

# Programa académico CAMPUS



**Trainer**  
**Ing. Carlos H. Rueda C.**




# SCRUM

## Metodologías Ágiles en el Desarrollo de Software



# 1. Qué son las metodologías tradicionales

 Como se había mencionado anteriormente, las metodologías se pueden aplicar a nivel general de acuerdo a la línea profesional en la que se piense utilizar. Dicho esto, en la línea del desarrollo de software encontraremos diversos tipos de metodologías, las cuales estarán alineadas a su manera de ejercer. Entre ellas encontraremos las metodologías tradicionales y las metodologías ágiles.

# 1. Qué son las metodologías tradicionales

Podemos entender como ***metodologías tradicionales*** a todo tipo de declaración de etapas que tengan un alcance y requisitos definidos de manera rígida desde el inicio del proyecto.

Las metodologías tradicionales buscarán establecer un paso a paso definido, prevaleciendo la **precisión** de la ejecución del proyecto.

<p style="text-align:right;">  </p>

# 1. Qué son las metodologías tradicionales

## Enfoque lineal secuencial:

- Las etapas del proceso se suceden una tras otra.
- No se puede empezar la siguiente etapa sin terminar la anterior.
- Útil para proyectos con requisitos bien definidos y estables.
- Un problema para proyectos con requisitos cambiantes o con un entorno de desarrollo dinámico.

<p style="text-align:right;">  </p>

# 1. Qué son las metodologías tradicionales

## Planificación detallada:

- Requieren una planificación detallada del proyecto, en la que se definen los requisitos, el diseño, la construcción y las pruebas, ayudando a garantizar que el proyecto se desarrolle de manera eficiente y eficaz.
- Puede ser un proceso laborioso y costoso.

<p style="text-align:right;">  </p>

# 1. Qué son las metodologías tradicionales

## Documentación exhaustiva:

- Generan una gran cantidad de documentación, que sirve para registrar y documentar todos los aspectos del desarrollo del software, ayudando a garantizar que el proyecto se pueda comprender y mantener fácilmente.
- Puede ser una tarea tediosa y puede generar un exceso de documentación si no se sabe planificar su alcance y manera de redacción.

<p style="text-align:right;">  </p>

# 1. Qué son las metodologías tradicionales

- **Rigidez:** Las metodologías tradicionales son muy rígidas, lo que significa que no son adaptables a cambios en los requisitos o en las condiciones del proyecto. Esto puede ser un problema si los requisitos cambian o si surgen nuevos desafíos durante el desarrollo.

<p style="text-align:right;">  </p>



## 2. Tipos de Metodologías Tradicionales

Conociendo las diversas características de las metodologías tradicionales, revisaremos a continuación algunas metodologías utilizadas actualmente en el mercado del desarrollo:

<p style="text-align:right;">  </p>

## 2.1. Metodología Tradicional - Cascada

El modelo en cascada, también conocido como modelo ***Waterfall***, es una metodología de gestión de proyectos que se caracteriza por su división en fases bien definidas, donde cada etapa inicia únicamente después de que la fase anterior se ha completado por entero.

<p style="text-align:right;">  </p>

## 2.1. Metodología Tradicional - Cascada

- se originó en los sectores de fabricación y construcción.
- Es imposible comenzar a construir las paredes de una casa si los cimientos aún no están completamente terminados. E
- Lógica de progresión secuencial para garantizar que cada parte del proyecto esté sólidamente establecida antes de pasar a la siguiente fase.

<p style="text-align:right;">  </p>

## 2.1. Metodología Tradicional - Cascada

`<p style="text-align:center;" >  </p>`

## 2.1. Metodología Tradicional - Casca

- Al momento de presentar dicha metodología se presenta generalmente en forma de un diagrama de flujo o un gráfico de Gantt, ya que su nombre de "**waterfall**" o cascada proviene en que cada tarea fluye de manera secuencial hacia la siguiente, de forma similar a cómo el agua descende en una cascada.
- En un diagrama de Gantt, esta progresión secuencial se hace visible a medida que una fase previa cae en cascada sobre la fase siguiente, creando una representación gráfica de la secuencia de tareas.

<p style="text-align:right;">  </p>

## 2.1.1. Metodología en Cascada - Fase Requisitos

En esta fase inicial de planificación en la metodología en cascada se reúnen exhaustivamente todos los datos necesarios para asegurar el éxito del proyecto. Como en esta metodología cada etapa dependen de las anteriores, es imperativo prever en detalle todo antes de iniciar. Dicho esto, será de vital importancia esta etapa de planificación, por lo cual será la que consumirá la mayor parte del tiempo del proyecto.

<p style="text-align:right;">  </p>

## 2.1.1. Metodología en Cascada - Fase Requisitos

```
<p style="text-align:center;">  </p>
```

## 2.1.1. Metodología en Cascada - Fase Requisitos

- Se debe elaborar un plan detallado del proyecto que abarque todos los aspectos, desde los recursos requeridos hasta los miembros específicos del equipo involucrados.
- Se le denomina “***levantamiento de requerimientos del proyecto***”.
- Al finalizar esta fase, se debe contar con una visión clara y completa del proyecto, que incluye cada etapa del proceso, quién estará a cargo de cada fase, las dependencias clave, los recursos necesarios y un cronograma que detalle la duración de cada etapa.

<p style="text-align:right;">  </p>



## 2.1.1. Metodología en Cascada - Fase Requisitos

### Cómo se levantan los requerimientos del proyecto:

1. **Entrevistas:** Las entrevistas representan una técnica fundamental y ampliamente utilizada para adquirir información valiosa de los usuarios y las partes interesadas. Estas conversaciones cara a cara son esenciales para comprender a fondo las necesidades y los requisitos de los usuarios, pues a través de las entrevistas, se establece un diálogo con los usuarios para explorar sus expectativas y demandas.

<p style="text-align:right;">  </p>

## 2.1.1. Metodología en Cascada - Fase Requisitos

**Cómo se levantan los requerimientos del proyecto:**

**2. Reuniones JAD (Joint Application Development):** Las reuniones JAD son sesiones de trabajo colaborativas en las que participan usuarios, analistas de sistemas y otras partes interesadas, donde su propósito es recopilar y refinar los requisitos del software. En estas sesiones, se fomenta la interacción directa entre todos los actores involucrados en el proyecto para garantizar una comprensión completa y consensuada de lo que se requiere.

<p style="text-align:right;">   

## 2.1.1. Metodología en Cascada - Fase Requisitos

### Cómo se levantan los requerimientos del proyecto:

5. **Escenarios:** Los escenarios son narrativas que describen cómo los usuarios interactuarán con el software en situaciones específicas. Estas historias ayudan a visualizar y comprender cómo el software debe funcionar en la práctica. Los escenarios son útiles para definir tanto los requisitos funcionales como los no funcionales del software.

<p style="text-align:right;">   
</p>

## 2.1.1. Metodología en Cascada - Fase Requisitos

### Cómo se levantan los requerimientos del proyecto:

6. **Casos de Uso:** Los casos de uso son descripciones detalladas de las interacciones entre los usuarios y el software. Estas descripciones se utilizan para documentar los requisitos funcionales del software, centrándose en cómo el sistema responderá a las acciones de los usuarios. Los casos de uso proporcionan una estructura clara para comprender y especificar las funcionalidades necesarias del software.

<p style="text-align:right;">   <img  
src="https://th.bing.com/th/id/R.2ca500743cca22834d50a2277253  
9742?  
rik=e8s1mnkPrv6FkQ&riu=http%3a%2f%2f2.bp.blogspot.com%2f\_M  
s0XIJo9SRg%2fTVFmI0afbvI%2fAAAAAAAAAABQ%2fid4wmpUnn14  
%2fs1600%2fANALIS%25257E1.JPG&ehk=Ku%2f5JXpV1G2hXQNE



## 2.1.3. Metodología en Cascada - Programación

- En esta fase, **todo comienza a tomar forma**, pues aquí se basará en los documentos de requerimientos de la Fase 1 y en el proceso de análisis de la Fase 2.
- Construir el software conforme a las especificaciones definidas en las etapas previas.

<p style="text-align:right;">  </p>

## 2.1.4. Metodología en Cascada - Pruebas

- En este punto del método en cascada, el equipo de desarrollo hace entrega del proyecto al equipo de Calidad para llevar a cabo las pruebas necesarias.
- Buscarán cualquier error que deba ser corregido antes de la implementación del proyecto,
- Los resultados de las pruebas se documentan minuciosamente.

<p style="text-align:right;">   </p>

## 2.2. Metodología Tradicional - Prototipado

```
<p style="text-align:center;">   
</p>
```

## 2.2. Metodología Tradicional - Prototipado

```
<p style="text-align:center;">   
</p>
```

## 2.2. Metodología Tradicional - Prototipado

- En lugar de finalizar los requisitos antes del diseño o la codificación, se construye un prototipo desechable para comprender los requisitos.
- Este prototipo se crea en base a los requisitos conocidos en ese momento.
- Al utilizarlo, el cliente obtiene una visión real del sistema.
- Las interacciones con el prototipo les ayudan a comprender las necesidades del sistema deseado.
- Adecuado para sistemas complejos y extensos que carecen de un proceso manual o sistema existente para la determinación de requisitos.

<p style="text-align:right;">   </p>

## 2.2. Metodología Tradicional - Etapas del Prototipado

- **Fase de Diseño:** En esta fase se busca darle prototipo al software, el cual se podrá manifestarse como un modelo físico, modelo de software, y en algunos casos la combinación de ambos. Aquí se busca priorizar que el prototipo refleje de manera precisa y coherente los requerimientos que se definieron previamente para el software.

<p style="text-align:right;">  </p>



## 2.2. Metodología Tradicional - Etapas del Prototipado

- **Fase de Construcción del Prototipo:** En unión de los ítems anteriormente mencionados, aquí se llevará la creación del primer prototipo del software en mención, el cual se podrá desarrollar con diversas técnicas previamente establecidas por el equipo de desarrollo.

```
<p style="text-align:right;">  </p>
```

## 2.2. Metodología Tradicional - Etapas del Prototipado

- **Fase de Evaluación del Prototipo:** Aquí será una etapa importante, pues en esta se buscará revisar la opinión y retroalimentación dada por los usuarios finales antes de hacer la entrega del producto final. En esta etapa será de gran importancia hacer la documentación de dicho proceso, pues de acá se procederá a hacer las debidas mejoras.

<p style="text-align:right;">  </p>

## 2.2. Metodología Tradicional - Etapas del Prototipado

- **Fase de Refinamiento del Prototipo:** Teniendo el feedback por parte del usuario, se procede a ajustar y refinar el prototipo en mención. Dicho proceso puede repetirse varias veces de acuerdo a las necesidades encontradas hasta cumplir con los requerimientos anteriormente planteados.

<p style="text-align:right;">  </p>

## 2.2. Metodología Tradicional - Etapas del Prototipado

- **Fase de Ingeniería del Producto:** Finalmente tendremos en esta etapa nuestro primer MVP (Mínimo Producto Viable), en el cual deberá haber pasado por diferentes pruebas y técnicas con el fin de garantizar una buena calidad para producción.

<p style="text-align:right;">  </p>

## 2.3. Metodología Tradicional - Espiral

```
<p style="text-align:right;">  </p>
```

## 2.3. Metodología Tradicional - Espiral

- Utilizado para la gestión de riesgos
- Fusiona el modelo de desarrollo iterativo con elementos del modelo en cascada.
- Este enfoque es apreciado por los ingenieros de software y se prefiere en proyectos grandes, costosos y complejos.
- Se asemeja a una bobina con numerosos bucles, cuya cantidad se adapta a cada proyecto y generalmente es determinada por el director del proyecto.
- Cada bucle de la espiral representa una fase en el proceso de desarrollo de software, el cual tendrá una forma cíclica en su ejecución.

<p style="text-align:right;">  </p>

## 2.3. Metodología Tradicional - Espiral

1. **Determinar objetivos y encontrar soluciones alternas:** En esta etapa se lleva a cabo la recopilación y análisis de requisitos. Basándose en estos requisitos, se definen los objetivos y se proponen diversas soluciones alternativas.

<p style="text-align:right;">  </p>

## 2.3. Metodología Tradicional - Espiral

2. **Análisis de riesgos y resolución:** En este cuadrante, se analizan todas las soluciones propuestas y se identifican posibles riesgos en el proyecto. Luego, se realiza un análisis detallado de los riesgos identificados y se toman medidas para resolverlos.

<p style="text-align:right;">  </p>



## 2.3. Metodología Tradicional - Espiral

3. **Desarrollo y pruebas:** Esta fase implica la implementación real de las diferentes características del proyecto. Posteriormente, todas estas características implementadas se someten a pruebas exhaustivas para verificar su correcto funcionamiento.

<p style="text-align:right;">  </p>

## 2.3. Metodología Tradicional - Espiral

4. **Revisión y planificación de la próxima fase:** En esta fase, el software es evaluado por el cliente. También se lleva a cabo la identificación y seguimiento de riesgos, como posibles excesos de costos o retrasos en el cronograma. Después de evaluar el estado actual del proyecto, se inicia la planificación de la siguiente fase.

<p style="text-align:right;">  </p>

### 3. Qué son las metodologías ágiles

<p style="text-align:center;">  </p>

### 3. Qué son las metodologías ágiles

- Toda creación de fases que se fundamentan en el enfoque ***incremental***.
- Cada ciclo de desarrollo, se introducen nuevas funcionalidades en la aplicación final.
- Se distinguen por la ***rapidez y la brevedad*** de los ciclos, que resultan en la incorporación de pequeñas mejoras en lugar de realizar cambios significativos en una sola instancia.
- Equipos de trabajo que sean ***autónomos e independientes***, los cuales se reúnen de manera regular para compartir los avances logrados.
- El cliente tiene la posibilidad de participar y contribuir en el proceso de desarrollo a medida que avanza.

# 3. Qué son las metodologías ágiles

## Características

### Iteración:

- En el enfoque se caracteriza por su naturaleza cíclica y repetitiva,
- Cada ciclo, o iteración, culmina con la entrega de una versión funcional del software al cliente.

<p style="text-align:right;">  </p>

# 3. Qué son las metodologías ágiles

## Características

### Colaboración:

- Equipos multidisciplinarios trabajan juntos, aprovechando sus conocimientos y habilidades individuales para lograr un producto final de alta calidad.

<p style="text-align:right;">  </p>

# 3. Qué son las metodologías ágiles

## Características

### Comunicación:

<p> <li> La comunicación es un pilar fundamental. <li> Comunicación constante y abierta. <li> Reuniones regulares donde se coordinan las tareas, se resuelven problemas y se comparten avances. <li> La comunicación fluida y efectiva  </p>

# 3. Qué son las metodologías ágiles

## Características

### Adaptación a los cambios:

- Capacidad para adaptarse a los cambios en los requisitos del proyecto o en las condiciones del entorno.

<p style="text-align:right;">  </p>



# 4. Tipos de Metodologías Ágiles

```
<p style="text-align:center;">  </p>
```

## 4.1. Metodología Ágil - Kanban

- Busca tener un enfoque integral para la gestión de servicios,
- Firme compromiso en la mejora continua desde la perspectiva de los clientes.
- Visualiza el trabajo en curso
- Se destaca por su adaptabilidad, ya que puede ser ajustado para satisfacer las necesidades específicas de cada empresa, cumplir con las expectativas de los clientes y afrontar los desafíos del entorno empresarial.

## 4.1. Metodología Ágil - Kanban

```
<p style="text-align:center;">  </p>
```

Como podemos ver el **tablero** se divide por ***columnas y tarjetas***, las cuales pueden ser ajustadas de acuerdo a la necesidad que haya por parte del cliente.

## 4.1.1. Metodología Kanban - Construcción de columnas y tarjetas

Instructivo gráfico para la creación del tablero Kanban.

```
<p style="text-align:center;">  </p>
```

## 4.1.1. Metodología Kanban - Construcción de columnas y tarjetas

Mínimo tres columnas claramente definidas para gestionar el flujo de trabajo de manera efectiva, donde dichas columnas hacen lo siguiente:

### 1. **Por Hacer (To Do):**

- Tareas que aún no han sido iniciadas. Generalmente y están pendientes de ser abordadas.

<p style="text-align:right;">  </p>

## 4.1.1. Metodología Kanban - Construcción de columnas y tarjetas

### 2. En Progreso (Doing o Develop):

- Tareas que están actualmente en curso. Por lo general,
- Tareas en fase activa de desarrollo o pruebas.
- El equipo está trabajando activamente en ellas para avanzar hacia su finalización.

<p style="text-align:right;">  </p>

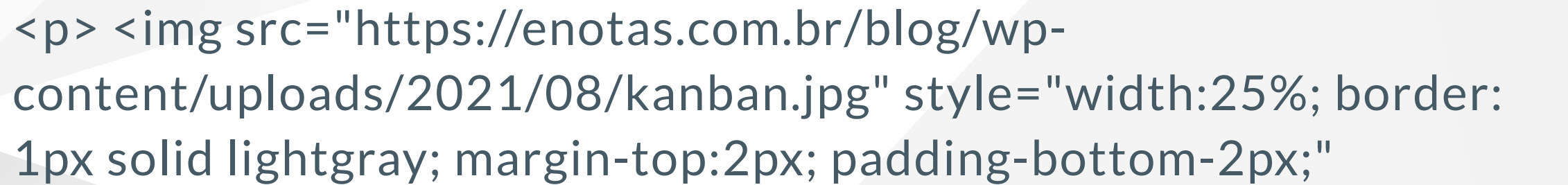
## 4.1.1. Metodología Kanban - Construcción de columnas y tarjetas

### 3. Completadas (Done):

- Tareas que han sido finalizadas con éxito.
- Tareas que han alcanzado una fase de implementación o mantenimiento.
- Representan los logros y el progreso alcanzado en el proyecto.

```
<p style="text-align:right;">  </p>
```

## 4.1.1. Metodología Kanban - Construcción de columnas y tarjetas

 Cada tarea se representa mediante una tarjeta. Suelen contener información detallada sobre la tarea: nombre, descripción, fechas y responsables. Se mueven a través de las columnas a medida que las tareas avanzan. Permiten al equipo visualizar el progreso del trabajo. Identifican posibles cuellos de botella o problemas en el flujo de trabajo.



## 4.2. Metodología Ágil - Lean

```
<p style="text-align:center;">  </p>
```

## 4.2. Metodología Ágil - Lean

- Es un enfoque de gestión de procesos que se centra en la ***eliminación de actividades*** que no tienen aporte al desarrollo del proyecto, con el fin de buscar una ***mejora constante en la eficiencia***.
- Sus raíces se encuentran en la industria manufacturera japonesa
- Este enfoque tiene como objetivo aumentar la productividad y la calidad al optimizar procesos y recursos de manera constante.

<p style="text-align:right;">  </p>

## 4.2. Metodología Ágil - Lean - 5 Principios

### 1. Definir Valor (Define Value):

- **El valor** en la metodología lean es aquello por lo que el cliente está dispuesto a pagar,
- Crucial descubrir las necesidades reales o latentes del cliente.
- Utilizar técnicas cualitativas y cuantitativas, para descubrir lo que los clientes **desean, cómo desean** que se les entregue el producto o servicio y **el precio** que están dispuestos a pagar por ello.

<p style="text-align:right;">  </p>

## 4.2. Metodología Ágil - Lean - 5 Principios

### 2. Mapeo del Flujo de Valor (Map Value Stream):

- Se centra en el reconocimiento y la representación visual del flujo de valor.
- El valor del cliente como **punto de referencia** para identificar todas las actividades que contribuyen a dicho valor
- Actividades que no aportan valor al cliente se consideran como desperdicio.
- Categorías del desperdicio: ***actividades necesarias pero no valiosas y actividades innecesarias sin valor.***
- Las primeras deben reducirse al máximo
- Estas últimas deben ser eliminadas.

## 4.2. Metodología Ágil - Lean

### 3. Creación de Flujo (Create Flow):

- El siguiente paso es garantizar que las actividades que añaden valor fluyan sin interrupciones ni retrasos.
- La *descomposición de pasos*, la *reconfiguración de procesos*, la *nivelación de la carga de trabajo*, la creación de *departamentos multidisciplinarios* y la *capacitación de empleados* para desempeñar múltiples roles son fundamentales para asegurar un flujo sin problemas.

<p style="text-align:right;">  </p>

## 4.2. Metodología Ágil - Lean

### 4. Establecimiento del Sistema de Requerimiento (Establish Pull):

- El inventario un gran desperdicio en cualquier sistema de producción.
- Limitar el inventario garantizando al mismo tiempo que los materiales y la información necesarios estén disponibles en el momento adecuado.
- Producción y entrega ***just-in-time***,
- Los productos se crean solo cuando son necesarios y en las cantidades requeridas.

<p style="text-align:right;">     
</p>
```



## 4.3. Metodología Ágil - Programación Extrema (XP)

- Prioriza la velocidad y la simplicidad.
- Implementación de ciclos de desarrollo cortos y una menor cantidad de documentación.
- Se basa en la iteración continua
- Revisión de la programación mediante diversas técnicas de planificación de reuniones.
- Mejoras y pruebas en el código desarrollado.

<p style="text-align:right;">   </p>

## 4.3.1. Valores de la Programación Extrema (XP):

### 2. Simplicidad:

- El propósito de esto es evitar el desperdicio
- Realizar únicamente lo absolutamente necesario,
- Mantener el diseño del sistema lo más simple posible para que sea más fácil de mantener, respaldar y revisar.
- La simplicidad también significa abordar solo los requisitos que conoces.

```
<p style="text-align:right;">  </p>
```

## 4.3.1. Valores de la Programación Extrema (XP):

### 3. Retroalimentación:

- Los equipos pueden identificar áreas de mejora y ajustar sus prácticas.
- El equipo construye algo, recopila retroalimentación sobre su diseño e implementación, y luego ajusta tu producto en el futuro.

```
<p style="text-align:right;">   
</p>
```

## 4.3.1. Valores de la Programación Extrema (XP):

### 4. Valentía:

- Preferencia por la acción basada en otros principios para que los resultados no sean perjudiciales para el equipo.
- Valentía para plantear problemas organizativos que reduzcan la efectividad de tu equipo.
- Dejar de hacer algo que no funciona y probar algo diferente.
- Aceptar y actuar sobre la retroalimentación.

<p style="text-align:right;">  </p>

## 4.3.1. Valores de la Programación Extrema (XP):

### 5. Respeto:

- Los miembros de tu equipo deben respetarse mutuamente para poder comunicarse entre ellos, proporcionar y aceptar retroalimentación que honre su relación y trabajar juntos para identificar diseños y soluciones simples.

<p style="text-align:right;">  </p>

## 4.3.2. Ciclo de vida de la programación Extrema (XP)

1. Se comienza por **describir los resultados deseados del proyecto** al tener a los clientes definir un conjunto de historias. Mientras se crean estas historias, el equipo estima:
  - el tamaño de cada una. Esta estimación de tamaño,
  - el beneficio relativo estimado por el cliente.

<p style="text-align:right;">  </p>

## 4.3.2. Ciclo de vida de la programación Extrema (XP)

- Si hay algunas historias que no pueden estimar porque no comprenden todas las consideraciones técnicas involucradas, pueden introducir una "**sonda**" (**spike**) para realizar una investigación centrada en esa historia en particular o en un aspecto común de múltiples historias.
- **Las sondas** son períodos de tiempo cortos y limitados en los que se lleva a cabo una investigación sobre un aspecto específico del proyecto.

<p style="text-align:right;">  </p>



## 4.3.2. Ciclo de vida de la programación Extrema (XP)

2. Se reúne para crear un plan de lanzamiento que todos consideren razonable.

- Es una primera aproximación de qué historias se entregarán en un trimestre o versión en particular.
- Las historias entregadas deben basarse en el valor que proporcionan.

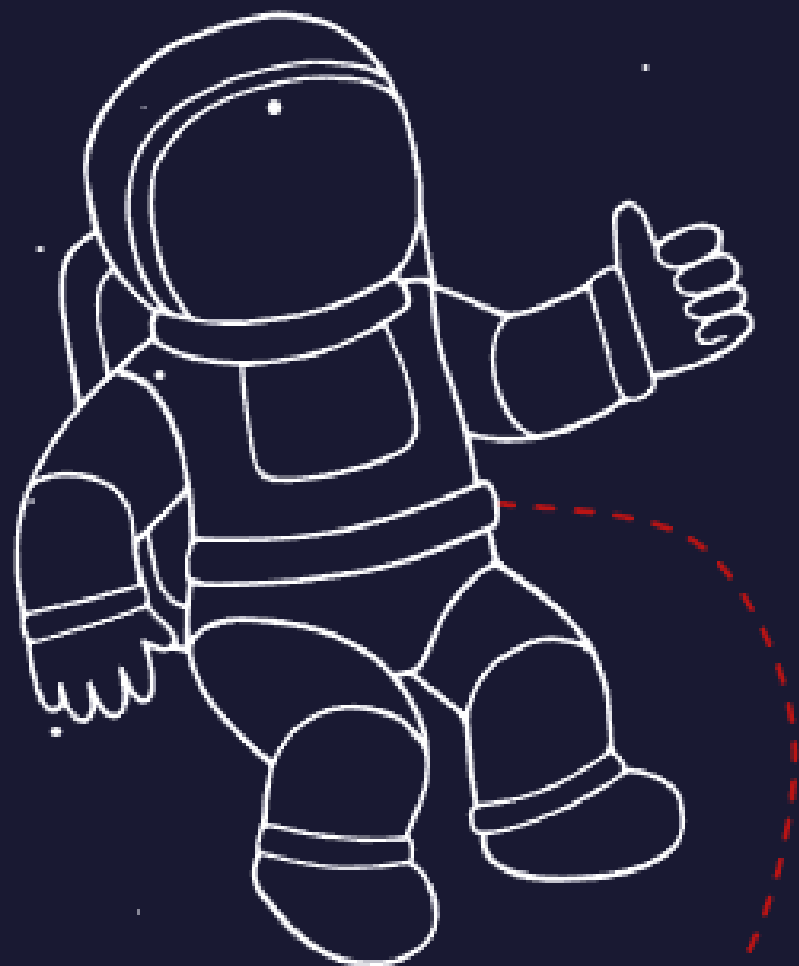
<p style="text-align:right;">  </p>

## 4.3.2. Ciclo de vida de la programación Extrema (XP)

- El equipo se sumerge en una serie de ciclos semanales.
  - **Al comienzo** de cada ciclo semanal, el equipo (junto con el cliente) se reúne para decidir qué historias se llevarán a cabo durante esa semana,
  - Luego el equipo **divide** esas historias en tareas que deben completarse dentro de esa semana.
  - **Al final de la semana**, se revisan el progreso hasta la fecha,
    - El cliente puede decidir si el proyecto debe continuar o si se ha entregado un valor suficiente.

<p style="text-align:right;">  </p>





# Programa académico CAMPUS



**Trainer**  
**Ing. Carlos H. Rueda C.**



