El futuro digital es de todos

MinTIC

Hechos QUE CONECTAN

CICLO 3

EJE TEMÁTICO 1

INTRODUCCIÓN A LA INGENIERÍA DE SOFTWARE

Universidad Industrial de Santander

MisiónTIC2022

Introducción a la ingeniería de software

Ruta de aprendizaje

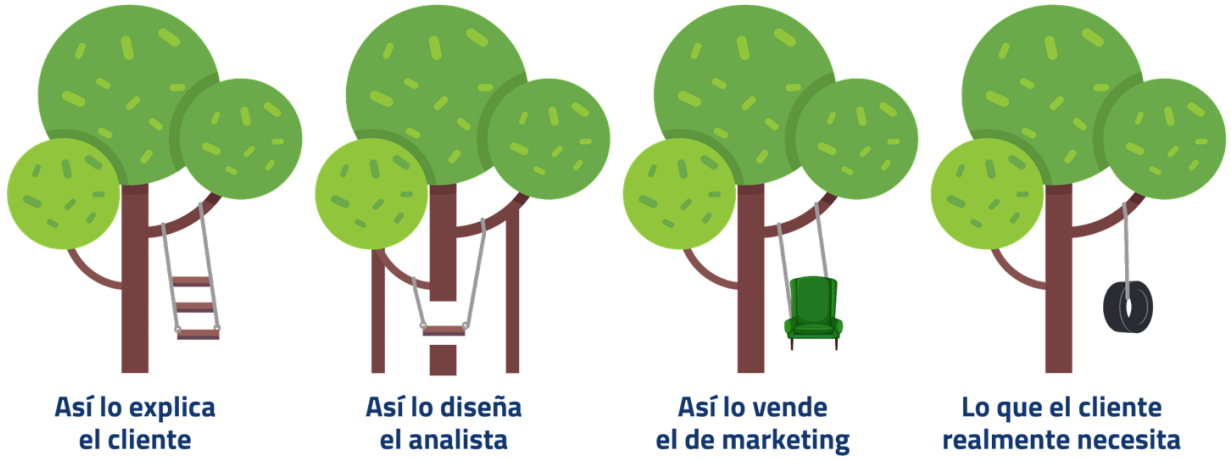
Introducción a la ingeniería de software

Este recurso pretende dar las bases para entender el ciclo de vida del desarrollo de software y, la metodología ágil Scrum junto con su implementación.

Palabras clave: Scrum, Ciclos iterativos, Product backlog, SDLC.

Introducción a la ingeniería de software

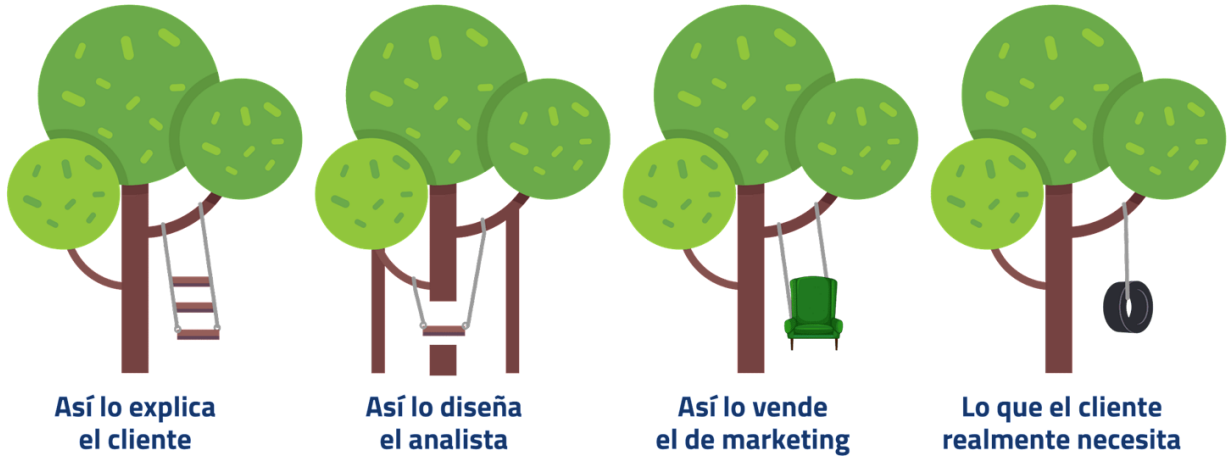
La ingeniería por definición, es: "el estudio de principios científicos para diseñar y construir máquinas, estructuras, entre otras cosas… " (https://dictionary.cambridge.org/dictionary/english/engineering), esto nos enseña que la ingeniería es un proceso creativo, por lo tanto, es variable, en pocas palabras, existen muchas maneras de diseñar y construir.



Introducción a la ingeniería de software

Existen diversas formas de hacer ingeniería y por supuesto, de hacer software, esto ha provocado que a lo largo del tiempo se hayan definido algunas guías, estándares y procesos con el objetivo de que, todas las partes involucradas estén satisfechas y que los resultados perduren en el tiempo.

Por lo tanto, dentro de esta parametrización se definen una serie de fases que, en conjunto, denotan el llamado "**Ciclo de vida del software**".



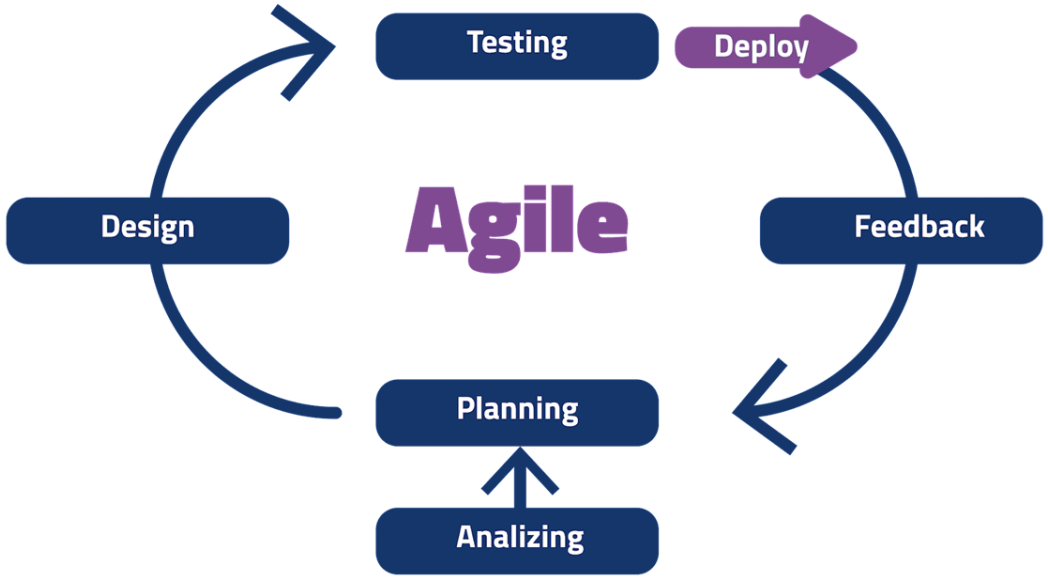
Ciclo de vida del software (SDLC)

Un ciclo de vida de software, busca definir las diferentes etapas por las que puede pasar un proyecto de software. Estas etapas se categorizan de la siguiente manera:

- **Procesos primarios**: análisis, diseño, desarrollo, operación y mantenimiento.
- **Procesos de soporte:** gestión de la configuración, aseguramiento de calidad, verificación y validación.
- **Procesos organizacionales:** capacitación, análisis de medición de proceso, gestión de infraestructura, reutilización y mejora de procesos organizacionales.
- **Procesos de proyectos cruzados**: reutilización y línea de productos de software.

Modelos de desarrollo de software

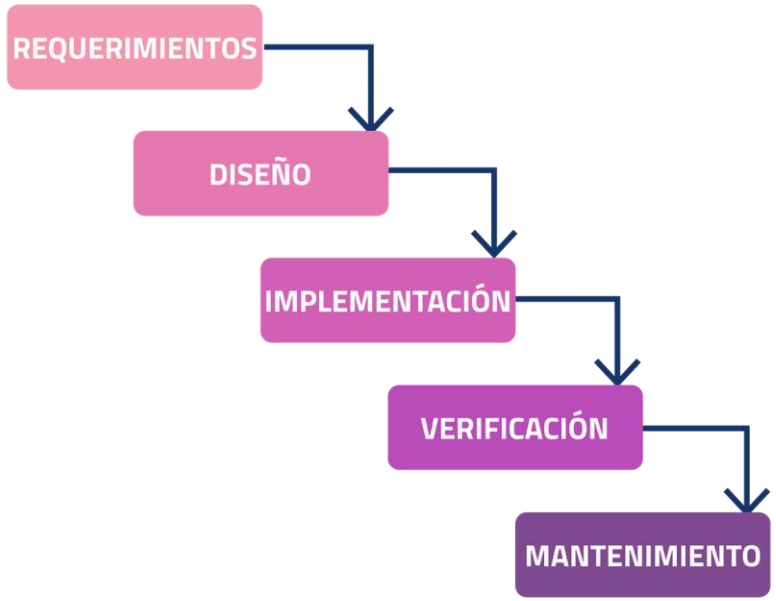
Frente a esa necesidad de parametrización y con el fin de evitar la incertidumbre a la hora de trabajar en proyectos de software, se plantean algunas estrategias que buscan ordenar y definir una estructura a esos procesos del ciclo de vida del software. Estas estrategias reciben el nombre de **Modelos de desarrollo de software,** y algunos de estos modelos son:



Modelos de desarrollo de software

- **Modelos lineales (modelo predictivo)**: las fases del desarrollo de software, se ubican secuencialmente para lograr una entrega de producto única.

- **Modelos iterativos (modelo adaptativo):** las fases del desarrollo de software, se logran iterativamente para lograr varias entregas de producto y feedback por parte del cliente.

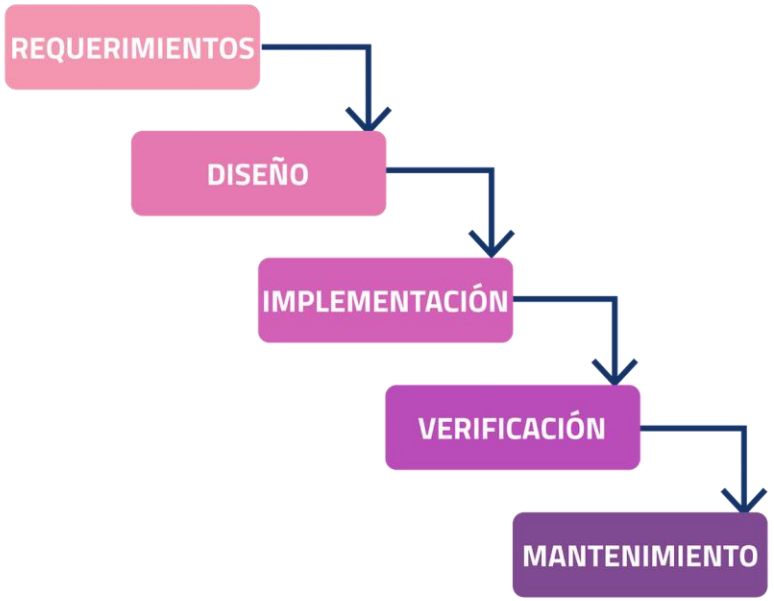


Disclaimer

No olvides que a final de cuentas estamos hablando de modelos, por lo que no constituyen una camisa de fuerza en su implementación y, por lo que es normal encontrar que algunas personas o compañías hayan realizado algunos ajustes y adaptaciones para garantizar que la operación de los mismos se ajuste a sus necesidades particulares.

Modelos básicos de ciclo de cura del software

Ciclo en cascada



Como se puede observar en la imagen, el ciclo en cascada corresponde a un modelo lineal en el que se organizan los procesos primarios uno tras de otro, es decir, se comienza por una fase de análisis para que hasta cuando ésta se dé por terminada, se procesa a una fase de especificación de requerimientos y una vez se culmine esta fase, se proceda al diseño y así sucesivamente.

Este modelo propone un acuerdo con la parte interesada al inicio del proyecto, este acuerdo estará compuesto de presupuesto, tiempo y recursos, con los que se buscará cumplir y entregar al cliente en la última fase, es decir, al finalizar la fase de implementación o despliegue.

Modelos básicos de ciclo de cura del software

Agile y ciclos iterativos

Partiendo del Manifiesto Ágil, se propone una filosofía y cultura materializada en los modelos ágiles, los cuales son iterativos, incrementales y evolutivos. Para esto se organizan los procesos primarios en iteraciones pequeñas y adaptativas, logrando así, entregas constantes al cliente y adaptando el proyecto a las necesidades del mismo.

Dentro de Agile se definen un conjunto de estrategias más específicas, estas son conocidas como las **metodologías ágiles**.

SCRUM

Partiendo del manifiesto ágil, SCRUM es un marco de trabajo en el que se definen elementos como la duración ideal de las iteraciones, el tamaño de los equipos de trabajo, las diferentes reuniones que se deben llevar a cabo, los participantes de las mismas, los involucrados, entre otros.

La metodología SCRUM se recomienda para proyectos de alta complejidad, proyectos que manejan incertidumbre debido a los entornos cambiantes y también porque son entornos en los cuales intervienen todos los integrantes de un equipo.

En esta metodología todo el equipo de trabajo estará enfocado en generar valor a la parte interesada de forma continua. Ese valor podríamos medirlo como ¿qué tanto uso podrá darle el cliente a nuestro producto una vez esté terminada una iteración?, pues en esta medida, tan pronto como la parte interesada pueda hacer uso del proyecto, así mismo podrá encontrar retorno a su inversión.

SCRUM entiende que en el proceso están involucrados **seres humanos,** por lo tanto, presenta un enfoque humanista con el objetivo de optimizar cada proceso. Dentro de este enfoque, la metodología plantea una serie de valores y pilares, los cuales serán referidos a continuación.

Valores

- **Compromiso**: Scrum reconoce que frente a la presencia del constante cambio, es necesario que todas las partes estén comprometidas, puesto que así se pueden hallar y aplicar los ajustes necesarios, y se podrán ejecutar de forma fluida y eficaz.

- **Coraje:** Si se va a trabajar en un entorno de incertidumbre, es necesario que el coraje esté presente en los involucrados en el proyecto, con coraje se podrá combatir la frustración que puede implicar el cambio constante.

- **Foco:** La metodología plantea la necesidad de concentración en un objetivo único, si bien, este objetivo cambiará y evolucionará con el tiempo, es importante que no se pierda de vista el producto al que queremos llegar con el paso del tiempo.

- **Apertura:** De nada servirían los valores mencionados previamente, si los equipos no estuviesen abiertos a ese cambio y evolución constante, con la apertura se comprenderá que el cambio constante y evolutivo es beneficioso para el proyecto, y para la satisfacción de las partes involucradas.

Pilares

- **Transparencia:** Frente al constante cambio y el hecho de que el cliente está 100% involucrado en el proyecto, la transparencia jugará un papel importante dejando muy claro el estado del proyecto en cualquier fase para evitar la frustración de alguna de las partes.

- **Inspección:** Scrum plantea que para poder mantener un proyecto evolutivo e iterativo, es necesaria la inspección constante, pues así todos tendrán conocimiento en cualquier momento del estado del desarrollo.

- **Adaptación:** Como ya se ha venido hablando, el cambio y la iteración es la columna vertebral de la metodología Scrum, sin embargo, esto solo será aplicable en la práctica si el equipo en su totalidad logra adaptarse a esto, de no ser así; se podría entrar en una fase en la que no avanzará el proyecto.

Roles

Evidentemente estos roles y pilares los aplicarán **personas,** por lo tanto, Scrum trabaja estableciendo un orden, pues define la necesidad de unos roles puntuales, los cuales ayudarán a que la metodología se ejecute de la mejor manera.

- **Stakeholder**: Serán las personas o la persona, que tendrá la necesidad del proyecto, en pocas palabras, será el cliente.

- **Product Owner:** Este rol lo ocupará una persona experta en el tema a desarrollar, idealmente será una persona con experiencia que pueda aportar de manera contundente con la mejor solución posible a los problemas de la parte interesada.

- **Scrum master:** La persona que ocupe este rol se convertirá en el puente entre los diferentes roles, también, se encargará de la motivación y de que todo marche bien, solucionará los impedimentos y trabajará en que todas las partes se encuentren satisfechas sobre el proyecto.

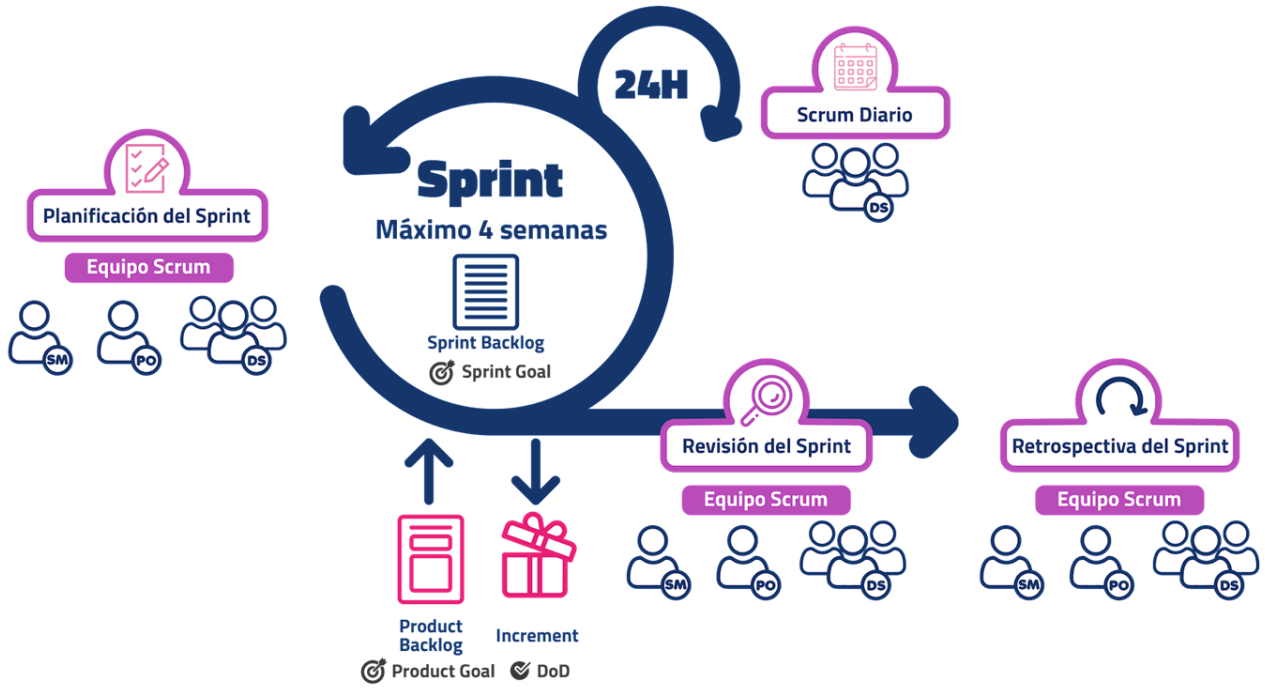
- **Team Scrum:** Estará conformado por el personal operativo y creativo. En este rol encontraremos el equipo de desarrolladores, diseñadores, testers y todas las personas necesarias para el desarrollo del software.

Disclaimer

Puede que en el camino encuentres compañías o proyectos que tienen roles adicionales o incluso, que hayan removido alguno de los roles base. Asimismo, pueden existir ceremonias (ya hablaremos de ellas) que no se apliquen o que, al contrario, se definan algunas que no están especificadas en la guía scrum.

Sprint

El concepto de *Sprint* es importante en Scrum, dado que este es el término con el que se define el espacio de tiempo en el que se estará trabajando el proyecto. Esta ventana de tiempo, idealmente puede ser de 15 a 30 días.



Ceremonias

Para hacer la metodología aplicable en la práctica se ha definido una serie de ceremonias (o reuniones), en donde se busca la aplicación de esos pilares y valores, y donde también cada rol tenga un papel definido.

- **Sprint planning**: Este espacio se dará al iniciar cada sprint, se buscará definir con el cliente el trabajo a realizar durante el siguiente sprint, buscando enfocarse siempre en la generación de valor para él.

- **Daily Scrum:** Será un momento diario en el que analizaremos y comunicaremos el estado del proyecto y sus avances.

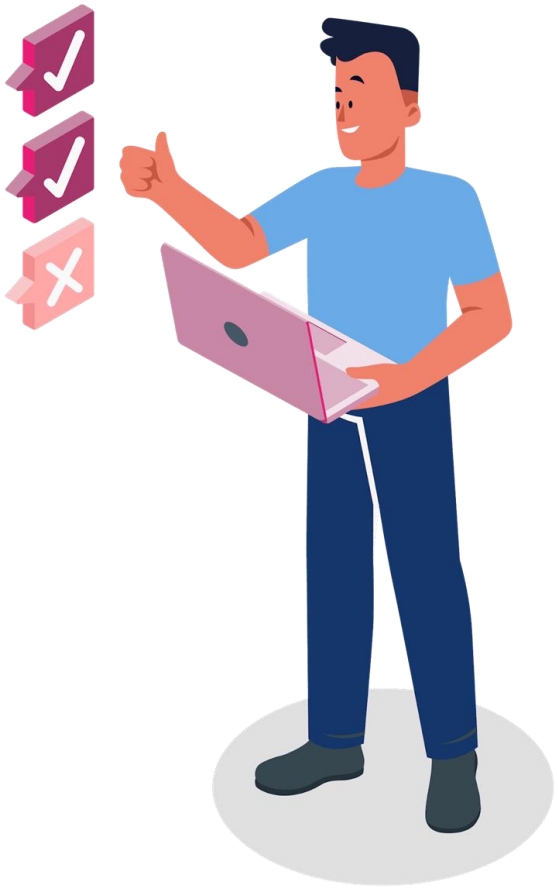
- **Sprint Review:** Una vez terminado el sprint, se definirá un espacio para mostrarle al cliente el producto, sus avances, en este espacio se recibirá el feedback del mismo para tenerlo presente en la siguiente iteración.

- **Sprint retrospective:** Partiendo del feedback proporcionado por el cliente, se analizará el desarrollo del sprint en tres puntos: ¿Qué salió bien?, ¿Qué salió mal? y ¿Cómo podemos mejorar? Esto con el objetivo de tener una mejora constante tanto del equipo como del producto.

Product backlog

Teniendo un contexto sobre la metodología Scrum, es necesario comprender el punto de partida de la misma y así como en el modelo Cascada, esta partía de un análisis de requerimientos para entender la serie de ítems a trabajar; en Scrum se parte de una lista de **necesidades** que se irán solucionando sprint a sprint, y esta lista recibe el nombre de Product Backlog.

La construcción exitosa de un product backlog, ayudará en gran parte a que se dé el flujo del desarrollo del proyecto, y para ello se pueden definir una serie de parámetros, los cuales buscarán garantizar la calidad del mismo.



DEEP

Existen diferentes estrategias para garantizar un product backlog de calidad, algunas guías son la SMART y DEEP, las cuales por sus siglas denominan una serie de puntos a considerar a la hora de escribir objetivos, o en este caso, ítems del product backlog. En este caso en específico nos enfocaremos en la guía DEEP:

- **D**etailed appropriately: define que cada item de product backlog deberá estar tan detallado como tan prioritario sea. Es decir, si un ítem tiene mayor prioridad, debe estar más detallado y si no es tan urgente, no es necesario que sea muy específico.
- **E**mergent: cada ítem podrá cambiar en el tiempo, se podrá adaptar al cambio y evolucionar con el tiempo.
- **E**stimated: preferiblemente se deberá definir una estimación en tiempo/recursos/esfuerzo para cada ítem, de tal modo que sea claro el costo de desarrollo que conlleva.
- **P**rioritized: se tendrá claro cuál ítem del product backlog es más importante, o tiene más urgencia en ser desarrollado. Es necesario tener siempre presente que su desarrollo deberá generar valor a la parte interesada.

Product backlog items (PBI's)

Ahora, a diferencia de un requerimiento, el product backlog no es solo una lista de tareas o un esquema de elementos pendientes. Un product backlog puede estar compuesto por diferentes elementos que toman el nombre de product backlog items o PBI's. Estos elementos se describen de la siguiente manera:

- **Épica**: Se definirá como épica aquella funcionalidad general a implementar. Por ejemplo: Implementar un sistema de usuarios.
- **Historia de usuario**: Desde la necesidad de la parte interesada, se definirán las historias de usuario que componen una épica. Por ejemplo: Como usuario del aplicativo, quiero poder acceder con usuario y contraseña para garantizar que las transacciones quedarán asociadas a mi perfil.
- **Tarea:** Una vez definida la historia de usuario, esta se desglosará en tareas puntuales. Por ejemplo: Implementar un sistema de inicio de sesión bajo el protocolo OAuth 2.0 en comunicación con la API de Google.
- ***Spike**:* Si bien, este ítem no es como tal parte de la guía de Scrum, se podrá adaptar para cuando dentro del desarrollo del sprint sea necesario hacer una investigación. Por ejemplo: Consultar con el equipo jurídico si se podría hacer uso de una librería para la implementación del sistema de usuarios.

Historias de usuario (User Stories)

Entender las necesidades de la parte interesada nos dará una visión sobre el producto que él espera. Para ello se definen las historias de usuario, las cuales nos permitirán enfocarnos en ese dolor que tiene el cliente y nos dejará la libertad de buscar una solución.

Existe una plantilla bastante sencilla para construir este tipo PBI, sin embargo, puede que en el camino se utilice alguna otra que brinde más claridad. Esta plantilla parte de tres verbos:

Como…Quiero…Para…

Y, por ejemplo, podría utilizarse de la siguiente forma:

*"**Como** formador **quiero** llevar un control de asistencia automatizado **para** evitar el registro manual de los estudiantes"*

Software de gestión de proyectos

Una vez definidas las necesidades de la parte interesada utilizando las herramientas del product backlog probablemente busquemos la forma de hacer un seguimiento a este trabajo, esfuerzo, estimaciones y demás. Para ello existen infinidad de plataformas que ayudan a gestionar un proyecto, incluso, unas más específicas a la hora de gestionar proyectos de software.

Revisión de lo aprendido

1. ¿Cuáles son las siglas en inglés para el término "Ciclo de vida del software"?

   a. CVS
   b. SDLC
   c. SCRUM
   d. DEEP

Revisión de lo aprendido

2. Indique si es falso o verdadero la siguiente afirmación. Existe una única manera de llevar a cabo un proyecto de software:

a. Falso

b. Verdadero

Revisión de lo aprendido

3. Indique si es falso o verdadero la siguiente afirmación. Las diferentes metodologías y modelos de desarrollo de software son inmutables, las modificaciones a ellas provocan el fracaso del proyecto:

a. Falso

b. Verdadero

Revisión de lo aprendido

4. Scrum está basado en un movimiento cultural y filosófico llamado:

a. Neoliberal
b. Capitalismo
c. Manifesto ágil
d. Desarrollo de software eficaz

Revisión de lo aprendido

5. Complete las siguientes palabras basado en el enunciado.
¿Qué significa cada sigla del acrónimo DEEP?

D	Detailed appropriately.
E	Emergent.
E	Estimated.
P	Prioritized.

El futuro digital es de todos

MinTIC

Hechos QUE CONECTAN

CICLO 3

EJE TEMÁTICO 1

INTRODUCCIÓN A LA INGENIERÍA DE SOFTWARE

Universidad Industrial de Santander

UIS

MisiónTIC 2022