

# DISEÑO Y PROGRAMACIÓN ORIENTADA A OBJETOS

## - INTRODUCCIÓN A LA INGENIERÍA DE SOFTWARE

```
34 self.logout()
35 self.debug = debug
36 self.logpath = logpath
37 if path:
38     self._logpath = path
39     self.file = open(self._logpath, "w")
40     self.fingerprints = set()
41
42     @classmethod
43     def from_settings(cls, settings):
44         debug = settings.getbool("SUPERVISOR_DEBUG")
45         return cls(job_dir(settings), debug)
46
47     def request_seen(self, request):
48         fp = self.request_fingerprint(request)
49         if fp in self.fingerprints:
50             return True
51         self.fingerprints.add(fp)
52         if self.file:
53             self.file.write(fp + os.linesep)
54
55     def request_fingerprint(self, request):
56         fingerprint(request)
```

CLASE 24

# INTRODUCCIÓN A LA INGENIERÍA DE SOFTWARE

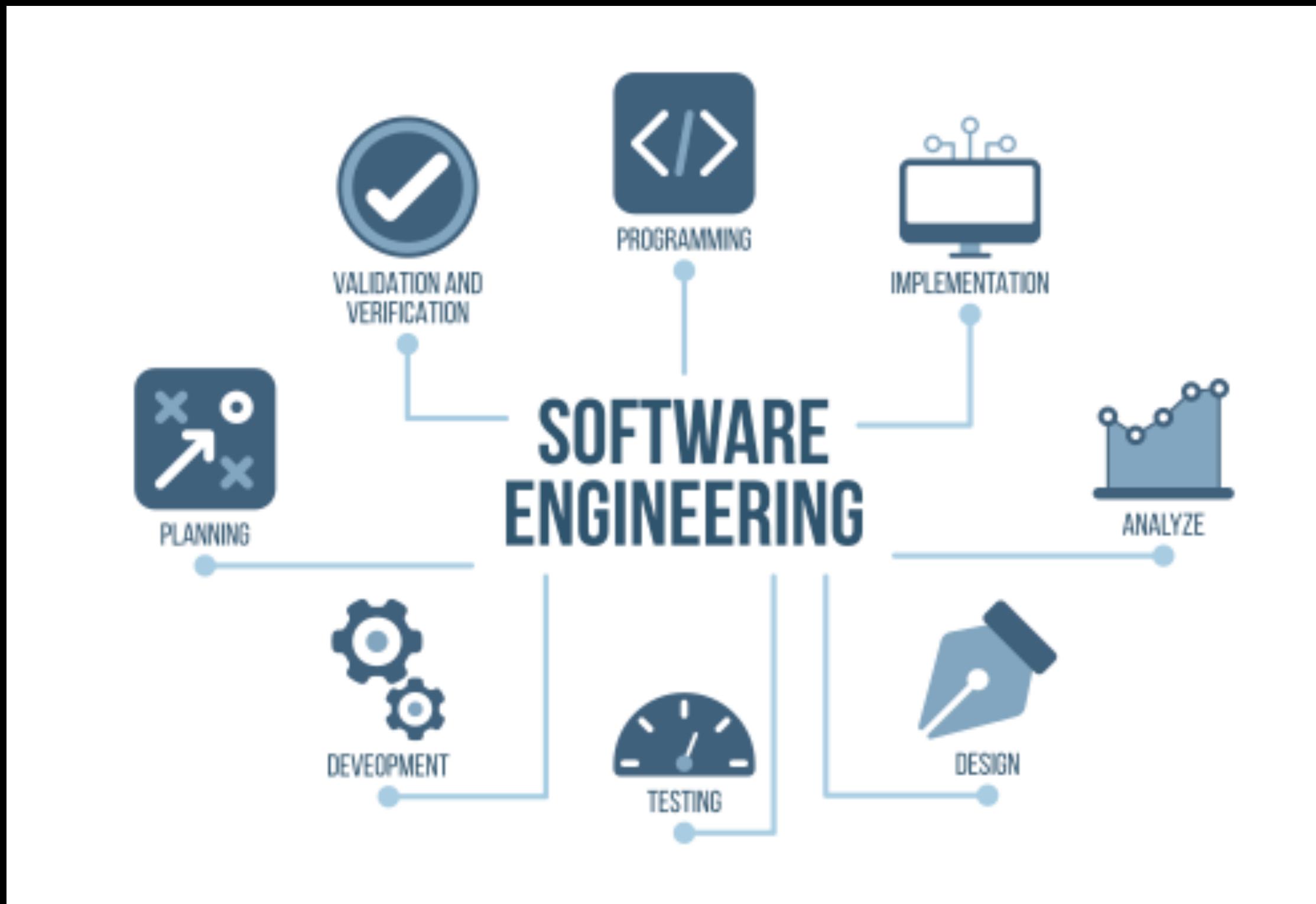
# PARTAMOS POR LA BASE... ¿QUÉ ES EL SOFTWARE?

Software es un programa o un set de programas que contienen instrucciones que ejecutan una funcionalidad deseada.

# ...¿Y LA INGENIERÍA?

Es el proceso de diseñar y construir algo que sirve a un propósito particular, siendo una solución eficiente y efectiva, minimizando el costo.

# ENTONCES, LA INGENIERÍA DE SOFTWARE ES...



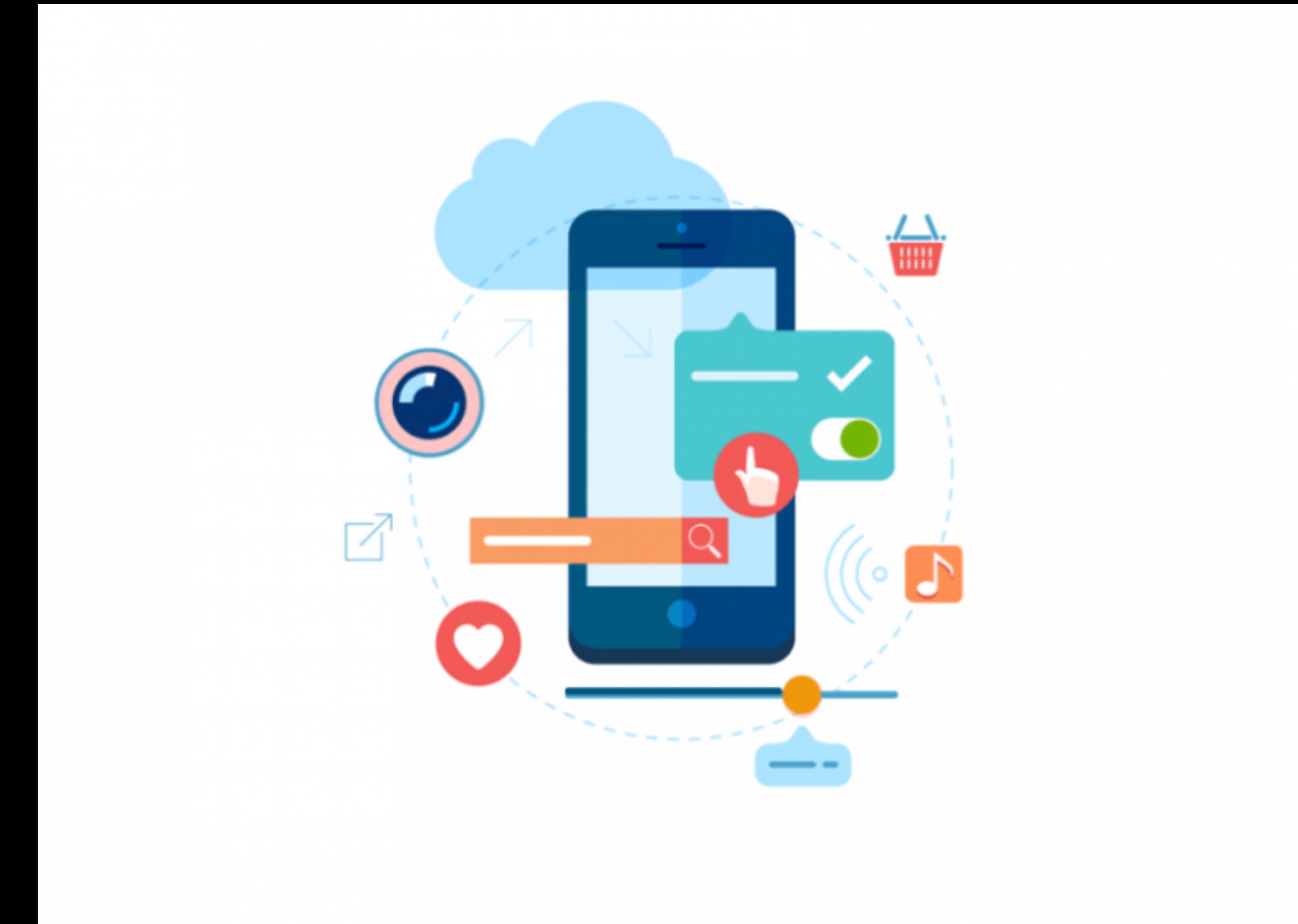
Corresponde a la disciplina que estudia la construcción sistemática, disciplinada y cuantificable aplicada al diseño, desarrollo, operación y manutención de un sistema de software

SE BUSCA ENTONCES RESOLVER UN PROBLEMA  
APLICANDO SOFTWARE, EN TIEMPOS Y COSTOS  
CONTROLADOS

# EL SOFTWARE TIENE UN ROL DUAL

## Como producto

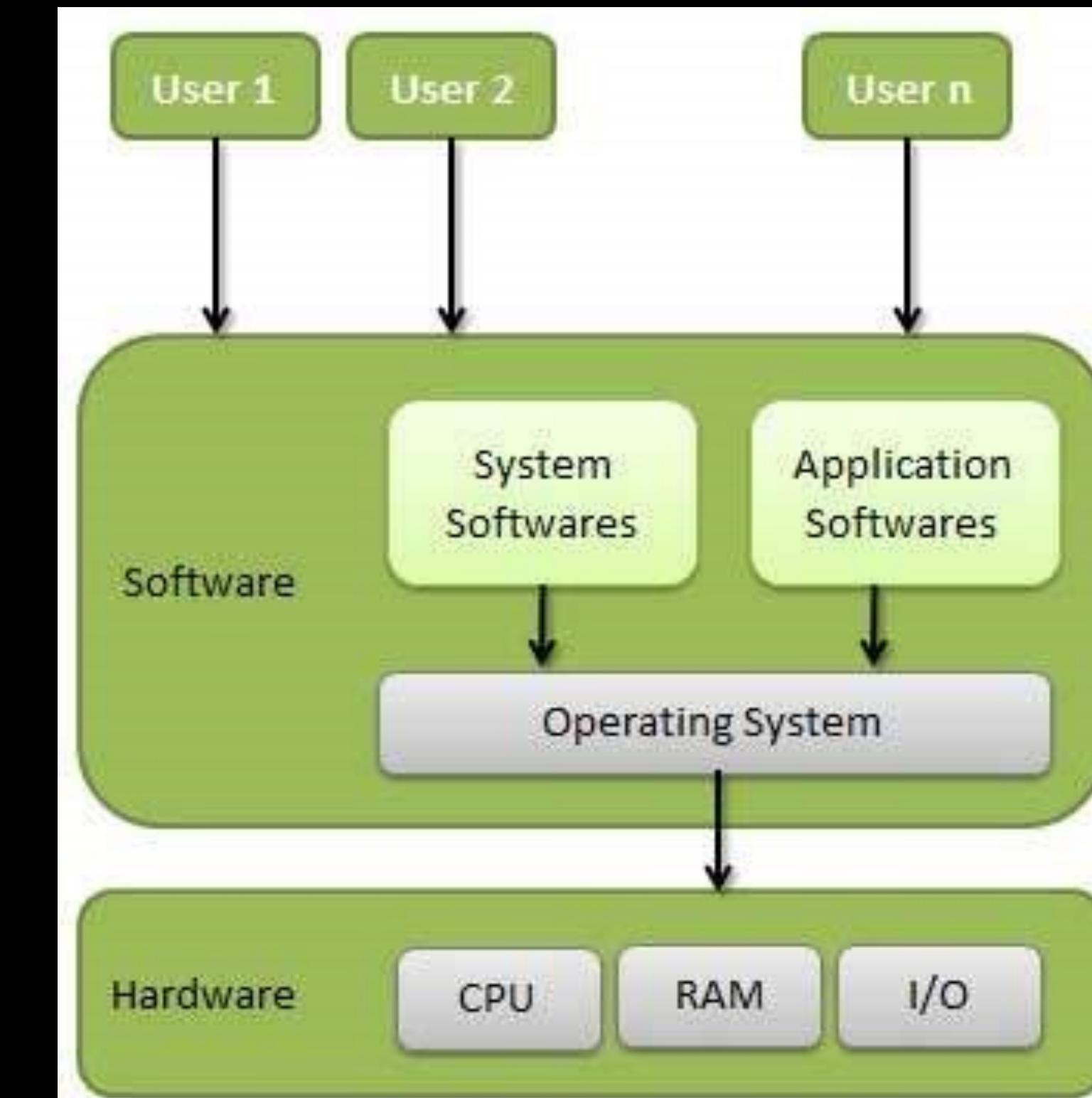
- Entrega potencial de computación a través de redes de Hardware.
- Habilita el Hardware para entregar una funcionalidad esperada.
- Actúa como transformador de la información, dado que produce, gestiona, adquiere, modifica, muestra o transmite información.



# EL SOFTWARE TIENE UN ROL DUAL

Como vehículo para construir un producto

- Provee funcionalidad sistemática (ej. Sistema de payroll).
- Controla otros software (ej. Sistema operativo).
- Ayuda a construir otros software (ej. Herramientas de desarrollo, IDE's)



# OBJETIVOS DE LA INGENIERÍA DE SOFTWARE

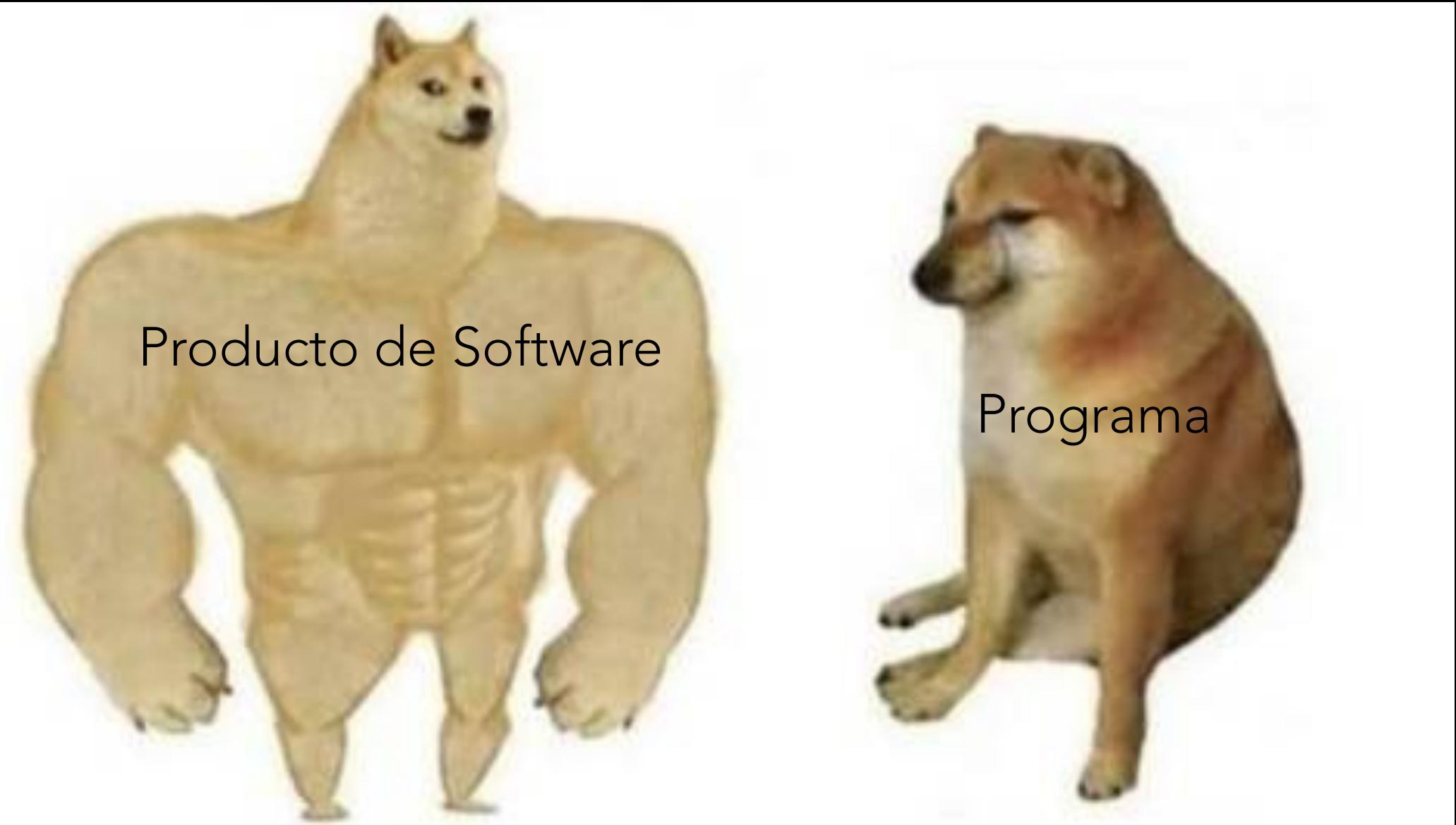
- Escalabilidad: Facilidad en la evolución a medida que los requerimientos cambian.
- Eficiencia: El software no debería desperdiciar recursos computacionales como la memoria, ciclos del procesador, etc.
- Calidad: El producto de software cumple con los criterios de aceptación especificados en los requerimientos.
- Reusabilidad: Un producto de software debe garantizar que los módulos se puedan reutilizar para desarrollar otros nuevos.
- Testable: El software debe estar encapsulado de tal forma que cada componente sea susceptible a ser testado para garantizar calidad.

# OBJETIVOS DE LA INGENIERÍA DE SOFTWARE

- Confiable (Reliability): Es un atributo de la calidad de software. Se espera que el producto pueda cumplir su función en cualquier instante de tiempo.
- Portable: El software debería ser fácilmente transferible desde un sistema computacional a otro.
- Adaptable: El producto debe poder evolucionar según los cambios que vayan surgiendo en los requerimientos.
- Interoperatividad: Capacidad de comunicación entre 2 o más unidades funcionales para procesar datos de forma cooperativa.

# PROGRAMA VS PRODUCTO DE SOFTWARE

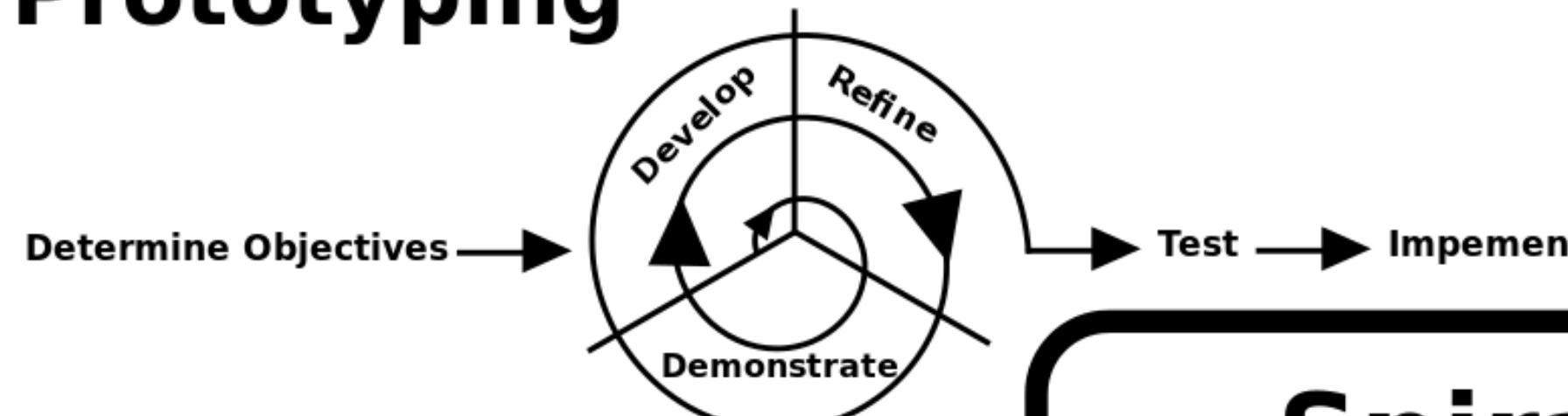
- Un programa es solo un set de instrucciones que son dadas a un computador, con el fin de completar una tarea específica; versus el producto de software, el cual es un programa (o conjunto de programas), que interactúan entre sí, disponibilidad para uso comercial, y debidamente documentado y licenciado.
- Un programa es solo una de las etapas correspondiente al ciclo de vida de un software, mientras que un producto hace referencia al ciclo de vida completo, desde su concepción a través de una idea, generación de hipótesis, creación de prototipos, validación, implementación y testing.



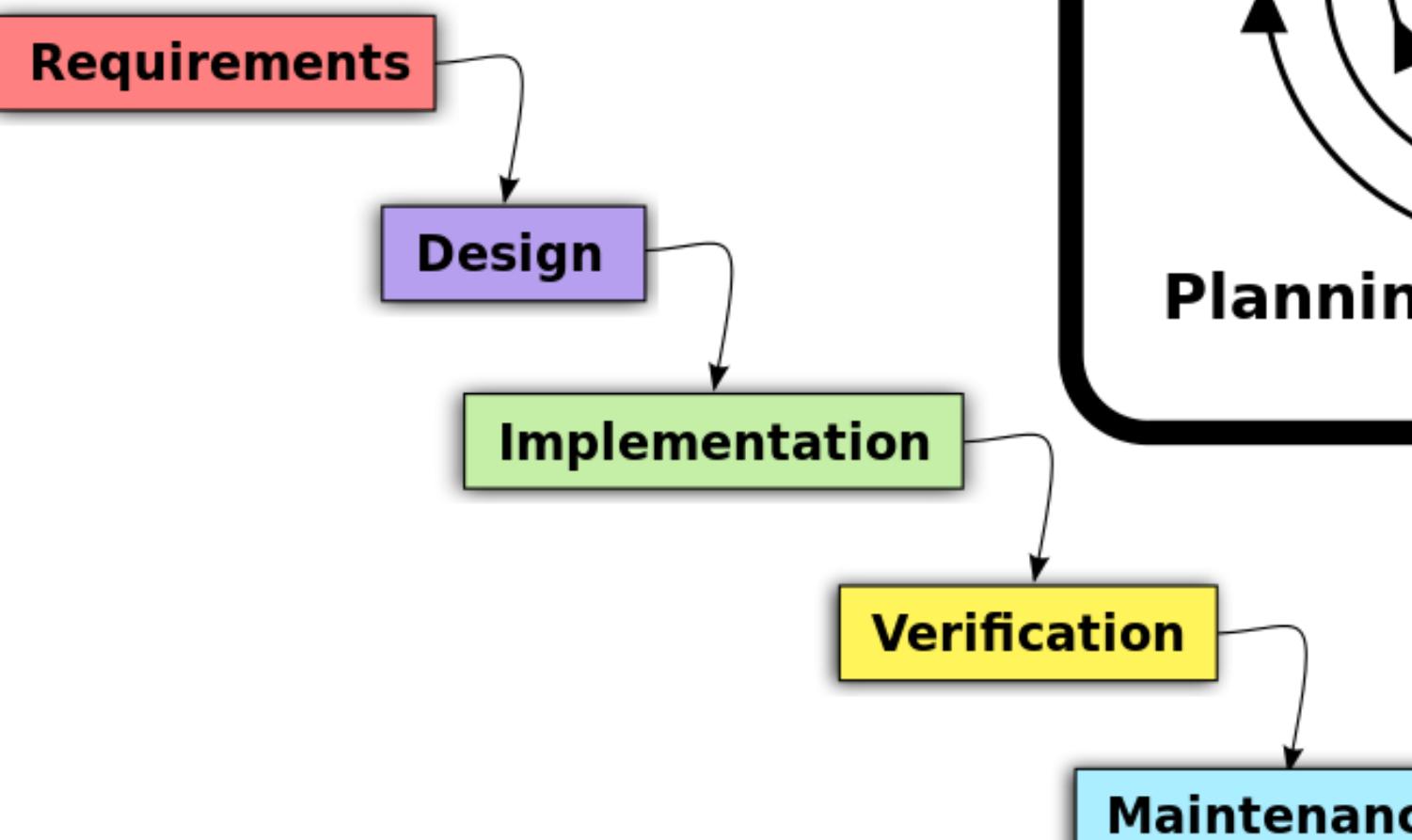
Por tanto, para desarrollar un producto de software, necesitamos valernos de una metodología de desarrollo, que sea integral, y que permita definir una estrategia de desarrollo adecuada para el lanzamiento de un producto digital en tiempo y forma.

# EXISTEN VARIAS METODOLOGÍAS

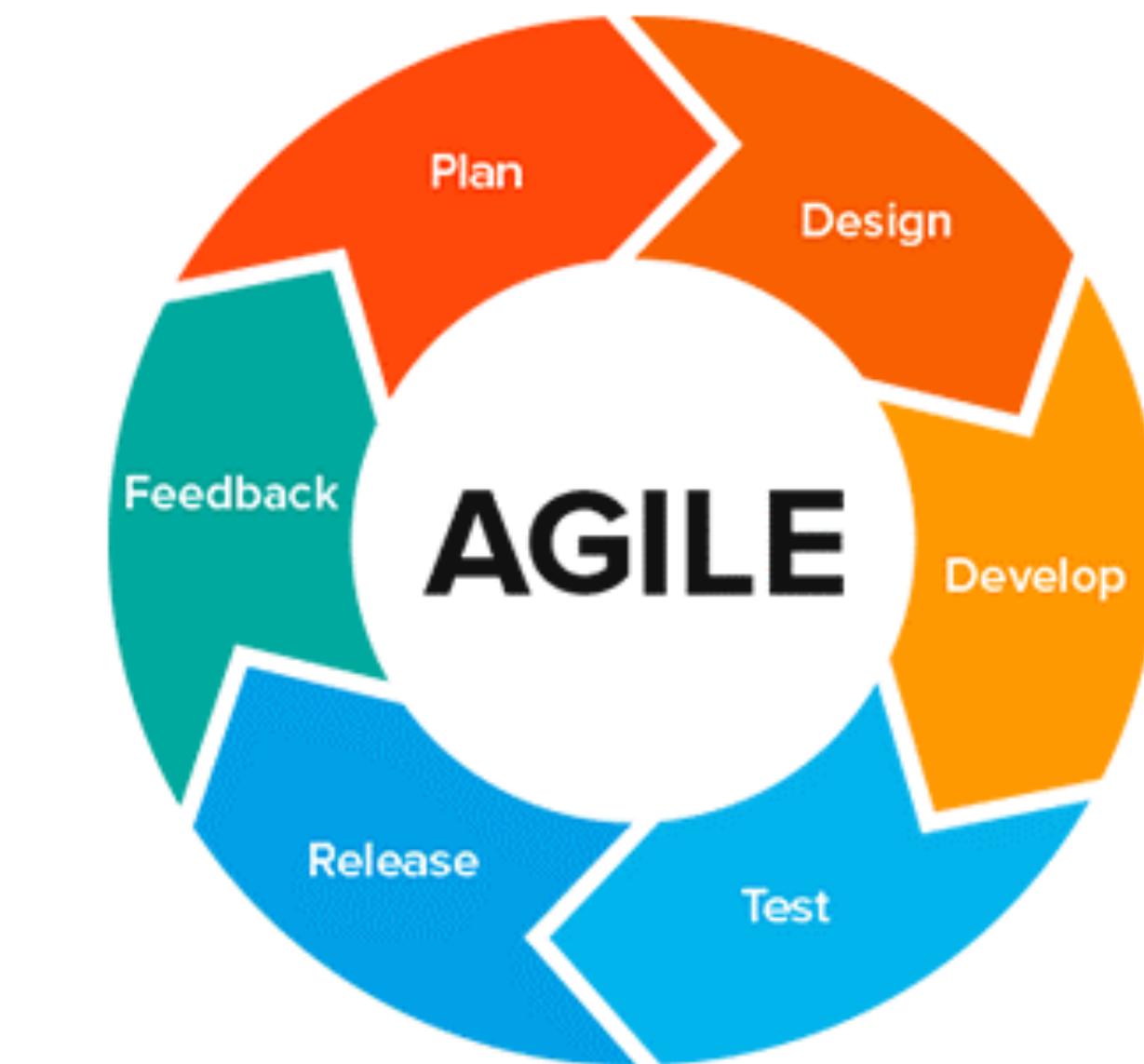
## Prototyping



## Waterfall



## Spiral



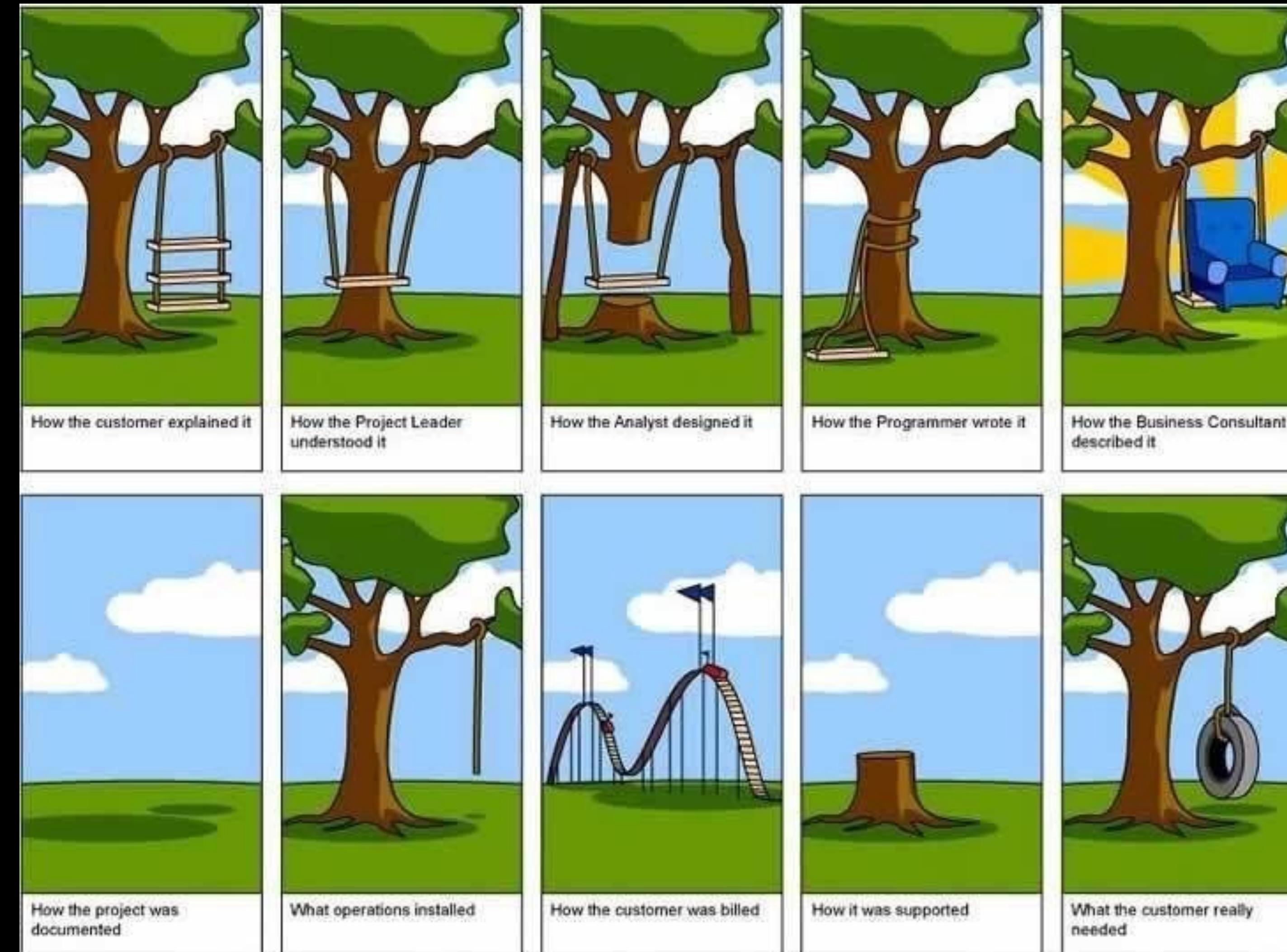
# ¿PERO CUÁL ES LA MEJOR?... DEPENDE

- En efecto, depende del tipo de producto que vayamos a construir.
- Si trabajaremos en un producto que tiene un objetivo muy específico, requerimientos bien definidos y con muy poca tendencia al cambio, en donde la gestión se pueda predecir, entonces, es mejor usar metodologías predictivas tradicionales como Waterfall.
- Por otro lado, si queremos construir un producto en un ambiente de alta incertidumbre, en donde constantemente se están validando modelos de negocio, realizando prototipos, y se necesita entrega continua de valor, lo mejor es usar metodologías ágiles de desarrollo.

EXPLOREMOS LAS DIFERENCIAS

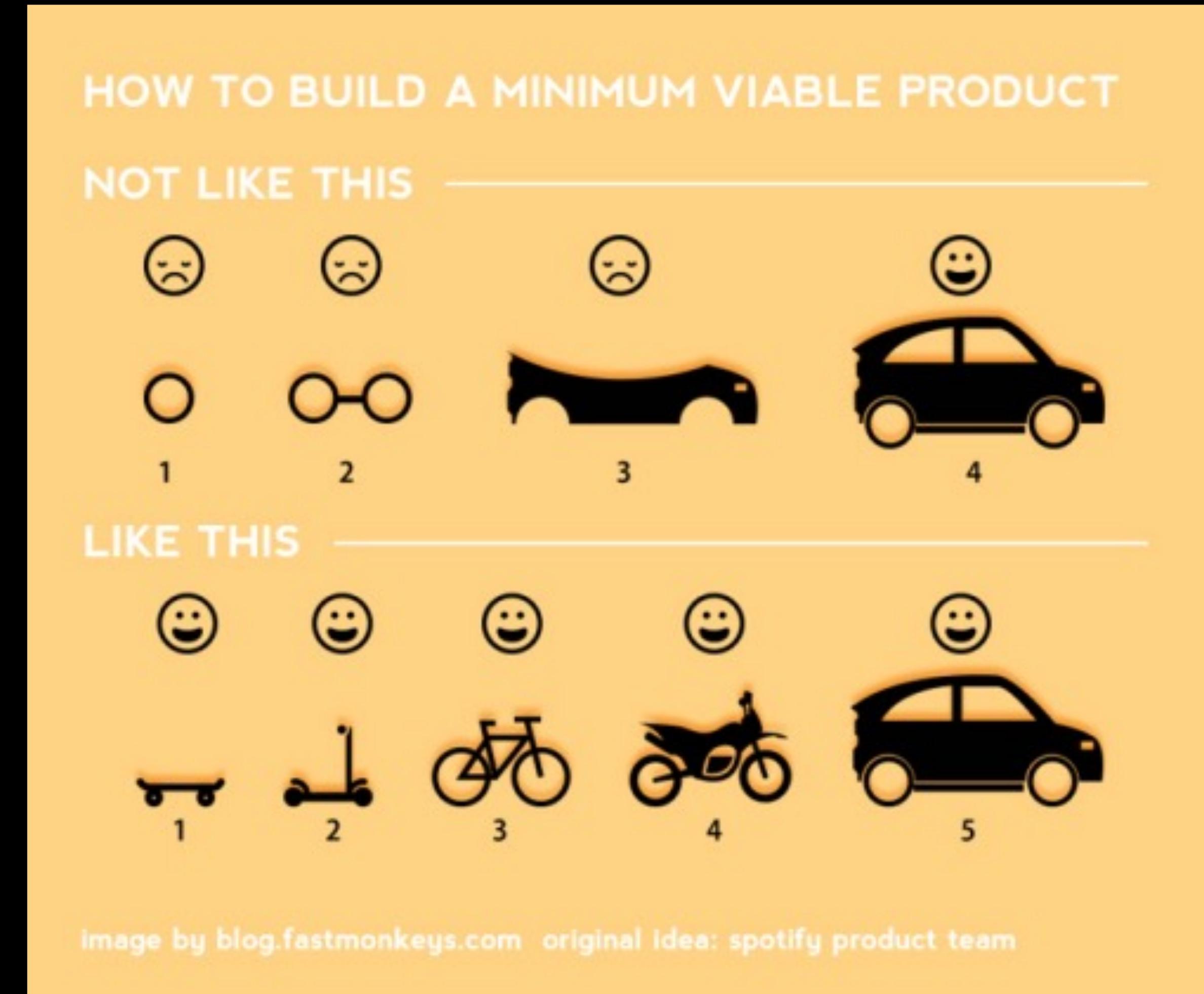
REQUERIMIENTOS

VARÍA SEGÚN LA CANTIDAD DE ACTORES POR LOS QUE PASAN LAS DEFINICIONES...

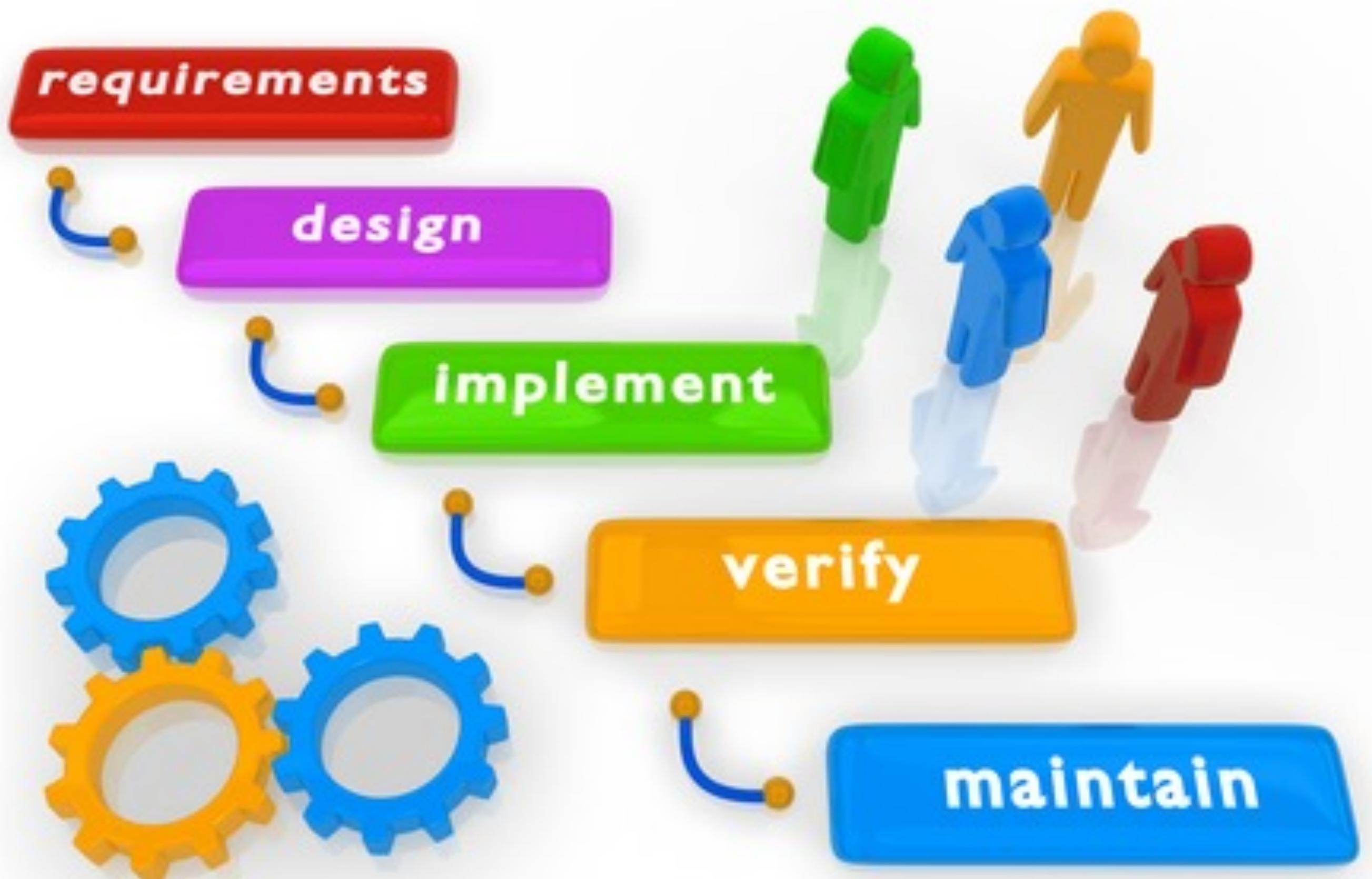


# ENTREGA TEMPRANA DE VALOR A TRAVÉS DE UN MVP

- Básicamente un MVP hace referencia a Minimum Viable Product (Mínimo Producto Viable), el cual permitirá probar una serie de hipótesis.



# ¿CÓMO ES LA GESTIÓN TRADICIONAL?



# ¿CÓMO ES LA GESTIÓN TRADICIONAL?

- Basadas en procesos bien definidos.
- Garantizar la calidad de los sistemas aplicando producción basada en procesos.
- Prácticas de ingeniería en el desarrollo de software: Estimaciones y requisitos, entre otras.
- Gestión predictiva.

# PERO... ¿CÓMO PUEDO ESTIMAR?

- Basadas en líneas de código (LOC o SLOC)
- Modelo SLIM (Ecuación de software)
- Cocomo (COnstructive COst MOdel)
- Puntos de función (Entradas, salidas del sistema, consultas, etc.)
- Wideband Delphi



AÚN CUANDO SE INTENTÓ ESTABLECER UNA FORMA DETERMINISTA DE OBTENER ESTIMACIONES, TODAS FALLAN EN INTENTAR SIMULAR UN ENTORNO REAL CON PROBLEMAS REALES...

# REQUERIMIENTOS

- Licitación de requisitos:
  - Entrevistas y cuestionarios
  - Lluvia de ideas (Brainstorming)
  - Prototipos
  - Casos de uso
  - Historias de usuario, etc.



# PROCESOS

- Tienen mayor asidero en ambientes en donde los resultados deben ser repetitivos.
- Son muy buenos para asegurar escalabilidad en estos ambientes.
- Permite la mejora continua.
- ...pero los procesos no funcionan cuando las condiciones del negocio varían continuamente.

# ANTÍTESIS DE LA GESTIÓN TRADICIONAL → AGILIDAD

- El resultado es la gestión ágil de proyectos, que no se formula sobre el concepto de anticipación (requisitos, diseño, planificación y seguimiento), sino sobre el de adaptación (visión, exploración y adaptación).

# MANIFIESTO ÁGIL

*A los individuos y su interacción, por encima de los procesos y las herramientas.*

*El software que funciona, por encima de la documentación exhaustiva.*

*La colaboración con el cliente, por encima de la negociación contractual.*

*La respuesta al cambio, por encima del seguimiento de un plan.*

# PRINCIPIOS DEL MANIFIESTO ÁGIL

- Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.
- Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo. Los procesos ágiles se doblegan al cambio como ventaja competitiva para el cliente.
- Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los períodos breves.
- Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana a través del proyecto.
- Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan y procurándoles confianza para que realicen la tarea.
- La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.

# PRINCIPIOS DEL MANIFIESTO ÁGIL

- El software que funciona es la principal medida del progreso.
- Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica enaltece la agilidad.
- La simplicidad como arte de maximizar la cantidad de trabajo que se hace, es esencial
- Las mejores arquitecturas, requisitos y diseños emergen de equipos que se auto-organizan.
- En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia.

FIN CLASE 24

GRACIAS POR SU ASISTENCIA  
¿TIENEN PREGUNTAS?