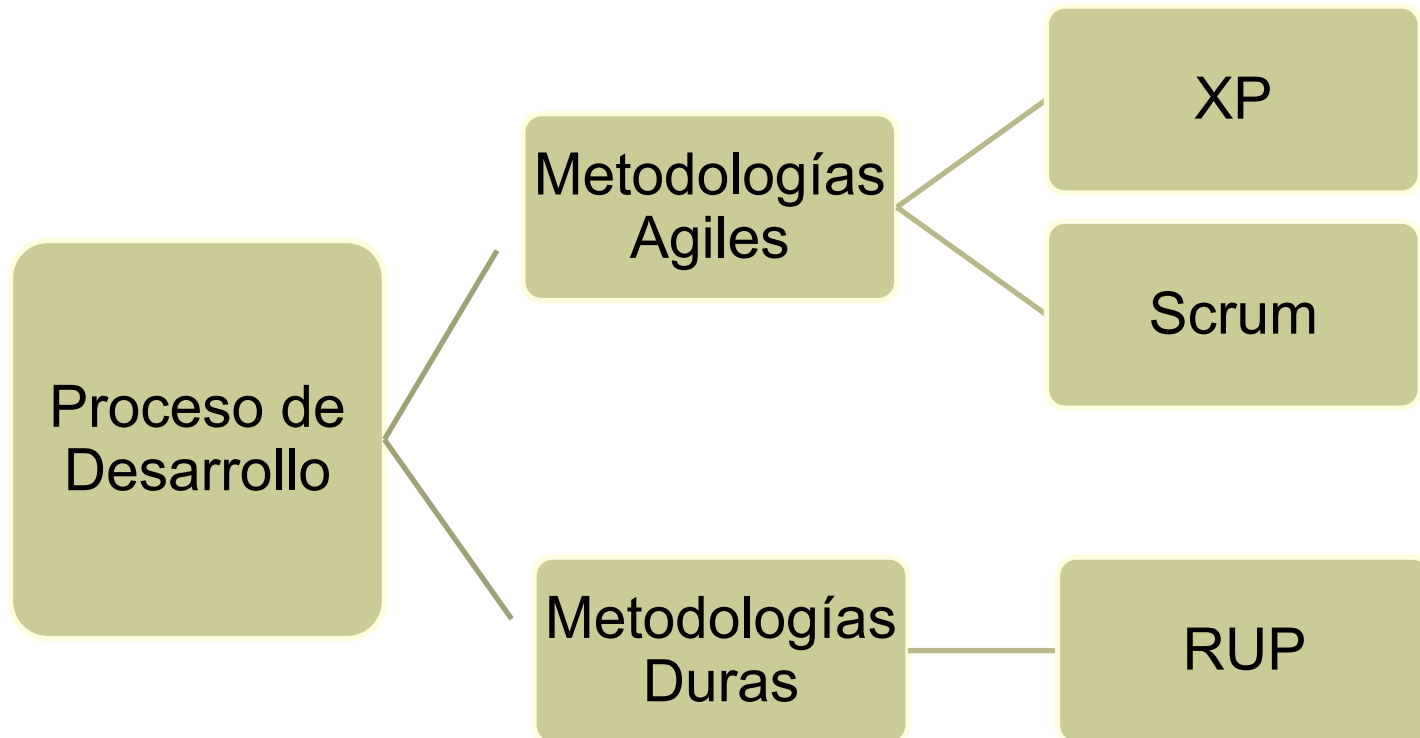
Segundo Cuatrimestre 2021



# Metodología de desarrollo de software

# Desarrollo de Software

El proceso de desarrollo de software debe seguir ciertas etapas que dependen de la metodología elegida

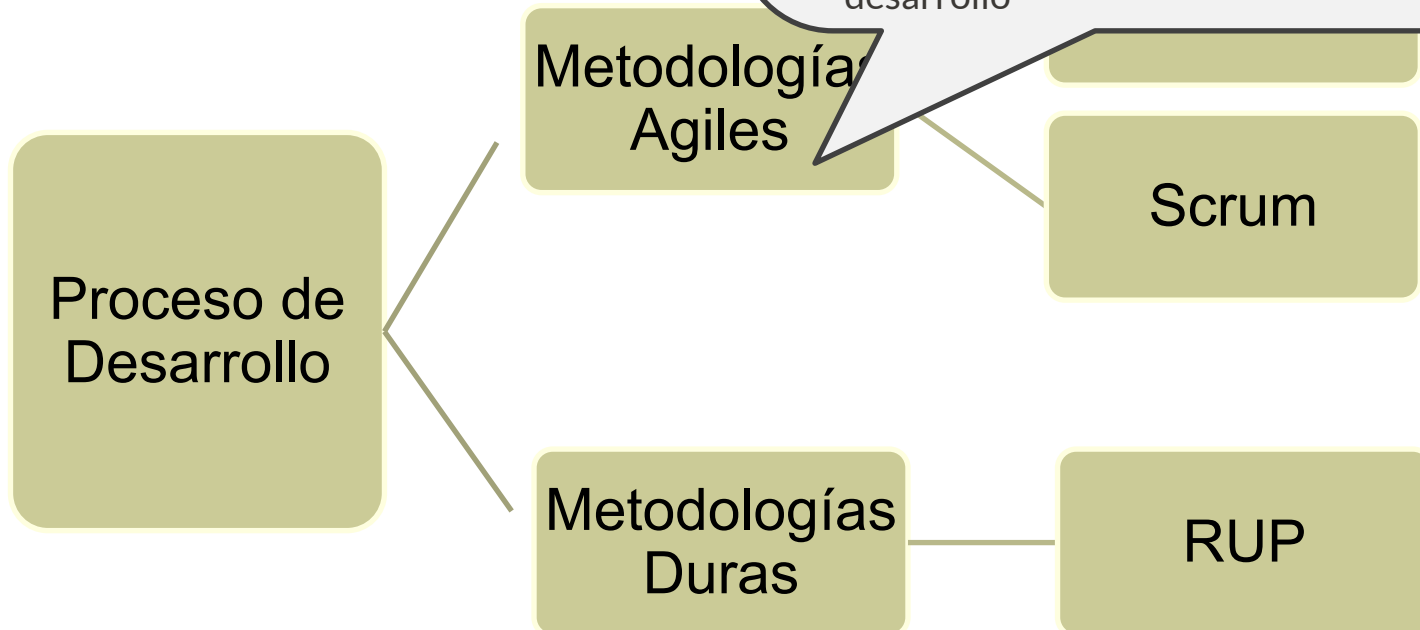


# Desarrollo de Software

El proceso de desarrollo de software dependen de la metodología elegida

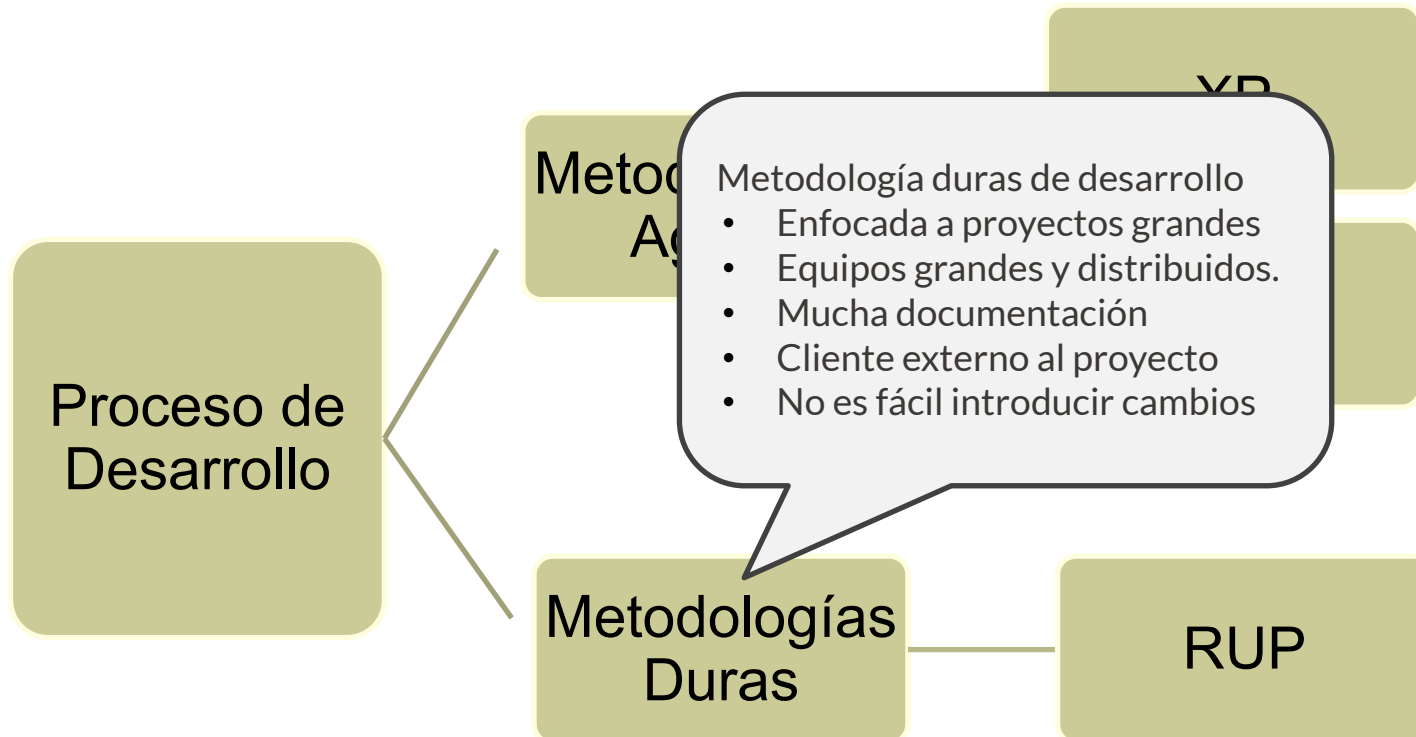
## Metodología ágil de desarrollo

- Enfocada a proyectos que requieren rapidez y flexibilidad.
- Equipos pequeños ( $7 \pm 2$ ), multidisciplinares y localizados en un mismo sitio.
- Métodos adaptativos (rápida adopción de cambios según necesidades).
- Métodos de desarrollo iterativo e incremental.
- Mínima producción de documentación.
- Colaboración constante cliente-equipo desarrollo



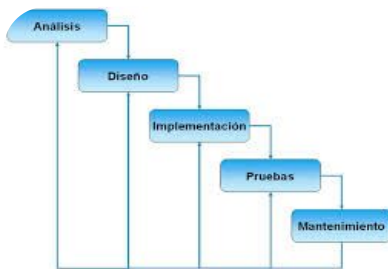
# Desarrollo de Software

El proceso de desarrollo de software debe seguir ciertas etapas que dependen de la metodología elegida

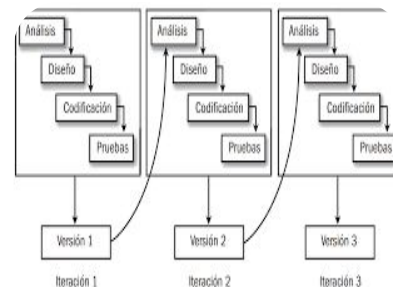


# Modelo de ciclo de vida

Cualquiera sea el proceso utilizado y aplicado en el desarrollo de software se debe aplicar un modelo de ciclo de vida

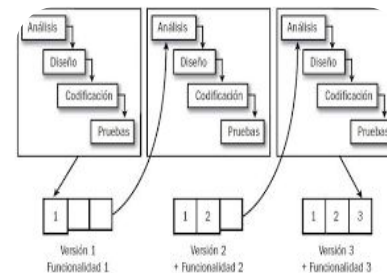


Modelo en cascada



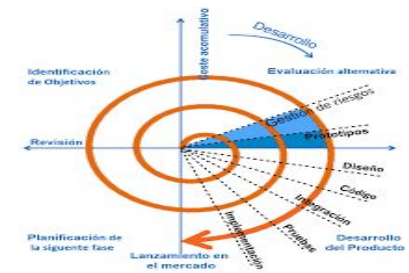
Modelo Iterativo

Al final de cada iteración se entrega una versión mejorada.



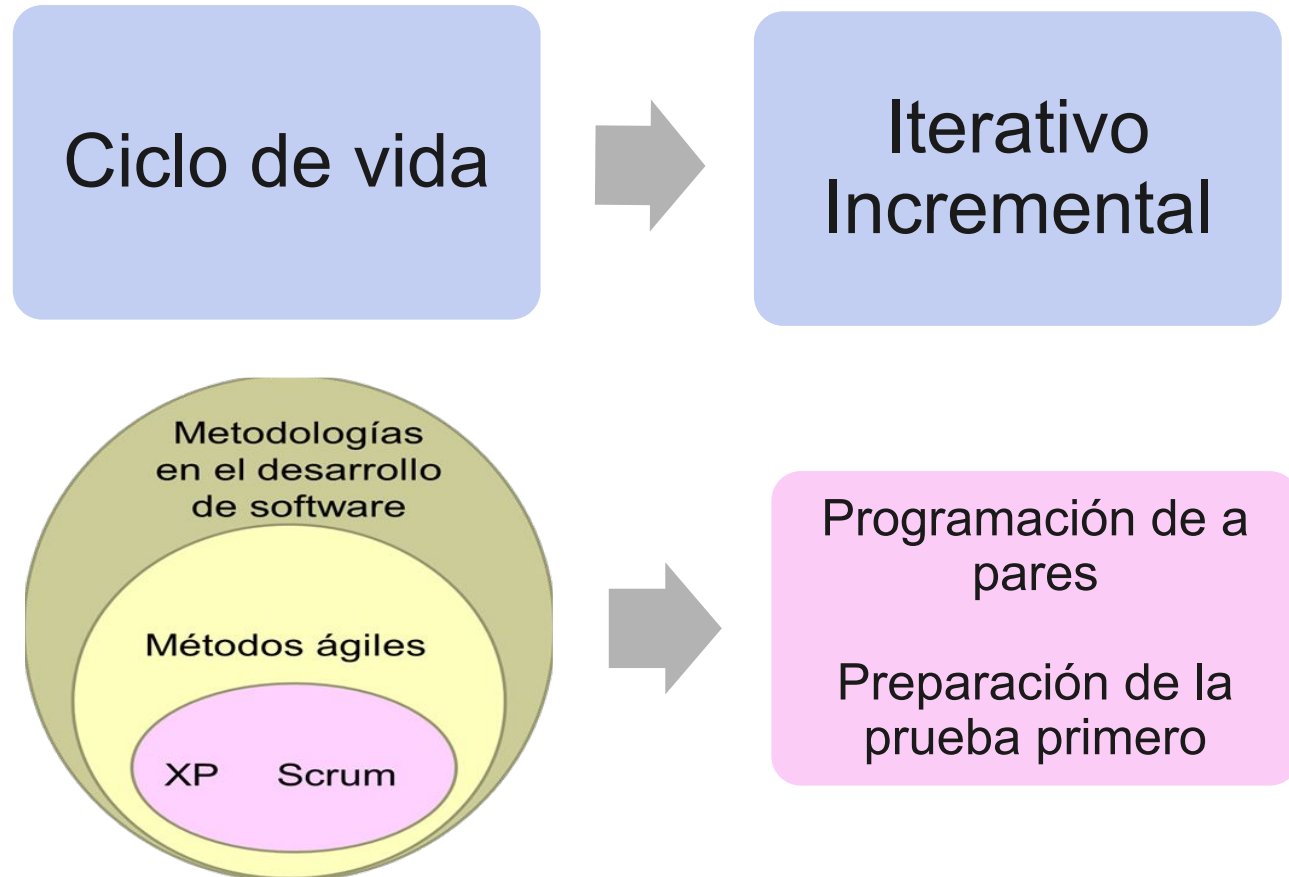
Modelo Incremental

En cada iteración se agregan funcionalidades



Modelo Espiral

# Metodologías aplicadas en el aula



# Proceso Personal de Software (PSP)



Para producir productos de calidad, los ingenieros deben planear, medir y rastrear constantemente los indicadores de calidad del producto.

La calidad debe ser tomada en cuenta desde el principio del proyecto.

Finalmente, deben analizar los resultados de cada trabajo y utilizar estos resultados para mejorar sus procesos personales.



# PSP- Antecedentes



Después de la segunda guerra mundial, la estrategia de calidad en la mayoría de las organizaciones industriales se basaba casi por completo en las pruebas. Las empresas establecieron departamentos especiales de calidad para encontrar y arreglar problemas después de la producción de los productos.

En los años 1970 y 1980 la industria estadounidense fue cambiando centrándose en mejorar la forma en la que la gente hacía sus trabajos y desarrollaban sus procesos. Este enfoque a los procesos de trabajo produjo mejoras importantes en la calidad de muchos productos.

Para el software, la primera medida en este sentido la constituyó la introducción de las inspecciones del software en 1976 y posteriormente en 1987, el Modelo de Capacidad de Madurez (CMM).

Después de que Watts S. Humphrey condujera el desarrollo inicial de este modelo para software, se decidió a aplicar los principios de CMM a los programas pequeños. Así nace PSP, que aplica el proceso de mejora a quienes realizan el trabajo de desarrollo de software, concentrándose en las prácticas de trabajo de los ingenieros en una forma individual.

# Principios del PSP

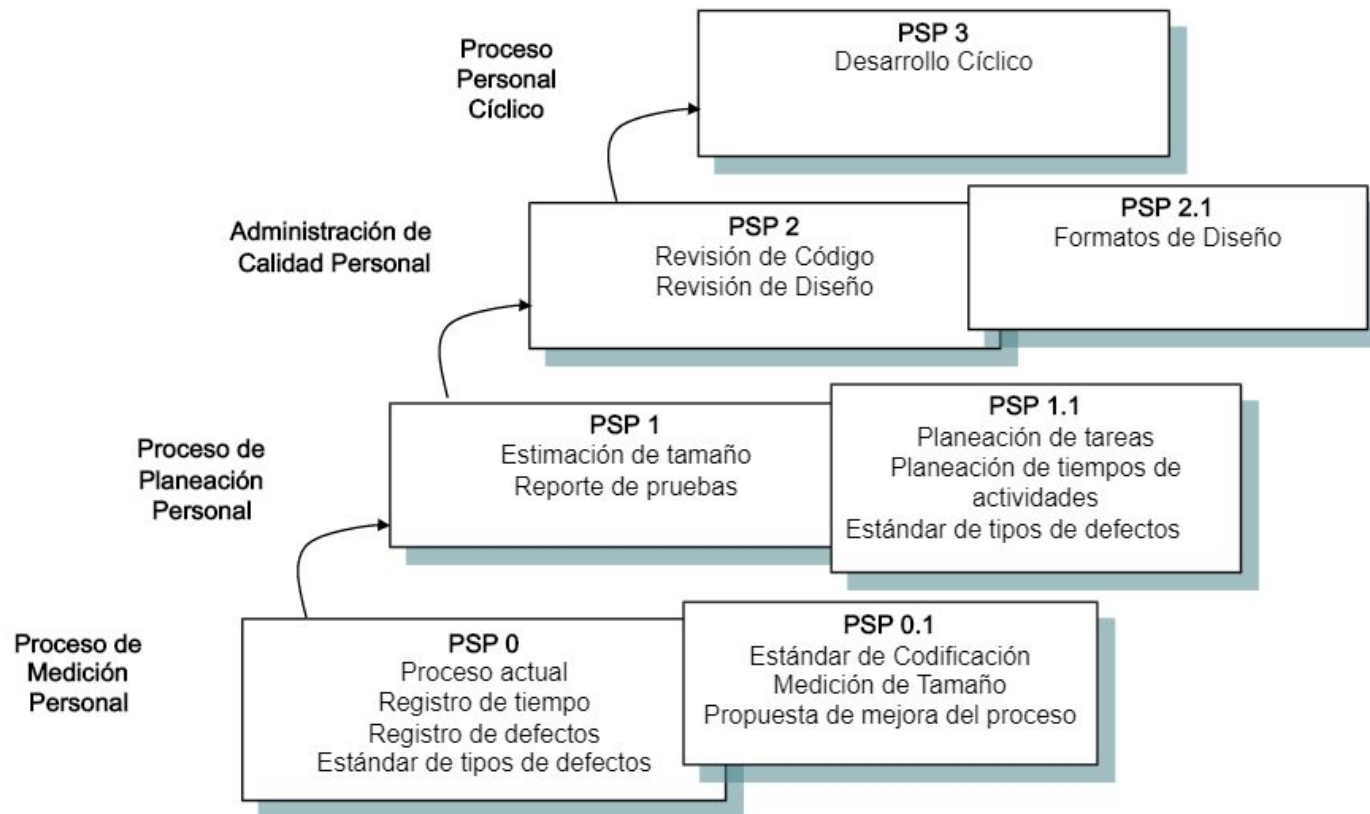


El diseño de PSP se basa en los siguientes principios de planeación y de calidad [HUMPHREY]

- Cada ingeniero es esencialmente diferente; para ser más precisos, los ingenieros deben planear su trabajo y basar sus planes en sus propios datos personales.
- Para mejorar constantemente su funcionamiento, los ingenieros deben utilizar personalmente procesos bien definidos y medidos.
- Para desarrollar productos de calidad, los ingenieros deben sentirse personalmente comprometidos con la calidad de sus productos.

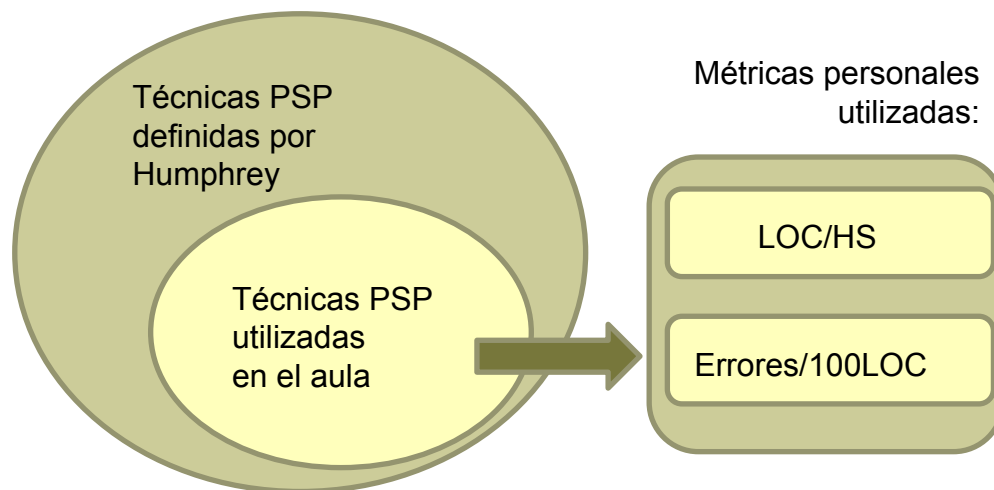
Cuesta menos encontrar y arreglar errores en la etapa inicial del proyecto que encontrarlos en las etapas subsecuentes. Es más eficiente prevenir defectos que encontrarlos y arreglarlos. La manera correcta de hacer las cosas es siempre la manera más rápida y más barata de hacer un trabajo.

# Camino de PSP



# Metodología aplicada en el aula

PSP es otra metodología de trabajo de la industria de software la cual adaptamos y simplificamos, con el objetivo que el alumno sea capaz de medir sus habilidades para producir software de calidad y estimar su tiempo de desarrollo.




Las técnicas PSP que se usan en el aula, son un subconjunto de las definidas por Humphrey en el libro "Introducción al Proceso Personal de Software".

Las métricas personales de software utilizadas en el curso son las LOC/hs (líneas de código por hora) y la tasa de errores cometidos por cada 100 LOC. La medición de la tasa de errores permite conocer la calidad del código elaborado. Conocer cuantos errores se cometen y comprender sus causas y consecuencias permite realizar mejoras significativas.

# Planilla para tomar métricas

PROYECTO: Algoritmo de Quicksort										
<b>Análisis</b>										
Tiempo Estimado	Hora Inicio	Hora Fin	Tiempo Real							
0:05	0:05	0:15	0:10							
<b>Preparación de la Prueba</b>										
Tiempo Estimado	Hora Inicio	Hora Fin	Tiempo Real							
0:15	0:20	0:40	0:20							
<b>Diseño</b>										
Tiempo Estimado	Hora Inicio	Hora Fin	Tiempo Real							
0:30	12:20	13:00	0:40							
<b>Desarrollo y correctivos</b>										
N° Incremento	Descripción de las tareas del Incremento	Estimación		Desarrollo			Correctivos		Líneas Reales	Tiempo Real
		Líneas Cód.	Tiempo	Hora Inicio	Hora Fin	Tiempo	Errores Lógicos	Tiempo Corrección		
5	Preparar el programa probador	80	1:00	6:00	6:30	0:30	1	0:10	100	0:40
6	Función de lectura desde archivo	15	0:20	13:00	13:40	0:40	0	0:00	20	0:40
7	Algoritmo de ordenamiento por fusión	50	1:00	15:00	15:20	0:20	3	0:20	45	0:40
8	Función de grabar resultados	15	0:20	17:20	18:00	0:40	0	0:00	45	0:40
<b>TOTALES</b>		<b>160</b>	<b>2:40</b>			<b>2:10</b>	<b>4</b>	<b>0:30</b>	<b>210</b>	<b>2:40</b>
<b>Ejecución de la Prueba</b>										
Tiempo Estimado	Hora Inicio	Hora Fin	Tiempo Real							
0:10	18:00	18:15	0:15							
<b>Resumen</b>										
Líneas de Código (LOC)	210									
LOC / Hora	78,75									
Errores Lógicos / 100 LOC	2									
Porcentaje Errores Lógicos	1,9%									
Tiempo de Análisis	0:10	4,1%								
Tiempo de Preparación de	0:20	8,2%								
Tiempo de Diseño	0:40	16,3%								
Tiempo de Ejecución de	0:15	6,1%								
Tiempo Resolución Errores	0:30	12,2%								
Tiempo Efectivo de	2:10	53,1%								
Tiempo Total	4:05									



- Tiempo de Análisis
- Tiempo de Preparación de Prueba
- Tiempo de Diseño
- Tiempo de Ejecución de Prueba
- Tiempo Resolución Errores Lógicos
- Tiempo Efectivo de Desarrollo

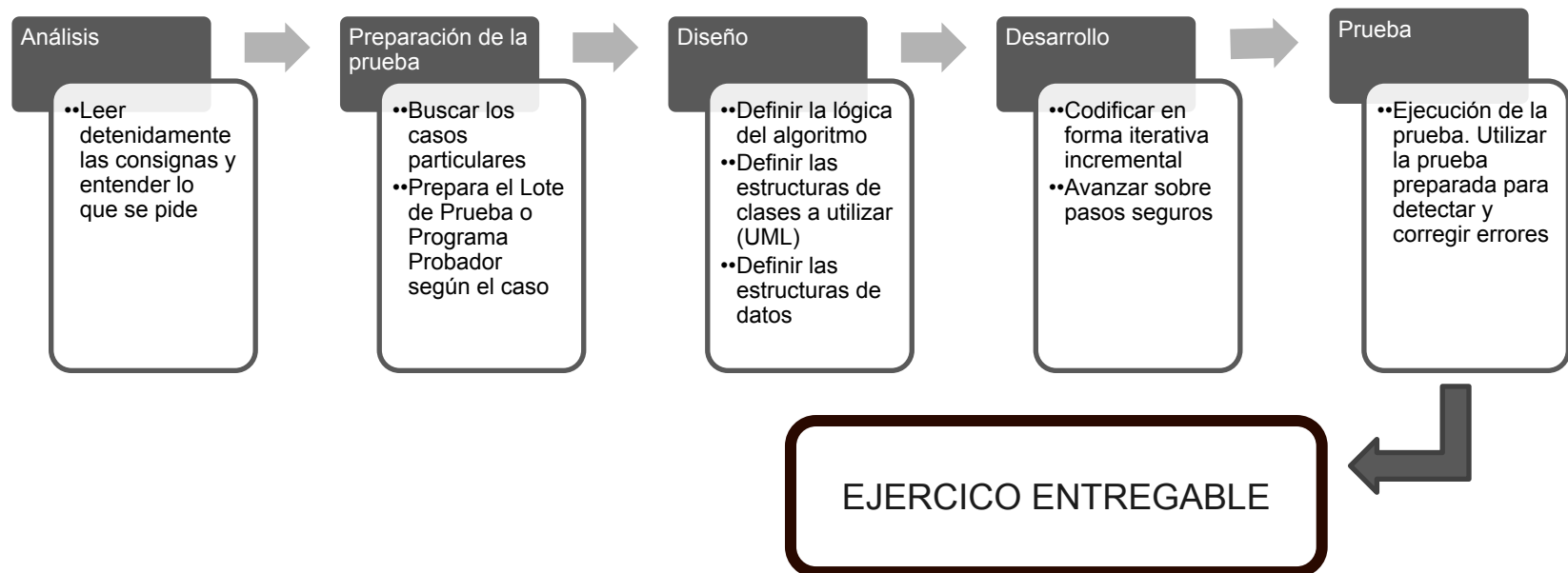
# El uso de PSP permitirá a los alumnos:



- Aprender a administrar el tiempo entre las distintas fases del desarrollo de software.
- Entender la relación entre el tamaño de los programas que escriben y el tiempo que les toma desarrollarlos.
- Aprender a tomar compromisos que puedan cumplir.
- Preparar un plan ordenado para realizar su trabajo
- Establecer una base para realizar un seguimiento de su trabajo.

## En concreto

Para la resolución de un ejercicio debemos seguir todos los pasos del desarrollo de software con el agregado de la prueba primero



Aplicar PSP, en cada etapa llenar la planilla de métricas. Estimar de acuerdo a los datos históricos, medir el proceso actual e ir mejorando el proceso.

## Bibliografía



Humphrey, Watts S. Introducción al PSP. Madrid 2001.