



## **UNIDAD 2**

- **¿Cómo enfrentar el cambio?**
  - **Prototipos.**
  - **Entrega incremental.**
- **Organización de equipos y metodología ágil:**
  - **Manifiesto ágil.**
  - **XP (Programación Extrema).**
  - **SCRUM. Roles y responsabilidades. Ciclo de trabajo.**



## ¿Cómo enfrentar el cambio?

El cambio es inevitable en todos los grandes proyectos de software.

Los requerimientos del sistema varían conforme:

- La empresa procura que el sistema responda a **presiones externas**.
- Se modifican las **prioridades administrativas**.
- A medida que se ponen a disposición **nuevas tecnologías**, surgen nuevas posibilidades de diseño e implementación.

Cualquiera que sea el modelo del proceso de software utilizado, es esencial que ajuste los cambios al software a desarrollar.



## ¿Cómo enfrentar el cambio?

El cambio significa que el trabajo ya terminado debe volver a realizarse.

- Existen dos enfoques relacionados que se usan para reducir los costos del rehacer:
  1. Evitar el cambio, el proceso de software incluye actividades que anticipan cambios posibles antes de requerirse.
  2. Tolerancia al cambio, esto comprende algunas formas de desarrollo incremental. Los cambios propuestos pueden implementarse en incrementos que aún no se desarrollan.



## ¿Cómo enfrentar el cambio?

- Formas de enfrentar el cambio y los requerimientos cambiantes del sistema.
  1. Prototipo de sistema, donde rápidamente se desarrolla una versión del sistema o una parte del mismo, para comprobar los requerimientos del cliente y la factibilidad de algunas decisiones de diseño.
  2. Entrega incremental, donde los incrementos del sistema se entregan al cliente para su comentario y experimentación.



## Prototipos

Un prototipo es una versión inicial de un sistema de software que se usa para demostrar conceptos, tratar opciones de diseño y encontrar más sobre el problema y sus posibles soluciones..

- Usos del prototipo:
  - En el proceso de ingeniería de requerimientos, ayuda con la selección y validación de requerimientos del sistema.
  - En el proceso de diseño de sistemas, sirve para buscar soluciones específicas de software y apoyar el diseño de interfaces del usuario.



## Prototipos

La siguiente etapa del proceso consiste en decidir qué poner y qué dejar fuera del sistema de prototipo.

Para reducir tiempos de entrega y costos puede dejarse fuera del prototipo por ejemplo:

- Manejo y gestión de errores.
- Hacer mas flexible el tiempo de respuesta y la utilización de memoria.

La etapa final del proceso es la evaluación del prototipo:

- Se debe capacitar al usuario.
- Tener un plan de evaluación (según objetivos).
- Dar tiempo para que los usuarios utilicen el sistema de forma normal.



## Prototipos

Un problema general con la creación de prototipos es que quizás el prototipo no se utilice necesariamente en la misma forma que el sistema final.

Motivos:

- El revisor del prototipo tal vez no sea un usuario típico del sistema.
- Insuficiente el tiempo de capacitación.
- Si el prototipo es lento, los evaluadores podrían ajustar su forma de trabajar.



## Prototipos

Los prototipos no tienen que ser ejecutables para ser útiles.

Los modelos en papel de la interfaz de usuario del sistema pueden ser efectivos para ayudar a los usuarios a refinar un diseño de interfaz y trabajar a través de escenarios de uso. Su desarrollo es muy económico y suelen construirse en pocos días.





## Entrega incremental.

Es un enfoque al desarrollo de software donde algunos de los incrementos diseñados se entregan al cliente y se implementan para usarse en un entorno operacional.

Los clientes identifican los servicios que proporciona el sistema. Identifican cuáles servicios son más importantes y cuáles son menos significativos para ellos. Entonces, se define un número de incrementos de entrega, y cada incremento proporciona un subconjunto de la funcionalidad del sistema.

La asignación de servicios por incrementos depende de la prioridad del servicio, donde los servicios de más alta prioridad se implementan y entregan primero.



## Entrega incremental.

Una vez identificados los incrementos del sistema, se definen con detalle los requerimientos de los servicios que se van a entregar en el primer incremento, y se desarrolla ese incremento.

**Durante el desarrollo, puede haber un mayor análisis de requerimientos para incrementos posteriores, aun cuando se rechacen cambios de requerimientos para el incremento actual.**

Una vez completado y entregado el incremento, los clientes lo ponen en servicio. Esto significa que toman la entrega anticipada de la funcionalidad parcial del sistema. Pueden experimentar con el sistema que les ayuda a clarificar sus requerimientos, para posteriores incrementos del sistema. A medida que se completan nuevos incrementos, se integran con los incrementos existentes, de modo que con cada incremento entregado mejore la funcionalidad del sistema.



## Entrega incremental.

### Ventajas:

1. Los clientes pueden usar los primeros incrementos como prototipos y adquirir experiencia que informe sobre sus requerimientos, para posteriores incrementos del sistema. A diferencia de los prototipos, éstos son parte del sistema real.
2. Los clientes no deben esperar hasta la entrega completa del sistema, antes de ganar valor del mismo. El primer incremento cubre sus requerimientos más críticos, de modo que es posible usar inmediatamente el software.
3. El proceso mantiene los beneficios del desarrollo incremental en cuanto a que debe ser relativamente sencillo incorporar cambios al sistema.
4. Puesto que primero se entregan los servicios de mayor prioridad y luego se integran los incrementos, los servicios de sistema más importantes reciben mayores pruebas.



## Entrega incremental.

Sin embargo, existen problemas con la entrega incremental:

- La mayoría de los sistemas requieren de una serie de recursos que se utilizan para diferentes partes del sistema. Dado que los requerimientos no están definidos con detalle sino hasta que se implementa un incremento, resulta difícil identificar recursos comunes que necesiten todos los incrementos.
- Asimismo, el desarrollo iterativo resulta complicado cuando se diseña un sistema de reemplazo. Los usuarios requieren de toda la funcionalidad del sistema antiguo.
- En los procesos iterativos es que la especificación se desarrolla en conjunto con el software, esto se puede contradecir con el modelo de adquisiciones de muchas organizaciones, donde la especificación completa del sistema es parte del contrato. En el enfoque incremental, no hay especificación completa del sistema, sino hasta que se define el incremento final. Esto requiere una nueva forma de contrato que los grandes clientes, como las agencias gubernamentales, encontrarían difícil de adoptar.



## Organización de equipos y metodología ágil

Aun cuando existen muchos enfoques para el desarrollo de software rápido, comparten algunas características fundamentales:

- a.- Los procesos de especificación, diseño e implementación están entrelazados.
- b.- No existe una especificación detallada del sistema.
- c.- La documentación del diseño se minimiza o es generada automáticamente por el IDE.
- d.- El documento de requerimientos sólo define las características mas importantes.
- e.- El sistema se desarrolla en diferentes versiones.
- f.- Los usuarios finales intervienen en la especificación y evaluación de cada versión.
- g.- Las interfaces de usuario del sistema se desarrollan usando con frecuencia un sistema de elaboración interactivo, que permita que el diseño de la interfaz se cree rápidamente en cuanto se dibujan y colocan iconos en la interfaz.



## Organización de equipos y metodología ágil

*“Los **métodos ágiles se apoyan en el enfoque incremental** para la especificación, el desarrollo y la entrega del software”*

*“Son más adecuados para el diseño de aplicaciones en que los requerimientos del sistema cambian rápidamente durante el proceso de desarrollo”*

*“Se dirigen a simplificar el proceso burocrático al evitar trabajo con valor dudoso a largo plazo, y a eliminar documentación que quizá nunca se emplee.”*

**La filosofía detrás de los métodos ágiles se refleja en el manifiesto ágil, que acordaron muchos de los desarrolladores líderes de estos métodos.**



## Organización de equipos y metodología ágil

Probablemente el método ágil más conocido sea la **programación extrema** (Beck, 1999; Beck, 2000). Otros enfoques ágiles incluyen los de **Scrum** (Cohn, 2009; Schwaber, 2004; Schwaber y Beedle, 2001), de **Crystal** (Cockburn, 2001; Cockburn, 2004), de **desarrollo de software adaptativo** (Highsmith, 2000), de DSDM (Stapleton, 1997; Stapleton, 2003) y el **desarrollo dirigido por características** (Palmer y Felsing, 2002).

El éxito de dichos métodos condujo a cierta integración con **métodos más tradicionales de desarrollo, basados en el modelado de sistemas**, lo cual resulta en la noción de **modelado ágil** (Ambler y Jeffries, 2002) y **ejemplificaciones ágiles del Proceso Racional Unificado** (Larman, 2002).



## Organización de equipos y metodología ágil

Aunque todos esos métodos ágiles se basan en la noción del desarrollo y la entrega incrementales, [proponen diferentes procesos para lograrlo.](#)

Sin embargo, comparten una serie de principios, según el manifiesto ágil y, por ende, tienen mucho en común.

Diferentes métodos ágiles ejemplifican esos principios en diversas formas.



## Organización de equipos y metodología ágil

### Manifiesto ágil:

*El manifiesto Ágil surge el 17 de febrero del 2001 cuando se reunieron diecisiete críticos del desarrollo de software. En este evento, acuñaron el término “metodología Ágil” para definir los métodos que estaban surgiendo como alternativa a las metodologías formales.*



Image Source: <http://udayanbanerjee.wordpress.com/category/agile>

El manifiesto Ágil está conformado por 12 principios asociados a 4 aspectos o pilares.

<https://agilemanifesto.org/iso/es/manifesto.html>



## Organización de equipos y metodología ágil

### Manifiesto ágil:

*Estamos descubriendo mejores formas para desarrollar software, al hacerlo y al ayudar a otros a hacerlo. Gracias a este trabajo llegamos a valorar:*

*A los individuos y las interacciones sobre los procesos y las herramientas*

*Al software operativo sobre la documentación exhaustiva*

*La colaboración con el cliente sobre la negociación del contrato*

*La respuesta al cambio sobre el seguimiento de un plan*



*Esto es, aunque exista valor en los objetos a la derecha, valoraremos más los de la izquierda.*

## Organización de equipos y metodología ágil

### Manifiesto ágil:

#### Valores

-  Individuos e Interacciones
-  Software Funcional
-  Colaboración con el cliente
-  Respuesta al cambio

-  Procesos y Herramientas
-  Documentación Extensiva
-  Negociación Contractual
-  Seguir un plan

*“si bien hay valor en los ítems de la derecha, **valoramos más los ítems de la izquierda...**”*



## Organización de equipos y metodología ágil

### Manifiesto ágil:

#### *Principios*

1

Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.

2

Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.

3

Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.

4

Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.



## Organización de equipos y metodología ágil

### Manifiesto ágil:

#### *Principios*

5

Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan además de confiarles la ejecución del trabajo.

6

El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.

7

El software funcionando es la medida principal de progreso.

8

Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.



## Organización de equipos y metodología ágil

### Manifiesto ágil:

#### *Principios*

9

La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.

10

La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.

11

Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

12

A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para, a continuación, ajustar y perfeccionar su comportamiento en consecuencia.



## Organización de equipos y metodología ágil

### Dificultades de los métodos ágiles:

- a.- Al enfocarse en equipos reducidos y fuertemente integrados, hay problemas en escalarlos hacia grandes sistemas.
- b.- Dificultad para involucrar al cliente en el proceso de desarrollo el tiempo suficiente.
- c.- Limitación de miembros del equipo que no cuenten con una personalidad para la participación intensa.
- d.- Priorizar los cambios es difícil en equipos con muchos participantes.
- e.- Mantener la simplicidad requiere trabajo adicional.
- f.- Dificultad para adaptar la cultura organizacional a un modelo de trabajo donde los procesos sean informales y sean definidos por equipos de desarrollo.
- g.- Cuando se acude a externos para desarrollar los sistemas se dificulta la elaboración del contrato dado que debe ser por tiempo en lugar de especificaciones.



## Organización de equipos y metodología ágil

Se presentan entonces dos preguntas que deberían considerarse junto con los métodos y el mantenimiento ágiles:

1. ¿Los sistemas que se desarrollan usando un enfoque ágil se mantienen, a pesar del énfasis en el proceso de desarrollo de minimizar la documentación formal?
2. ¿Los métodos ágiles pueden usarse con efectividad para evolucionar un sistema como respuesta a requerimientos de cambio por parte del cliente?





## Organización de equipos y metodología ágil

La documentación formal describe el sistema y, por lo tanto, facilita la comprensión a quienes cambian el sistema. Sin embargo, en la práctica, con frecuencia la documentación formal no se conserva actualizada. Los apasionados de los métodos ágiles argumentan que escribir esta documentación es una pérdida de tiempo y que la clave para implementar software mantenible es producir un código legible de alta calidad.

*“La falta de documentación no debe representar un problema para mantener los sistemas desarrollados con el uso de un enfoque ágil”*



## Organización de equipos y metodología ágil

Es factible que las prácticas ágiles, usadas en el proceso de mantenimiento en sí, resulten efectivas, ya sea que se utilice o no se utilice un enfoque ágil para el desarrollo del sistema.

La entrega incremental, el diseño para el cambio y el mantenimiento de la simplicidad tienen sentido cuando se modifica el software. De hecho, se pensaría tanto en un proceso de desarrollo ágil como en un proceso de evolución del software.



## Organización de equipos y metodología ágil

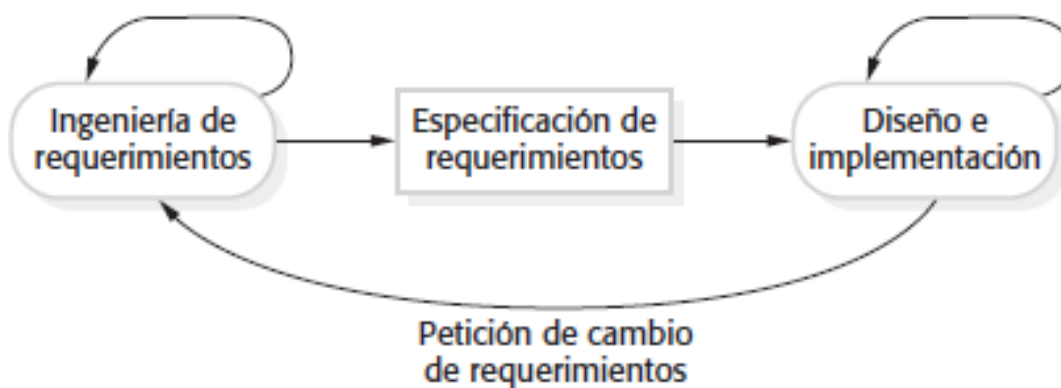
### Diferencia entre desarrollo dirigido por un plan y desarrollo ágil.

En un enfoque basado en un plan, la iteración ocurre dentro de las actividades con documentos formales usados para comunicarse entre etapas del proceso.

Por ejemplo, los requerimientos evolucionarán y, a final de cuentas, se producirá una especificación de aquéllos. Esto entonces es una entrada al proceso de diseño y la implementación.

En un enfoque ágil, la iteración ocurre a través de las actividades. Por lo tanto, los requerimientos y el diseño se desarrollan en conjunto, no por separado.

## Desarrollo basado en un plan



## Desarrollo ágil





## Organización de equipos y metodología ágil

Un proceso de software dirigido por un plan soporta el desarrollo y la entrega incrementales. Es factible asignar requerimientos y planear tanto la fase de diseño y desarrollo como una serie de incrementos.

Un proceso ágil no está inevitablemente enfocado al código y puede producir cierta documentación de diseño. El equipo de desarrollo ágil puede incluir un “pico” de documentación donde, en vez de producir una nueva versión de un sistema, el equipo generará documentación del sistema.



## **Organización de equipos y metodología ágil**

**“La mayoría de los proyectos de software incluyen prácticas de los enfoques ágil y basado en un plan”**



## Organización de equipos y metodología ágil

Para decidir sobre el equilibrio entre un enfoque basado en un plan y uno ágil, se deben responder algunas preguntas técnicas, humanas y organizacionales:

1. ¿Es importante tener una especificación y un diseño muy detallados antes de dirigirse a la implementación?
2. ¿Es práctica una estrategia de entrega incremental, donde se dé el software a los clientes y se obtenga así una rápida retroalimentación de ellos?
3. ¿Qué tan grande es el sistema que se desarrollará?
4. ¿Qué tipo de sistema se desarrollará?
5. ¿Cuál es el tiempo de vida que se espera del sistema?



## Organización de equipos y metodología ágil

5. ¿Qué tecnologías se hallan disponibles para apoyar el desarrollo del sistema?
6. ¿Cómo está organizado el equipo de desarrollo?
8. ¿Existen problemas culturales que afecten el desarrollo del sistema?
9. ¿Qué tan buenos son los diseñadores y programadores en el equipo de desarrollo?
10. ¿El sistema está sujeto a regulación externa?