Parcial 1

viernes, 8 de septiembre de 2023 12:58

TEMAS

- **Unidad 1 completa (incluye paper No silver bullet)**
  - Introducción a la Ingeniería del Software. ¿Qué es?
  - Estado Actual y Antecedentes. La Crisis del Software.
  - Disciplinas que conforman la Ingeniería de Software.
  - Ejemplos de grandes proyectos de software fallidos y exitosos.
  - Ciclos de vida (Modelos de Proceso) y su influencia en la Administración de Proyectos de Software.
  - Procesos de Desarrollo Empíricos vs. Definidos.
  - Ciclos de vida (Modelos de Proceso) y Procesos de Desarrollo de Software
  - Ventajas y desventajas de c/u de los ciclos de vida. Criterios para elección de ciclos de vida en función de las necesidades del proyecto y las características del producto.
  - Componentes de un Proyecto de Sistemas de Información.
  - Vínculo proceso-proyecto-producto en la gestión de un proyecto de desarrollo de software.
- **Gestión de Producto**
- **Requerimientos Ágiles - User Stories**
- **Estimaciones Ágiles**
- **SCM**

¿QUÉ ES EL SOFTWARE?

El software no es solo código, se tiende a hacer una mala asociación entre que el software es sólo el programa, pero es importante incorporar que el software es un set de programas y la documentación que lo acompaña. Saber programar nada más, no alcanza para hacer software.

Tipos básicos de SW

- Software de Sistema --> se encarga de gestionar y controlar los recursos de hardware de una computadora, así como de proporcionar una interfaz para que los usuarios interactúen
- Utilitarios --> se refiere a un subconjunto de software de sistema que proporciona utilidades o herramientas adicionales para mejorar la funcionalidad del sistema operativo y el rendimiento de la computadora. Su función principal es mejorar y optimizar el rendimiento del sistema operativo y del hardware de la computadora
- Software de aplicación --> este software se utiliza para realizar tareas específicas en una computadora, como procesar texto, crear presentaciones o navegar por internet. Se enfoca en satisfacer las necesidades de los usuarios.
  - Este es por lo general con el que vamos a trabajar

SOFTWARE != MANUFACTURA

No hay manera de comparar software con una línea de producción. Tenemos 5 razones principales:

1. El software es **menos predecible**, no como los productos tangibles en una línea de producción
2. Casi **ningún producto de software es igual a otro**, ya que de alguna manera los R o necesidades de los usuarios que van a hacer uso del mismo van a ser distintas
3. **No todas las fallas en el SW son errores**. El tratamiento de errores en SW es totalmente distinto en software que en la producción.
4. **El software no se gasta**. Cuando hablamos de productos, hablamos de tiempo, en el paso de este el producto se desgasta. Si bien el SW debe ir adaptándose a los cambios y las necesidades del usuario/organización, específicamente no tiene la cualidad de desgastarse.
5. **El software -obviamente- no está gobernado por las leyes de la física**.

Evidentemente, todas las diferencias están ligadas a la intangibilidad del software como producto. A demás de estas razones hay muchas más

PROBLEMAS AL DESARROLLAR/CONSTRUIR SOFTWARE

1. Más tiempo y costo que los presupuestados
   - problema más repetido al desarrollar SW
2. La versión final del producto no satisface las necesidades del cliente
   - quizás si es lo que el cliente pidió, y no es que nosotros entendimos mal, sino que el cliente puede haberse explicado mal, puede que se haya olvidado de algo, o que cambie de opinión
3. Problema en la escalabilidad y/o adaptabilidad del software
   - agregar una funcionalidad en otra versión es casi una misión imposible. Muchas veces se parte de un prototipo que se va mejorando y haciendo que funcione, entonces en términos arquitectónicos es muy difícil hacer que eso escale???
4. Mala documentación
   - desactualizada y que no acompaña al software, por ende trae inconsistencias en la gestión o mantenimiento del mismo.
   - documentación va desde lo más pequeño, como un comentario en el código.
5. Mala calidad del SW
   - muchas veces el software de calidad se cree relacionado con la cantidad de testing que se realizó sobre él, pero no es así.

Estas son las premisas de las nuevas formas de trabajar -como los frameworks ágiles, scrum, lean, etc.-
Como se descubrió que estos eran los factores de éxito dentro del desarrollo de SW, se buscó hacer que los mismos quedaran embebidos dentro del proceso de desarrollo

Uno de los aspectos claves que tiene que ver con los R, no solo con dejarlos claros, sino también con incentivar a que el usuario tenga participación dentro del proceso para ir aclarando los mismos, ya que los R van "madurando" a medida que avanzamos en el desarrollo del sistema
También debemos tener en cuenta la retroalimentación, algo súper importante para ir midiendo el avance del sistema y contar con información relevante para poder corregir fallas o errores y hacer mejoras

El manifiesto ágil y metodologías como Scrum nos hablan sobre equipos capacitados y con habilidades cuya participación dentro del desarrollo del sistema sea competente.

Podríamos decir que el software no es exitoso o resulta en costos más elevados cuando hacemos lo contrario a lo anterior.

- Tenemos R incompletos 13,1%
- R cambiantes 8,1%
- Cuando hay poco involucramiento del usuario 12,4%
- Cuando no tenemos los recursos suficientes 10,6%
- Cuando ponemos expectativas poco realistas y somos demasiado optimistas 9,3%
- Cuando no tenemos suficiente apoyo por parte de la gerencia 8,7%

EL PROCESO DE SOFTWARE

Un proceso de software se define como un conjunto estructurado de actividades que, a raíz de un conjunto de entradas o inputs, producen una salida. Estas actividades varían dependiendo el tipo de sistema y la organización que lo desarrolla. El proceso debe ser explícitamente modelado si va a ser administrado y llevado a cabo.

- Hay diferentes perceptiones de lo que es un proceso de desarrollo de SW, pero el objetivo siempre va a ser obtener un producto o servicio de SW a partir de los inputs.

No solo vamos a tener como entrada los R, que nos van a definir el alcance, sino también personas, que son el principal recurso/capital que se consume en el desarrollo, energía, equipamiento, materiales, y procedimientos.

Nuestro objeto es a raíz de que todas estas entradas colaboren para generar un producto de SW que aporte valor.

IEEE --> un **proceso** es una secuencia de pasos ejecutados para un propósito dado
- esta definición no es lo suficientemente clara y completa, es general. Por lo que nos basamos en...

CMM --> un **proceso** es un conjunto de actividades, métodos, prácticas y transformaciones que se usan para desarrollar o mantener software y sus productos asociados.

Dentro de esta definición podemos vincular más conceptos

1. Procedimientos y métodos: definición de cómo vamos a llevar a cabo las actividades de desarrollo
2. Personas: con habilidades, entrenamiento y motivación
3. Herramientas y equipos

Las personas hacen uso de las herramientas, equipos, procedimientos y métodos y producen un producto de SW

¿Quién define el proceso de desarrollo de SW?
La organización, en función de los objetivos y de las características del SW

PROCESOS DEFINIDOS

Estos se basan en una concepción basada en el modelo industrial (inspirados en líneas de producción), en la cual tengo un conjunto de inputs que van a ser utilizados para un conjunto de actividades, que van a producir el mismo resultado una y otra vez, siempre que los inputs sean los mismos.
Esto difícilmente es aplicable al software, ya que no es tan definible.
No podemos simplificar el SW a algunos inputs, que usados para determinadas aplicaciones producen el mismo output, es decir que el output es predecible

Este tipo de procesos asumen que podemos repetirlo indefinidamente y obtener los mismos resultados.
Muchas corrientes y procesos confían en estas premisas para llevar a cabo un proyecto de SW, como por ejemplo el PUD.

PROCESOS EMPÍRICOS

Viene del empirismo, es aquella corriente en la cual centramos el foco en la experiencia.
No solo la experiencia del usuario, sino también la experiencia en el negocio la técnica, en cómo voy a manipular las herramientas y aprender de los errores.
Los tomamos en contraste con los procesos definidos

Confiamos en que las variables no pueden ser definidas o controladas en su totalidad, estamos dentro de un ambiente complejo donde quizás no todo lo que está en juego, está bajo nuestro control y esto nos implica la necesidad de adaptación para poder garantizar la producción de un software de calidad.

Se basan en la experiencia y el aprendizaje, y la principal diferencia con los procesos definidos es la forma en la cual se puede mejorar el proceso.
Mientras que en un proceso definido el punto de mejora es el central (las actividades/etapas intermedias, ya que mis entradas y salidas son fijas), el proceso empírico no tiene un punto de mejora tan tangible, más bien voy a tener que ir trabajando sobre distintos aspectos, siempre centrándonos en las personas, que son la clave del empirismo, son quienes capitalizan la experiencia.
Esta capitalización de la que se habla se hace en un ciclo de retroalimentación que permite la adaptación y mejora, **aprendemos de la experiencia** --> tomo información, la contrasto con lo que se esperaba obtener, y en base a eso voy a ir haciendo adaptaciones y mejoras.

Experiencia técnica --> una de las claves para producir un producto de calidad, es necesario conocer un poco de todo.

PATRÓN DE CONOCIMIENTO DE PROCESOS EMPÍRICOS

Primero empezamos con el planteo de una hipótesis (ASUMIR), luego se sigue con la etapa de construcción (CONSTRUIR) y en base a ciertos puntos que podemos inspeccionar y analizar, tomamos mediciones, obtenemos información, y haciendo (RETROALIMENTACIÓN) revisamos (REVISAR) el resultado de la construcción.
Si contrastamos la hipótesis inicial con el resultado del proceso, podemos ir adaptándolo y continuar con su construcción (ADAPTAR).

Esto se basa en una concepción que necesita de repeticiones, es por esto que dentro de los procesos empíricos damos uso a un ciclo de vida específico

CICLOS DE VIDA

Un ciclo de vida es una abstracción.
Tiene dos características: etapas y fases, y como estas se van a llevar adelante, pero son una abstracción porque no me dice que tengo que hacer, ni quien lo va a hacer, ni las herramientas, procedimientos, solo las fases y en qué orden se van a ejecutar.

Son una serie de pasos a través de los cuales un producto o proceso progresa. Esto nos indica que tanto los productos como los proyectos tienen ciclos de vida.

Según las características del proyecto y del producto que se va a construir, se va a elegir el ciclo de vida que se va a implementar para poder llevarlo a cabo de la mejor forma posible.
Siempre va a depender de que beneficios tiene el mismo según costos asociados y el entorno en el que nos encontremos.

RELACIÓN ENTRE PRODUCTO, PROCESOS DE DESARROLLO Y CICLOS DE VIDA

El PROCESO es una implementación del CICLO DE VIDA -ya que nos dice cómo llevar adelante cada una de las etapas- que tiene un objetivo que lo guía, que es la creación de un PRODUCTO -que debe ser no ambiguo y alcanzable-

CICLO DE VIDA DE UN PROYECTO DE SOFTWARE

Es una representación de un proceso. Grafica una descripción del proceso desde una perspectiva en particular. El ciclo de vida de un proyecto nos define las fases del proceso y el orden en que se van a llevar a cabo.
- Empieza y termina cuando genera un producto

CICLO DE VIDA DE UN PRODUCTO DE SOFTWARE

El ciclo de vida de un proyecto termina cuando genera un producto, cuyo ciclo de vida no va a terminar, sino que va a mantenerse mientras que el objeto exista.

¿Cómo? --> el producto siempre va a necesitar mantenimiento una vez que "terminó" de desarrollarse, y se va a mantener bajo este proceso de mantenimiento hasta que se desheche, recién ahí podemos decir que el ciclo de vida de un producto llegó a su fin.

Hay tres ciclos de vida básicos: secuencial, iterativo, recursivo

SECUENCIAL

Se basan en ejecutar una etapa después de otra sin retorno generalmente.
[Las fases deben estar bien definidas: requisitos, diseño, implementación, pruebas, integración, verificación y mantenimiento.
- Los R se definen al comienzo del ciclo y son fijos. Cambiar los requisitos en etapas posteriores puede ser costoso y complicado, mismo cuando se identifican problemas o errores en etapas tardías, ya que hay que rehacer el trabajo.
- La falta de flexibilidad para ajustar requisitos o realizar cambios importantes una vez que ha comenzado una fase es una limitación, por lo que hay que tener cuidado al elegirlo si mi proyecto tiene R propensos a cambios.
- Debido a su naturaleza secuencial, se hace énfasis en la documentación, ya que sirve como base para la fase siguiente

Hay algunas que etapas que pueden llegar a devolver información, como los ciclos de vida en cascada.

ITERATIVO/INCREMENTAL

Los procesos empíricos en su mayoría implementan ciclos de vida iterativos e incrementales para poder dar la característica de adaptación y mejora del proceso. Esto permite que en cada una de las iteraciones se capitalice lo aprendido, se mejora la experiencia del equipo y se realice otra iteración para incrementar el producto.
- Hago iteraciones para ganar información, y convertir esa información en un incremento que será incorporado en el sistema.
- Cada iteración revisa y mejora la versión anterior

Es flexible y permite cambios frecuentes en los R a medida que se descubren o evolucionan.
Se obtiene retroalimentación temprana del cliente o usuarios a medida que se entregan las iteraciones, lo que ayuda a refinar y ajustar el producto a lo largo del tiempo
Los problemas y riesgos se pueden abordar en etapas tempranas, lo que reduce la posibilidad de que se conviertan en problemas graves más adelante.

RECURSIVO

Se utiliza para casos particulares como proyectos de alto riesgo, ya que planifica detalladamente y así identifica problemas potenciales en etapas tempranas.
- [El desarrollo se divide en fases, y estas en subfases. Cada subfase representa un ciclo de desarrollo más pequeño y se centra en desarrollar una parte específica del sistema.
- Es menos flexible en términos de cambios en los R una vez que se ha comenzado una fase. Son costosos y complicados en fases posteriores.
- Se requiere una planificación detallada desde el principio, ya que las subfases y fases se definen con anticipación]

Dentro de este tipo de ciclos tenemos
- Espiral, que se basa en la investigación de riesgos
- B, que permite hacer una vuelta en particular en cada una de las etapas e ir avanzando en los incrementos

Se basa en tomar una característica específica y en una iteración poner todo el foco en esa funcionalidad, en la siguiente en otra, y así
Estan en DESUSO ya que se ha comprobado que no son útiles y no funcionan.

COMPONENTES DE UN PROYECTO DE DESARROLLO DE SOFTWARE

Se suele decir que el proyecto es una instanciación del proceso definido en un ciclo de vida

Un PROYECTO es llevado a cabo por PERSONAS que implementan HERRAMIENTAS para automatizar los PROCESOS y que tiene como resultado
un PRODUCTO

El PROCESO es una plantilla, una definición abstracta que se materializa a través de los PROYECTOS, en donde se adapta a las necesidades concretas del mismo. En un proceso se expresa en forma teórica que se debe hacer para hacer software, y debo adaptarlo al proyecto (en caso de ser un proceso definido) o terminar de armarlo (proceso empírico).
Los procesos toman como entrada los R y a través de un conjunto de actividades guiadas por un objetivo, obtienen como salida un producto o servicio

El PROYECTO es la unidad organizativa/de gestión y necesita de personas, recursos y procesos para poder existir. A través de este, administro los recursos que tengo y las personas que van a formar parte, para obtener como resultado un producto de SW
Tiene determinadas características

1. Duración limitada: son temporales, cuando se alcanza el/los objetivos, el proyecto termina. Tiene inicio y fin
2. Orientado a un objetivo: esto me permite que cada proyecto sea único, los objetivos tienen que ser claros, no ambiguos y alcanzables, tengo que poder definir hacia dónde voy, y poder establecer/medir cuanto alcance de ese objetivo para poder dar por finalizado el proyecto.
3. Son únicos: por más similares que sean, tienen ciertas características que los hacen únicos. Cada resultado de un proyecto es distinto al resultado de otro.
4. Tiene tareas relacionadas entre sí, basadas en esfuerzos y recursos para alcanzar el objetivo: las tareas dependen unas de otras, tienen precedencia y a su vez tienen asociados/designados recursos, lo que hace que la gestión de proyectos sea una tarea compleja.

Cuando trabajamos en un proyecto de un proceso definido, lo llamamos...

GESTIÓN TRADICIONAL DE UN PROYECTO

Se basa en ejecutar las actividades definidas -con todo lo que necesito- para lograr el objetivo

La ADMINISTRACIÓN DE PROYECTOS o Project Managment PM en términos tradicionales, es la aplicación de conocimientos, habilidades, herramientas y técnicas a las actividades del proyecto para satisfacer los R de este y poder obtener como resultado el producto esperado.

LA TRIPLE RESTRICCIÓN

Alcance: son los R que el cliente expresa, el objetivo, lo que quiero alcanzar

Tiempo: tiempo que debería llevar completar el proyecto

Costo: costo en recursos y dinero

Es una de las bases de la GTP. Plantea que en un mismo proyecto vamos a tener cierta restricción de tiempo, una restricción de presupuesto y un alcance.
- Es responsabilidad del PM balancear los tres objetivos que están involucrados en el proyecto y que compiten entre sí dentro del mismo, y definir el tiempo y el costo dado el alcance del producto.

Estas variables están íntimamente relacionadas y dependen unas de otras de manera tal que, si modifico una de ellas, voy a tener que hacer variar alguna otra o ambas.
- El balance afecta directamente la CALIDAD del proyecto, la cual NO ES NEGOCIABLE.

Proyectos de alta calidad entregan el producto requerido, satisfaciendo los objetivos en el tiempo estipulado y con el presupuesto planificado

ROLES DENTRO DE UN PROYECTO

En la gestión tradicional nuestro equipo de proyecto está formado por un EQUIPO propiamente dicho y un LIDER DE PROYECTO PM

LIDER DE PROYECTO
Project Manager

Es quien se ocupa de todas las tareas de gestión que hacen que el equipo de proyecto sea guiado a lograr el objetivo. Es quien se relaciona con el ente que regula el avance del proyecto y el propio equipo, y con los demás involucrados.
Es el encargado de administrar todos los recursos, organizar el trabajo de cada una de las personas involucradas y hacer el seguimiento de la planificación verificando que todo vaya según lo planeado.

Administrar un proyecto incluye poder definir los R, el alcance del producto, establecer los objetivos claros y alcanzables y adaptar las especificaciones, planes y el enfoque a los diferentes interesados -stakeholders-.
Desde un principio el PM debe saber en términos de negocio o del producto de SW qué tenemos que construir y qué es lo que más pesa o lo más importante del proyecto, si cumplir con el tiempo, si no superar el costo o no tocar el alcance.
- En los procesos empíricos, el PM no es necesario, ya que los proyectos se autogestionan. Este es un rol propio de la gestión tradicional de P.

EQUIPO DE PROYECTO

Se trata de un grupo de personas comprometidas en alcanzar un conjunto de objetivos de los cuales se sienten responsables. Todo el equipo apunta al mismo objetivo, que debe ser alcanzable y medible.

Para poder llegar a cumplir el objetivo, el equipo debe poder **complementarse**, es decir, contar con distintos conocimientos y habilidades que sumen y hagan la diferencia entre el trabajo en equipo y un trabajo individual.
- También deben poder trabajar en equipo sin problema, por lo que en general se eligen grupos pequeños para mejorar y fomentar la comunicación.

Para llevar adelante el P, necesitamos un plan de P.

PLAN DE PROYECTO

Es como una hoja de ruta. Para comenzar con un proyecto debemos delinear un plan de P, que no va a estar completamente definido, pero va a esbozar que es lo que se va a hacer, el alcance, el objetivo, que recursos se tendrán disponibles, en que tiempo se desarrollará, cuáles son las personas involucradas, sus roles, funciones, etc.
- Lo primero que vamos a hacer entonces, es hacer un plan detallado que nos servirá como guía para encarar el P

A través del plan de P, documentamos:
QUÉ es lo que hacemos --> alcance
CUÁNDO lo hacemos --> calendarización
CÓMO lo hacemos --> recursos y decisiones disponibles
QUIÉN lo va a hacer --> asignación de tareas

La idea del plan es que sea algo **vivo** a lo largo del P, que se vaya nutriendo y corrigiendo a medida que se lleva a cabo. Nos va a servir siempre que esté actualizado, ya que en el entorno de desarrollo, nos vamos a encontrar con variaciones que no estimamos al determinar el plan de P debido a la incertidumbre inicial

El PM arma el PLAN DE PROYECTO

PLANIFICACIÓN DE PROYECTOS DE SOFTWARE

Cuando hablamos de planificar el P incluimos

1. DEFINICIÓN DEL ALCANCE
2. DEFINICIÓN DEL PROCESO Y CICLO DE VIDA
3. ESTIMACIÓN
4. GESTIÓN DE RIESGOS
5. ASIGNACIÓN DE RECURSOS
6. PROGRAMACIÓN DE PROYECTOS
7. DEFINICIÓN DE MÉTRICAS
8. MONITOREO Y CONTROL

1. DEFINICIÓN DE ALCANCE

Es importante hacer una distinción entre proyecto y producto de SW

Alcance del proyecto	Alcance del producto
Es todo el trabajo y solo el trabajo que debe hacerse para entregar el producto o servicio con todas las características y funciones especificadas en el objetivo. El cumplimiento del alcance del proyecto se mide contra el plan de proyecto	Son todas las características que pueden incluirse en un producto o servicio. El cumplimiento del alcance del producto se mide contra la especificación de requerimientos

La relación que hay entre ambos es que, si el producto que yo tengo que construir es grande, las tareas a definir en el proyecto van a ser más o van a requerir más tiempo y recursos, por lo tanto, primero tengo que definir el alcance del producto y luego el del proyecto

2. DEFINICIÓN DE PROCESO Y CICLO DE VIDA

Cuando iniciar el proyecto debo definir qué proceso y qué ciclo de vida quiero usar
- El proceso de desarrollo es el conjunto de actividades que necesito ejecutar para construir el producto de SW, y el ciclo de vida es de qué manera ejecuto esas actividades

Dentro de esta gestión tradicional y los procesos definidos, podemos elegir el proceso y cualquier ciclo de vida, en gestión ágil tenemos limitaciones (solo se puede elegir ciclo de vida iterativo)

3. ESTIMACIÓN

Cuando vamos a hacer un plan de P, vamos a estimar todo, debemos definirlo antes de hacerlo, ya que no tenemos certeza de nada.
Tenemos que ver de qué manera estimar todas las características necesarias para la planificación. Esto lo vamos a hacer con un cierto nivel de incertidumbre, pero vamos a intentar definir en orden lo siguiente:

1. Tamaño: definir el producto a construir
2. Esfuerzo: horas persona lineales (ideales)
3. Calendario: determinar qué días y horas se va a trabajar y cuantas personas van a trabajar
4. Costo: determinar el presupuesto necesario
5. Recursos críticos: tanto humanos como físicos que nos van a hacer falta.

Normalmente cuando hablamos de estimación, también se definen rangos, no es de manera taxativa.
- A medida que se avanza en el desarrollo del P, la incertidumbre se va achicando y voy teniendo más certezas.

4. GESTIÓN DE RIESGOS

Un riesgo es algo que podría o no suceder, pero que si sucede puede comprometer el éxito del proyecto. Lo que tenemos que hacer es listar/identificar los riesgos que son más probables de ocurrir o los que más impacten en el sistema, ya que es imposible gestionar todos (son infinitos)
- Lo que hacemos es multiplicar estas dos variables de impacto y ocurrencia y obtenemos así lo que se llama **EXPOSICIÓN DE RIESGO.** Esta nos va a permitir determinar los que vamos a tomar como principales y elegir cuales se van a gestionar.

Una vez identificados los riesgos, se gestionar, esto es, generar acciones para disminuir el impacto o evitar/frenar la ocurrencia.

Hay algunos riesgos donde no podemos incidir sobre la probabilidad de ocurrencia o sobre el impacto, lo que debemos hacer es encontrar la manera de alivianarlo.

A medida que avanzamos en el P, los riesgos pueden aumentar, disminuir, ser más probables o aparecer nuevos (ya que todo lo que es plan de P va modificándose y necesita constante actualización)

5. ASIGNACIÓN DE RECURSOS
6. PROGRAMACIÓN DE PROYECTOS

7. DEFINICIÓN DE MÉTRICAS

Como nos basamos en estimaciones para planificar, las métricas de SW nos van a permitir saber, una vez que el P se esté ejecutando, si nuestro proyecto está en línea con lo planificado o si está desviado.
- Esto lo hago midiendo la realidad, lo que está pasando y en base a eso saber si estamos bien o no.
- Si una métrica no tiene un fin en particular, no tiene sentido tomarla.

Se clasifican de acuerdo a su dominio

MÉTRICAS DE PROCESO — Con proceso nos referimos al proceso de desarrollo de software definido por la organización. Nos permiten saber independientemente de un proyecto en particular, saber si como organización estamos trabajando bien o mal.
Son básicamente las mismas métricas del proyecto, pero despersonalizadas de un proyecto en particular.
- Ej.: porcentaje de proyectos que terminan en término y porcentaje de los que terminan fuera de término

MÉTRICAS DE PROYECTO — Me permiten saber si un proyecto de SW en ejecución está cumpliendo con lo planificado o no
- Ej.: comparación del tiempo calendario real con el planificado

MÉTRICAS DE PRODUCTO — Miden cuestiones/características que tienen una relación directa con el producto que estamos construyendo
- Ej.: métricas de defectos, todas las métricas de tamaño.

MÉTRICAS BÁSICAS --> tamaño, esfuerzo, tiempo calendario, defectos.

¿Cada cuánto tomo métricas?
- Uno de los aspectos clave de las métricas es **hacer foco.** Esto quiere decir que mis métricas van a ser tomadas en base a decisiones sobre el tiempo que va a tardar en realizarse el proyecto.
- Tengo que minimizar los esfuerzos que se ejecutan para tomar métricas

El PM se va a encargar d
no. Para hacer esto se v

"Un proyecto se atrasa d
proyecto en el moment

Es muy normal en un P c
demasiado optimistas al
monitoreo y control del

FACTORES

1. Monitoreo y Feed
monitoreo permane
correctivas cuando
evitar una desviació
2. Misión/objetivo c
dónde vamos
3. Comunicación: en
con el líder de P, cor
clientes y stakehold

FILOSOFÍA ÁGIL

El agilismo o filosofía ágil es un movimiento que surge de los desarrolladores hace unos 20 años. Un grupo de referentes acordaron un conjunto de enunciados que llamaron manifiesto ágil.

El Manifiesto Ágil se trata de un compromiso que hacen todas las personas involucradas en un proyecto para trabajar de una determinada manera independientemente de las prácticas que realice cada uno.
- Esto ha sido una evolución cultural que tiene que ver con les experiencias que cada uno de los referentes ha vivido.

OBJETIVO DEL MOVIMIENTO ÁGIL
- Lograr un equilibrio entre seguir procesos rigurosos y formales en el desarrollo de SW y trabajar de una manera eficiente y efectiva para entregar un producto de calidad.
- En otras palabras, evitar los extremos de la falta de profesionalismo y la extrema burocratización, para lograr una entrega continua de SW funcional y satisfacer las necesidades del cliente

Es un compromiso útil/punto medio entre nada de proceso y demasiado proceso

El manifiesto ágil se sustenta en procesos empíricos que tienen como base la EXPERIENCIA, la cual sale del propio equipo, por ello es importante tener ciclos de retroalimentación cortos.
Esto implica comenzar con una idea o R, construir un Producto Mínimo Viable (MVP) y mostrarlo al cliente para obtener retroalimentación.
- A partir de esta, se realizan mejoras y correcciones continuas durante todo el proceso de desarrollo.

Este enfoque es compatible únicamente con ciclos de vida iterativos y se basa en la idea de que la incertidumbre y los cambios son inevitables en el desarrollo de SW, por lo que es necesario trabajar de manera flexible y adaptativa

Los procesos empíricos, centran sus bases en el empirismo. El cual tiene 3 pilares...

1. TRANSPARENCIA: se refiere a la comunicación abierta y honesta de la información relevante a todas las partes interesadas.
La transparencia promueve la confianza y la colaboración entre las partes, lo que conduce a una toma de decisiones más informada y acertada.
- Nos permite crecer como equipo, y transformar el conocimiento implícito (el que tiene cada uno) en conocimiento explícito (conocimiento que sea del equipo y se pueda transmitir)

2. ADAPTACIÓN: se refiere a la capacidad de adaptarse y ajustar el trabajo y el proceso para hacer frente a las desviaciones y problemas encontrados durante la inspección.
La adaptación permite una respuesta rápida y efectiva a los cambios y permite mantener la entrega de valor al cliente.

3. INSPECCIÓN: se refiere a la revisión regular y sistemática del progreso del trabajo y productos restantes para detectar problemas y desviaciones del plan.
La inspección permite una evaluación temprana y continua del progreso y ayuda a tomar decisiones informadas sobre los próximos pasos.

MANIFIESTO ÁGIL

Tiene 4 valores a los que llamamos VALORES ÁGILES
-ambas cosas son importantes pero las primeras se ponen por encima del resto-

1. Valorar a los individuos y las interacciones SOBRE el proceso y las herramientas
2. Valorar más el software funcionando SOBRE la una documentación extensiva
3. Valorar más la colaboración con el cliente SOBRE la negociación de un contrato
4. Valorar más la respuesta ante el cambio SOBRE seguir un plan

1
Quiere decir que se enfatizan los vínculos, la comunicación efectiva y la colaboración entre las partes interesadas y miembros del equipo, por sobre la adopción de la herramienta que tenemos que usar y el proceso a aplicar.

2
Este valor enfatiza la importancia de proporcionar software funcional y de alta calidad en lugar de enfocarse en documentar cada detalle del proceso de desarrollo.
- Esto no quiere decir que no se debe documentar, es necesario hacerlo, pero el enfoque ágil plantea la idea de generar la información que haga falta cuando haga falta.

3
Este valor enfatiza la importancia de trabajar en colaboración con el cliente y comprender sus necesidades y requisitos en lugar de simplemente negociar contratos y acuerdos formales. Debo hacer que el cliente se sienta parte.
- Está muy relacionado con la triple restricción (alcance/tiempo/costo)

Los problemas en las negociaciones se dan cuando el cliente firma un contrato con el equipo definiendo un alcance/costo/tiempo determinado y luego quiere realizar cambios en este. Estos problemas se pueden evitar si se involucra al cliente de una manera más activa en el proyecto y se fomenta una actitud de colaboración e integración.
- Es fundamental que el cliente esté dispuesto a participar, ya que de lo contrario la metodología ágil no se puede aplicar.
- Por lo tanto es importante formar tanto a los equipos que trabajarán con este enfoque, como al

8. MONITOREO Y CONTROL

e trabajar sobre lo definido en el plan de P y determinar si se está cumpliendo o
a basar en el plan y en las métricas.

le un día a la vez" --> esto quiere decir que si puedo corregir los desvíos de mi
adecuado, todavía estoy a tiempo de poder cumplir el objetivo.

omparar lo planificado con lo real. Como desarrolladores es muy común ser
momento de hacer estimaciones, por lo que es totalmente necesario hacer
desarrollo del proyecto.

PARA EL ÉXITO

back: tener un
nte y generar acciones
ta necesario para
n mayor.
aro: saber hacia

todos sus aspectos,
el equipo, con los
rs.

CAUSAS DE FRACASOS

1. Fallar al definir el problema
2. Plan basado en datos insuficientes
3. La planificación la hizo el grupo de planificaciones
4. No hay seguimiento del plan de P
5. Plan de P pobre en detalles
6. Planificación de recursos inadecuada
7. Las estimaciones se basaron en "supuestos" sin consultar datos históricos
8. Nadie estaba a cargo
9. Mala comunicación

Tiene 4 valores a los que llamamos VALORES ÁGILES

-ambas cosas son importantes pero las primeras se ponen por encima del resto-

2. Valorar más el software funcionando SOBRE la una documentación extensiva

3. Valorar más la colaboración con el cliente SOBRE la negociación de un contrato

4. Valorar más la respuesta ante el cliente SOBRE seguir un plan

(quiere decir que se entiendan los vínculos, la comunicacion efectiva y la colaboracion entre las partes interesadas y miembros del equipo, por sobre la adopción de la herramienta que tenemos que usar y el proceso a aplicar.

2

Este valor enfatiza la importancia de proporcionar software funcional y de alta calidad en lugar de enfocarse en documentar cada detalle del proceso de desarrollo.
- Esto no quiere decir que no se debe documentar, es necesario hacerlo, pero el enfoque ágil plantea la idea de generar la información que haga falta cuando haga falta..

Las decisiones tomadas con respecto al producto deben quedar documentadas y van a trascender más allá de la iteración en la cual se generó el incremento del mismo. Lo que debe decidirse es qué aspectos del producto van a quedar documentados, teniendo en cuenta que el conocimiento del producto debe ser transparente y de toda la organización, no de un individuo

Con respecto al proyecto es lo mismo, se decide que se va a documentar sobre este.
- Otro aspecto a tener en cuenta es la PERMANENCIA, no toda la información que se genera debe tener permanencia y disponibilidad infinita.

este valor entatiza la importancia de trabajar en colaboracion con el cliente y comprender sus necesidades y requisitos en lugar de simplemente negociar contratos y acuerdos formales. Debo hacer que el cliente se sienta parte.
- Está muy relacionado con la triple restricción (alcance/tiempo/costo)

Los problemas en las negociaciones se dan cuando el cliente firma un contrato con el equipo definiendo un alcance/costo/tiempo determinado y luego quiere realizar cambios en este. Estos problemas se pueden evitar si se involucra al cliente de una manera más activa en el proyecto y se fomenta una actitud de colaboración e integración.
- Es fundamental que el cliente esté dispuesto a participar, ya que de lo contrario la metodología ágil no se puede aplicar.
- Por lo tanto es importante formar tanto a los equipos que trabajarán con este enfoque, como al cliente, (en SCRUM es el Product Owner)

4

Este valor enfatiza la importancia de ser flexible y adaptativo en la respuesta a cambios y desviaciones en el proceso de desarrollo, en lugar de seguir un plan rígido y predefinido que puede no ser el adecuado para el contexto actual.
- Plantea que debemos construir en conjunto con el cliente, no definir tanto, más bien empezar a trabajar con una idea del producto e ir enfocándonos más a medida que se avanza para darle la posibilidad al cliente de cambiar de opinión.

En lugar de tener una ERS firmada por el cliente -que seguro no leyó completa- mejor tener una actitud más ajustada con la realidad de que los R cambian, la gente se equivoca, se olvida, se da cuenta de lo que quiere cuando lo ve y no por medio de un contrato.
- Comunicación frecuente y colaborativa.

Cuando hablamos de Requerimientos Ágiles, se trata de una manera de trabajar con los R que está alineada con los

12 PRINCIPIOS DEL MANIFIESTO ÁGIL

1. SATISFACCIÓN DEL CLIENTE

Nuestra mayor prioridad es satisfacer al cliente con entregas tempranas y continuas de software con valor

2. ADAPTACIÓN AL CAMBIO

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

3. ENTREGAS FRECUENTES

Entregamos software funcional con frecuencia, en preferencia a la documentación exhaustiva.

4. TRABAJO EN EQUIPO

Colaboramos con el cliente y los interesados durante todo el proyecto para asegurarnos de que el software entregado satisfaga las necesidades del negocio.

5. PERSONAS MOTIVADAS

Construimos proyectos en torno a individuos motivados. Les damos el entorno y el apoyo que necesitan y confiamos con que harán el trabajo.

6. COMUNICACIÓN DIRECTA

El método más efectivo y eficiente de comunicar información dentro de un equipo de desarrollo es la conversación cara a cara. La efectividad y la riqueza de la comunicación crece exponencialmente conforme estamos en un mismo espacio físico y compartiendo un pizarrón para construir juntos el producto.

7. SOFTWARE FUNCIONANDO

El software funcionando es la medida principal del progreso

8. CONTINUIDAD

El equipo de trabajo debe ser capaz de mantener un ritmo constante y sostenible de trabajo a lo largo del tiempo. Los frameworks ágiles apuntan a ciclos de vida iterativos de duración fija, y si no llegue con todo lo previsto en la iteración, entrego lo que tengo listo. La forma de llegar a que un equipo tenga un desarrollo sostenible, es logrear que el equipo tenga un ritmo de trabajo que asegure entregar en un ciclo de tiempo fijo.

9. EXCELENCIA TÉCNICA

La excelencia técnica y el buen diseño mejoran la agilidad. La calidad del producto no es negociable. ¿Debemos ajustar cualquier aspecto del producto menos su calidad al entregar

10. SIMPLICIDAD

Apunta a que lo mejor es lo simple. Al buscar la simplicidad, se puede maximizar la eficiencia y la efectividad, al mismo tiempo que se reduce la complejidad y el riesgo de errores y problemas. Esto se logra a través de la eliminación de tareas innecesarias y la concentración en los objetivos más importantes y críticos del proyecto. No agregar funcionalidades porque sí, si el cliente no lo pidió.

11. EQUIPOS AUTO-ORG

La calidad del producto, emergen de equipos aut diseñar soluciones de m contra de las definicione Debemos tener en cuen desarrollo tengan cierto proceso de diseño y des rígido.
El agilismo rompe totaln más aptos para definir c etc.- son los que van a d

TRIÁNGULO DE HIERRO vs TRIÁNGULO ÁGIL

(la triple restricción del enfoque ágil)

El triángulo de hierro es un modelo utilizado en la gestión tradicional de proyectos, que establece que los tres elementos fundamentales en cualquier proyecto son el alcance, el tiempo y el costo. Según este modelo, cualquier cambio en uno de estos puede afectar los otros dos.
En la gestión tradicional de proyectos, se busca establecer un plan detallado y fijo en cuanto a estos tres elementos antes de comenzar el proyecto, y se trabaja para asegurar que se cumplan en el transcurso del mismo.

Por otro lado, en las metodologías ágiles, se considera que el alcance, el tiempo y el costo son variables flexibles que pueden adaptarse y cambiarse a lo largo del proyecto. En lugar de establecer un plan fijo al inicio, se utilizan iteraciones cortas y regulares para poder adaptar el proyecto según las necesidades y los requisitos cambiantes.

La gestión ágil establece que los tres elementos fundamentales son el valor, la calidad y la restricción del tiempo. En este modelo se busca maximizar el valor que se le entrega al cliente, asegurar la calidad del producto y cumplir con los plazos establecidos, en lugar de centrarse en el alcance y el costo. A partir de esto se crea un nuevo triángulo llamado Triángulo Ágil.

Si planteamos un ciclo de vida iterativo para un proyecto definido, sabemos que vamos a dejar fijos los R y vamos a definir las iteraciones en base a porciones de funcionalidad (CU) a desarrollar en cada iteración.
Ahora bien, las iteraciones que plantea un framework ágil, se centran en el logro de entregas tempranas y por lo tanto la priorización del tiempo. Esto se plasma en la fijación del tiempo y los costos (ya que tengo un equipo fijo de personas) y la variación de las funcionalidades o alcances.

En la gestión ágil, si consideramos las mismas variables, primero me voy a guiar por el valor que le aportan las funcionalidades al desarrollo del producto. "Tego estos recursos y este tiempo, con esto fijo, ¿Cual le puedo entregar al cliente?

VALOR -> SW funcionando para el cliente

REQUERIMIENTOS ÁGILES

2 pilares

1. El foco de este enfoque apunta a construir un valor de negocio
2. Vamos a ir encontrando los R poco a poco

El enfoque ágil no solo plantea una gestión diferente del proyecto sino también una definición de R diferente

En la gestión ágil el valor de negocio y la construcción del producto correcto son fundamentales.
Se plantea que los R se irán descubriendo a medida que avanza el proyecto y se busca definir solo lo suficiente como para comenzar a trabajar con lo mínimo necesario y obtener retroalimentación lo más rápido posible para mejorar continuamente.
- En lugar de especificar todos los R desde el inicio, el desarrollo de R está enfocado en trabajar conjunto con el cliente.

Los R se expresan en forma de User Stories, que son breves descripciones de los R del cliente que se escriben en lenguaje natural y se enfocan en las necesidades del usuario. Estas stories se utilizan para definir el alcance de cada iteración del proyecto y se priorizan en función del valor que aportan al producto (esto significa que debe estar planteado el valor que tiene para el cliente, la satisfacción de cada R)
- Esto difiere del enfoque tradicional que especifica todos los R antes de avanzar. Lo que se llama BRUF o Big Requirements Up Front

El enfoque ágil no solo plantea una gestión diferente del proyecto, sino también una definición de requerimientos distinta.

Nos encontraremos con 3 enfoques:

Valor: Usaremos valor para construir el producto correcto. Es decir, nos preguntaremos a quién queremos o vamos a ayudar con la creación de nuestro producto. La idea central de la construcción de software está centrada en el hecho de construir valor de negocio.

Sólo lo suficiente: *Just in time*. La idea es saber que no tendremos nunca todos los requerimientos definidos en un principio. Dado al contexto de software el cual es volátil. Definamos los requerimientos y características a medida que avanzamos en el software.

En un principio encontremos lo mínimo y necesario y trabajemos a partir de allí para empezar a crear retroalimentación.

Una de las razones por las cuales debemos cambiar el enfoque (determinar solo lo suficiente), es porque todos los productos de software tienen un desperdicio significativo, es decir, de todas las funcionalidades desarrolladas, hay un alto porcentaje que nunca se utilizan.
- Lo que se usa siempre solo conforma un 7% del producto (pensar en cuando usamos Excel)

La gestión ágil de R intenta cambiar esta situación, priorizando solo aquellas funcionalidades que aportan valor al cliente. Para lograr esto, se utiliza el concepto de dueño de producto, PO o Product Owner.
El PO es responsable de identificar las necesidades y prioridades del negocio, y por ende, priorizar los R del producto (decidir que se necesita primero y que aporta un valor más significativo).
Estos R se organizan en una lista ordenada y priorizada llamada Product Backlog. Los R más importantes se encuentran en la parte superior de la lista.
- La organización en forma de pila del PB obliga a priorizar los R por orden de importancia.

¿Cómo vamos a trabajar con la gestión ágil de R?

Se definen los R con mayor prioridad, osea los que agregan más valor al cliente. Los que están primeros, son los que trabajaremos con mayor nivel de detalle. A medida que termino unos, paso a otros.

A medida que avanzamos, hay R que pueden ser removidos, otros que pueden subir o bajar de prioridad, o bien pueden sumarse nuevos R al PB.

Trabajamos solo con los R de alta prioridad, mientras los que se encuentran más debajo en el PB se mantienen en un estado de posibilidad de cambio en el tiempo sin que nos afecte, hasta que alcancen la suficiente prioridad como para que decidamos trabajar en ellos.

JUST IN TIME

Analizo cuando necesito, ni antes ni después (no se debe hacer tarde).
Analizo con detalle un R cuando tenga el suficiente valor.
Concepto que se usa fundamentalmente para evitar desperdicios (en este contexto sergia invertir tiempo en especificar R que después cambian).

LA DIFERENCIA MÁS MARCADA ENTRE LA GESTIÓN TRADICIONAL Y LA GESTIÓN ÁGIL SE ENCUENTRA EN LA TRIPLE RESTRICCIÓN.

METODOLOGIA TRADICIONAL vs METODLOGÍA ÁGIL

	TRADICIONAL	ÁGIL
Prioridad	Cumplir el plan	Entregar Valor
Enfoque	Ejecución. ¿Cómo?	Estrategia (¿Por qué? ¿Para qué?)
Definición	Detallados y cerrados. Descubrimientos al inicio	Esbozados y evolutivos. Descubrimiento progresivo
Participación	Sponsors, stakeholder de mayor poder e interés.	Colaborativo con stakeholders de mayor interés (clientes, usuarios finales)
Equipo	Analista de negocios, Project Manager y Áreas de Proceso.	Equipo multidisciplinario.
Herramientas	Entrevistas, observación y formularios	Principalmente prototipado. Técnicas de facilitación para descubrir.
Documentación	Alto nivel de detalle. Matriz de Rastreabilidad para los requerimientos.	Historias de Usuario. Mapeo de historias (Story Mapping)
Productos	Definidos en alcance	Identificados progresivamente
Procesos	Estables, adversos al cambio.	Incertidumbre, abierto al cambio

Lo que nosotros entregamos al cliente es valor de negocio, no características del SW. El SW es el medio por el cual nosotros entregamos valor de negocio

TIPOS DE REQUERIMIENTOS

1. De negocio --> son los de más alto nivel. Nos referimos a cuales son los R en términos de la visión del negocio.

2. De usuario --> tienen que ver concretamente con los R del usuario final.

3. Funcionales --> podríamos relacionarlos 1 a 1 con los casos de uso en la gestión tradicional.

4. No funcionales --> suelen ser los más ocultos, que el cliente no tiene en claro y debo especificarlos. Un RNF puede hacer que el software que estoy desarrollando no sirva.

5. De implementación --> se suele decir que están dentro de los NF, pero tienen que ver con cuestiones de restricción o implementación específicos.

A raíz de esta clasificación queremos identificar los que tienen que ver con el dominio del problema y de la solución.
Al enfocarse en los R del domino del problema y diferenciarlos de los que son específicos del SW, los equipos de desarrollo pueden tener una visión más clara de lo que el SW debe hacer y enfocar el esfuerzo en lo esencial para cumplir los objetivos.
- Esto es clave para el éxito de la metodología ágil y su capacidad para entregar SW de alta calidad en un plazo corto.

En otras palabras, las gestión ágil de R implica trabajar con técnicos y no técnicos entendiendo las necesidades y el negocio para construir un SW que de valor. Y luego junto con los usuarios descubrir cual es la mejor forma de satisfacer esas necesidades (acá aparece el PB)
Entregada la característica, obtenemos Feedback que nos va a ayudar a mejorar el PB si es necesario.

En la gestión ágil de R se trabaja a nivel de
- R de negocio
- R de usuario

Las US son R de usuario alineados a R de negocio

En este contexto, se aplican algunos CONCEPTOS RELACIONADOS CON LOS PRINCIPIOS ÁGILES

- LOS CAMBIOS VAN A EXISTIR TODO EL TIEMPO DE FORMA CONSTANTE --> sabemos que van a haber cambios, por lo que se busca gestionar los requerimientos orientados a poder aceptar el cambio. Si el PO, los clientes o los stakeholders no plantean cambios es porque el producto no está siendo utilizado.
- DEBEMOS TENER UNA MIRADA DE TODOS LOS QUE ESTÁN INVOLUCRADOS --> todos los que tienen algo para decir del producto (stakeholders)
- EL USUARIO DICE LO QUE QUIERE CUANDO RECIBE LO QUE PIDIÓ --> es por esto que las iteraciones cortas y las entregas continuas son esenciales, ya que permiten una retroalimentación y un enriquecimiento para futuras entregas.
- NO HAY TÉCNICAS NI HERRAMIENTAS QUE SIRVAN PARA TODOS LOS CASOS --> a veces se necesita trabajar con más prototipos o modelos, otras veces con técnicas sencillas es suficiente.
- LO IMPORTANTE NO ES ENTREGAR UNA SALIDA, UN REQUERIMIENTO, SINO ENTREGAR UN RESULTADO, UNA SOLUCIÓN DE VALOR --> apostamos a entregar algo útil para el negocio o el cliente minimizando el desperdicio.

PRINCIPIOS ÁGILES RELACIONADOS A LOS REQUERIMIENTOS ÁGILES

1. La prioridad es satisfacer al cliente a través de raleases tempranos y frecuentes (2 semanas a 1 mes)
2. Recibir cambios de R, aún en etapas finales
4. Técnicos y no técnicos (PO) trabajando juntos todo el proyecto: el PO es parte del equipo, sino difícilmente se pueda identificar qué es lo que da valor al negocio

(solo se enuncian los más relacionados, pero se pueden relacionar todos los principios con lo que tiene

ANIZADOS

es decir las mejores arquitecturas, diseños y R,
borganizados con libertad de tomar decisiones y
nera colaborativa. Esto de la emergencia va en
s complejas al principio, como R upfront.
a la importancia de permitir que los equipos de
grado de autonomía y capacidad de decisión en el
arrollo en lugar de tener un enfoque jerárquico y

ente con el enfoque tradicional y expresa que los
aracterísticas -estimar, cuanto tiempo se necesita,
esarrollarlo.

12. MEJORA CONTINUA

El equipo reflexiona sobre como ser más efectivo para luego justar y perfeccionar su comportamiento en secuencia durante ciertos intervalos de tiempo

PRINCIPIOS ÁGILES RELACIONADOS A LOS REQUERIMIENTOS ÁGILES

1. La prioridad es satisfacer al cliente a través de raleases tempranos y frecuentes (2 semanas a 1 mes)

2. Recibir cambios de R, aún en etapas finales

4. Técnicos y no técnicos (PO) trabajando juntos todo el proyecto: el PO es parte del equipo, sino difícilmente se pueda identificar qué es lo que da valor al negocio

6. El medio de comunicación por excelencia es cara a cara

11. Las mejores arquitecturas, diseños y R emergen de equipos autoorganizados.

(solo se enuncian los más relacionados, pero se pueden relacionar todos los principios con lo que tiene que ver con R ágiles)

USER STORIES

Una de las partes más difíciles de construir software es identificar los R. En primer lugar, sabemos que los R funcionales son difíciles de especificar cuando el usuario no logra definirlos, pero también está la dificultad de definir RNF.
Los errores cometidos en la etapa de definición de R son más difíciles de corregir ya que solo nos damos cuenta de que hay un error cuando el usuario tiene el sistema en frente, y por ende el problema se arrastra a lo largo de toda la vida del proceso.

Son una técnica para capturar R, se llaman así porque es una herramienta o técnica que la manera de utilizarlas es como si contáramos una historia.

No es tan importante lo que yo escribo como parte de la historia, sino todo lo que se habla acerca de esa historia.

Las US no son especificaciones detalladas de R (como un CU), sino que expresan la intención de hacer algo. En general no tienen demasiado detalle al principio del proyecto y son ¿elaboradas evitando especificaciones anticipadas sobre demoras en el desarrollo, inventario de R y una definición limitada de la solución?

Necesitan poco o nulo mantenimiento y pueden descartarse después de la implementación. Junto con el código, sirven de entrada a la documentación que se desarrolla incrementalmente después .

LAS 3 "C" DE UNA US

CONVERSATION

Es la parte más importante y no queda escrita en ningún lugar. Es la parte no visible. Es el momento en el que uno dialoga con el PO, o quien conoce esta interfaz en el equipo de desarrollo, para captar las necesidades del problema.

- En la conversación se obtienen todos los detalles para que el equipo pueda trabajar esa historia.

CONFIRMATION

Condiciones que se tienen que dar para dar por satisfecha la US. Esta dentro de la Card.

CARD

Es la parte visible de la US. En donde escribimos o expresamos algo que nos indique cual es la US, de que se trata y su valor de negocio.