## Proceso Definido vs. Proceso Empírico

*Definido*

Un proceso definido asume que podemos repetir el mismo proceso una y otra vez, indefinidamente, y obtener los mismos resultados.

- Estos procesos están inspirados en las líneas de producción.
- La administración y el control provienen de la predictibilidad del proceso.

*Empírico*

Asume procesos complicados con variables cambiantes, si el proceso se repite, los resultados obtenidos pueden ser diferentes.

- Estos procesos se ajustan de mejor forma a procesos creativos y complejos.
- La administración y el control es por medio de inspecciones frecuentes y adaptaciones.

## Ciclos de vida

*Definición de ciclo de vida*

Son modelos genéricos (no descripciones definitivas) de los procesos de software; es decir, son abstracciones del proceso que se usan para explicar los distintos enfoques del desarrollo de software. En definitiva, un ciclo de vida de software es una representación de un proceso, el cual grafica una descripción del mismo desde una perspectiva particular.

**Características**

- También se los conoce como modelo de proceso.
- Especifican las fases del proceso y el orden en el que se llevan a cabo (requerimientos, especificación, diseño, etc)
- Es una guía para la administración del proyecto ya que indica el progreso a través de hitos.
- Los modelos de ciclos de vida se han vuelto necesarios debido a que los sistemas son más complejos por el aumento de funcionalidad y la mayor variedad de usuarios.
- Son independientes de los procedimientos de cada actividad del ciclo de vida.

*Clasificación de los ciclos de vida*

Los ciclos de vida que se presentan a continuación NO son mutuamente excluyentes y con frecuencia se usan en conjunto, sobre todo para sistemas complejos y de gran envergadura. Estos

- **Secuenciales**

Toma las actividades fundamentales del proceso; especificación, desarrollo, validación y evolución y los representa como fases separadas del proceso: especificación de requerimientos, diseño de software, implementación, pruebas, etc.

- ✓ Desarrollo dirigido por un plan
- ✓ Cada etapa genera documentación para realizar un monitoreo constante contra el plan.
- ✓ Muy útil cuando los requerimientos son claros y es poco probable un cambio drástico durante el desarrollo

- La documentación puede ser burocrática y excesiva
- En etapas finales puede ser que haya que hacer re trabajo por cambios en requerimientos o fallas de diseño

- **Iterativos**

Este enfoque vincula las actividades de especificación, desarrollo y validación. El sistema se desarrolla como una serie de versiones (incrementos) y cada una añade funcionalidades a la versión anterior.

- La especificación, desarrollo y validación están entrelazadas en lugar de separadas y aisladas.
- Rápida retroalimentación a través de las actividades.
- Muy útiles para sistemas de requerimientos cambiantes (Ej.: e-commerce, empresariales, etc)
- Más fácil y menos costoso implementar cambios.

 Cada iteración genera funcionalidad para el cliente.

 Los incrementos progresivos tienden a degradar la estructura del sistema.

**· Recursivos**

Se inicia con algo en forma completa, como una subrutina que se llama a si misma e inicia nuevamente. Se presenta un prototipo que va mejorando con cada vuelta.
 Se generan productos independientes de la implementación, que pueden ser reusables en sistemas de características similares.

 Puede ser más costoso en tiempo y dinero readaptarlos para reutilizarlos para los diferentes proyectos.

 La tecnología puede ser obsoleta.

 Pueden carecer de mantenimiento o documentación.

## *Modelos de ciclos de vida*

**Code and Fix**
Se desarrolla sin especificaciones o diseño y se lo modifica hasta que el cliente este satisfecho. Los cambios se realizan durante el mantenimiento, lo que es muy caro.

**Modelo en Cascada Puro**
Se sigue una secuencia de pasos ordenada y se realiza revisión al final de cada fase. Es conducida por la documentación con fases que no se superponen.

· Permite encontrar errores en etapas tempranas.

· Hay exceso de documentación y falta de resultados hasta el final.

· Se utiliza cuando la definición del producto es estable o los requerimientos de calidad dominan a los de costo y tiempo.

**Modelo en Cascada con Fases Solapadas**
Hay un fuerte grado de solapamiento. La documentación es menor pero es más difícil hacer el seguimiento de progreso. **Modelo de Entrega por Etapas**
Se ven signos tangibles de progreso. Es útil cuando el cliente necesita funcionalidad de inmediato y las necesidades se modifican. Debe haber una planificación rigurosa para que funcione.

**Modelo en Cascada con Retroalimentación**
Es una variación del modelo clásico en que es posible volver a una etapa anterior antes de llegar al final aunque es muy caro hacerlo.

**Modelo en Cascada con Subproyectos**
Es un modelo secuencial. Se avanza en cascada hasta el diseño arquitectónico, de ahí en más se avanza en partes dividiendo en Subproyectos.

**Modelo en espiral**
Busca minimizar los riesgos haciendo un análisis de riesgos y buscando alternativas al iniciar cada fase. Al final de cada fase se planifica la siguiente y se evalúa si se sigue adelante.
 Permite evaluar la factibilidad de un nuevo producto.

 En proyectos grandes la identificación de los riesgos puede costar más que el desarrollo.

 Es un modelo adecuado para proyectos con ciclos de vida largos, que puede utilizar equipos diferentes en cada ciclo y muestra el progreso al fin de cada ciclo.

**Modelo Evolutivo**
Se fija el tiempo o el alcance mientras el otro se va modificando. Es útil cuando los requerimientos no son claros y se realizan entregas tempranas al cliente.
 El problema es que es un modelo recursivo que repite todas las tareas y depende de que los diseñadores desarrollen un sistema fácil de modificar.

**Diseño para Cronograma**
No se asegura alcanzar la versión final, aunque si una entrega del producto para la fecha definida con los aspectos más importantes. Es útil cuando no hay seguridad sobre las capacidades de programación.

**Diseño para Herramientas**
Consiste en incluir solo la funcionalidad soportada por las herramientas de software existentes. Se usa en proyectos muy sensibles al tiempo. Puede combinarse con otros ciclos de vida, pero no se podrán implementar todos los requerimientos.
**Prototipación Evolutiva**
Se da cuando hay cambios rápidos de requerimientos y los clientes no se comprometen con los requisitos. Requiere menos documentación pero no se pueden programar los release. Se desarrolla el concepto del sistema a medida que se avanza. *Elección de un ciclo de vida*
La elección depende de muchos factores tales como:
- Riesgos técnicos
- Riesgos de Administración
- Volatilidad de los requerimientos
- Ciclo de tiempo requerido
- Aspectos del cliente
- Tamaño del Equipo

*Algunas consideraciones*
 Una herramienta para la planificación y el monitoreo de proyectos.
- Un modelo de progreso del proyecto.
- Independiente de los métodos y procedimientos de cada actividad del ciclo de vida.
- Muy abstracto.