Desarrollo de Software I

Facultad de Ingeniería
Escuela de Ingeniería de Sistemas y Computación

Mauricio Gaona
mauricio.gaona@correounivalle.edu.co
Profesor

2023-I

Objetivos del curso

Objetivo general

Proporcionar al estudiante las bases conceptuales fundamentales de la ingeniería de software y los elementos metodológicos necesarios para llevar a cabo el desarrollo de aplicaciones de software.

Objetivos específicos

- Entender los **conceptos fundamentales** de la ingeniería de software.
- Entender y usar **metodologías ágiles** de desarrollo de software para ser un miembro efectivo y eficiente en un equipo de desarrollo ágil.
- Entender los conceptos principales de las **metodologías tradicionales** de desarrollo de software.
- Entender y aplicar los **roles** y las actividades que se realizan en las etapas de **análisis, diseño, codificación, pruebas y despliegue** de una aplicación de software.

Objetivos del curso

Objetivos específicos

- Identificar **buenas prácticas** aplicables a cada momento de un proceso de desarrollo de aplicaciones web con el fin de facilitar la creación colectiva de software.
- Aplicar algunos modelos, herramientas, lenguajes y plataformas, para implementar una aplicación de software usando **tecnologías de tendencia**.
- Entender como **estimar** un producto de software (tiempo y valor en $$$ de una aplicación de software)
- **Desarrollar una aplicación de software**.

Desarrollo del curso

Metodología
Clase presenciales preparadas por el profesor y con participación de los estudiantes

Evaluación (Por competencias)
Examen, Quices, Tareas, Exposiciones, Proyecto, informes escritos y comunicación oral

Examen	30
Quices	10
Tareas	10
Exposiciones	10
Proyecto	35
Informes escritos	2.5
Comunicación oral	2.5

Materiales del curso
Campus Virtual y Enlaces web

Contenido
Temas a tratar en el curso

Bibliografía
Libros y Artículos recomendados

Desarrollo de Software I

Desarrollo del curso: Evaluación

Evaluación por competencias

La **evaluación de competencias** es el proceso de recopilación de evidencias que muestran los resultados del aprendizaje del estudiante

Resultados de aprendizaje (RA)

Es lo que se espera que un estudiante conozca, comprenda y/o sea capaz de hacer al final del curso.

RA1 = 40%
RA2 = 55%
RA3 = 5%

R.A.1: Comprende los conceptos fundamentales que rigen la ingeniería de software e identifica y caracteriza las ventajas y desventajas de las metodologías de desarrollo de software.

R.A.2: Especifica los requerimientos del sistema considerando las necesidades del cliente con el fin de diseñar, Implementar, validar y desplegar un sistema de acuerdo con el diseño aprobado usando prácticas ágiles, con el fin de presentar el sistema desarrollado.

R.A.3: Aplica diferentes formas y herramientas de comunicación usando informes escritos y presentaciones orales para expresar sus ideas al solucionar problemas.

Exámenes	30.03	100%
Quices	10.03	
Tareas	9.86	
Exposiciones	10.02	
Proyecto	35.06	
Informes escritos	2.5	
Comunicación oral	2.5	

Desarrollo de Software I

Aspectos a tratar

Contenido

1	Conceptos Generales. El proceso de desarrollo de software: Ciclo de vida de desarrollo del software, modelos de ciclo de vida del software, conceptos sobre metodologías de desarrollo de software, otros modelos
2	El Proceso de Realización de un Producto de Software. - Metodologías Ágiles para el desarrollo de software (**Scrum**, XP, Kanban y otras) - Metodologías tradicionales (Artefactos de las metodologías tradicionales)
3	Análisis y especificación de requerimientos (funcionales y no funcionales) mediante el Product Backlog . Arquitecturas para el diseño de sistemas de software.
4	Planeación y desarrollo de un producto de software Usando prácticas ágiles: Prácticas de gestión y prácticas técnicas Iteración cero (Una idea, Visión del producto, necesidades del producto, equipo de trabajo, fechas críticas, Vistas de diseño del sistema, Tecnología a usar, Release plan, Priorización, Estimación y criterios de aceptación).
5	Aspectos conceptuales del world wide web (www) Protocolos web, anatomía de la URL El rol de los Servidores y Clientes web, peticiones Get y Post Front-end y Back-end

Aspectos a tratar

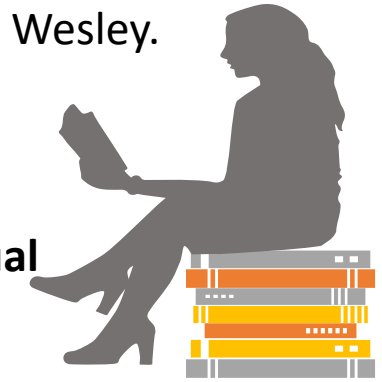
Contenido

5	Prácticas para el agilismo • Uso de prácticas ágiles como (reunión diaria, iteraciones cortas, planeación de la iteración, pruebas unitarias, incrementos de software (pequeñas entregas) y retrospectivas, entre otras.)
6	Implementación • Diseño de APIs • Mapeo de los diseños para codificación según la arquitectura definida
7	Pruebas de software • Conceptos sobre pruebas • Pruebas unitarias, pruebas funcionales y pruebas End-to-End • Construcción de pruebas para los diferentes componentes de un proyecto de software
8	Estimación de tiempos y costos de una aplicación de software
9	Despliegue de aplicaciones de software • Servidores físicos, maquinas virtuales, contenedores y plataformas • Despliegue local, despliegue en la nube
10	[Aplicaciones en la Nube (IaaS, PaaS, **SaaS**)]
11	[Tendencias emergentes en las aplicaciones web]

Bibliografía

- Concise Guide to Software Engineering: From Fundamentals to Application Methods, Gerard O´Regan, 2017.
- Agile Software Architecture: Aligning Agile Processes and Software Architectures, Muhammad Ali Babar, Alan W. Brown, Kai Koskimies, 2015.
- The Agile Samurai: How agile masters delivery great software. Jonathan Rasmusson, ISBN 13-978- 1-934356-58-6, 2017.
- User Story Mapping, Jeff Patton and Peter Economy, 2014.
- Ingeniería del Software (8a edición), Ian Sommerville, ISBN: 978-0-07-802212-8, Adisson Wesley, 2015.
- Adam Przybyłek and Miguel Ehécatl Morales-Trujillo. Advances in Agile and User-Centred Software Engineering: Third International Conference on Lean and Agile Software Development, LASD 2020
- Making Sense of Agile Project Management: Balancing Control and Agility, Charles G. Cobb, Jonh Wiley & Sons, ISBN: 978-0-470-94336-6 (2021)
- Booch G; Rumbaugh J; Jacobson I. The Unified Modeling Language, Reference Manual. Addison Wesley. 1996.
- Django for APIs: Build web APIs with Python & Django, Vincent William, 2018

- **Recursos online: Artículos y direcciones web (URL) que estarán disponibles en el campus virtual**
- Tutorial Django: https://www.youtube.com/watch?v=7XO1AzwkPPE



Desarrollo de Software I

Por que estudiar ingeniería de software ?

01 Ingeniería de software

APLICAR CONOCIMIENTO

La Ingeniería de Software aplica el conocimiento y la comprensión teórica obtenidos a través de la computación para la creación de aplicaciones de software de calidad.

02 Ingeniería de software

ALTA DEMANDA

Necesidad creciente de desarrolladores de software con talento. A medida que avanza la tecnología, se busca la capacidad de crear software de calidad teniendo en cuenta el diseño, el desarrollo, la seguridad y el mantenimiento entre otros.

03 Ingeniería de software

DISCIPLINA MADURA

Como una disciplina que madura, el software se vuelve cada vez más importante en nuestra vida cotidiana.

04 Ingeniería de software

RECONOCIMIENTO INTERNACIONAL

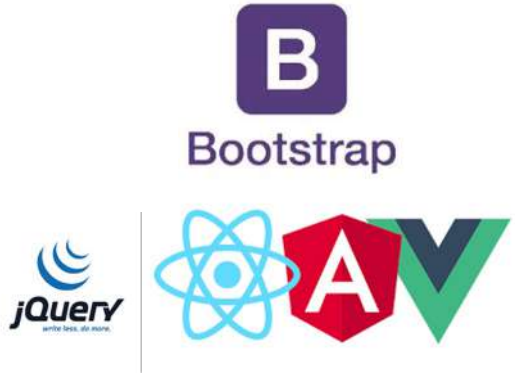
Es una profesión de clase mundial.

Herramientas

Algunas herramientas requeridas

Ambiente de trabajo: AV o MV

Herramientas de desarrollo : **Front end**
HTML5
JavaScript y CSS3



Back-end

Python
Django

Servidor Web: Apache, nginx

Bases de datos: PostgreSQL, Oracle

Herramientas de Gestión ([Google Docs, Jira, Taiga])
Manejador de versiones : GitHub , Bitbucket

IDE

https://aws.amazon.com/es/education/awseducate/

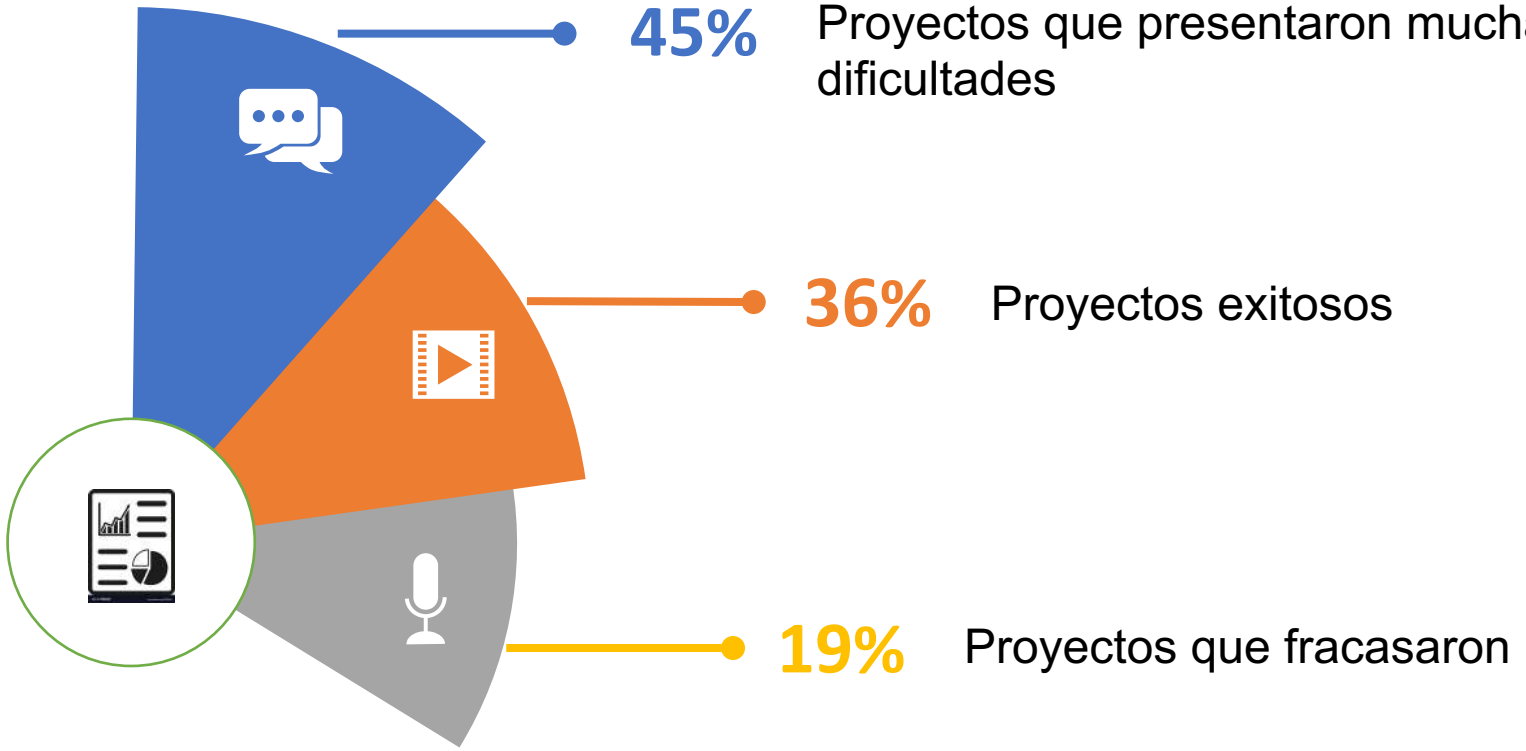
https://education.github.com/

https://azure.microsoft.com/es-es/free/students/

Estado actual del desarrollo de proyectos de software

Conceptos: **Crisis del software**

Reporte (2020) Caos Report Standish Group (encuesta a nivel mundial a empresas que hacen software)



Crisis del software: Problemas de calidad, Entregas tardías y Sobrecostos

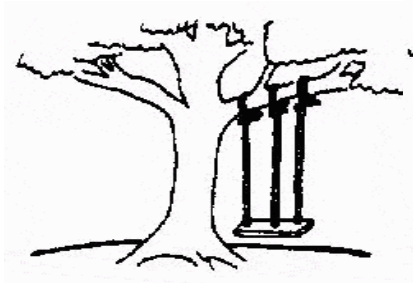
Conceptos

Por qué las estadísticas son tan malas ?

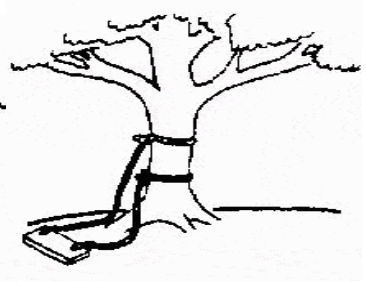
- Entendimiento equivocado del desarrollo de software.
- Mitos del software.
- Falsas asunciones.
- No distinguir entre hacer un programa de computador y el desarrollo de un producto de software.
- Los programas de software crecen exponencial mente en complejidad y nivel de dificultad con respecto al tamaño.
- Los desarrollos sin metodologías no funcionan.
- Uso inadecuado de las metodologías de desarrollo de software.
- Complejidad interna.

Problemas de comunicación

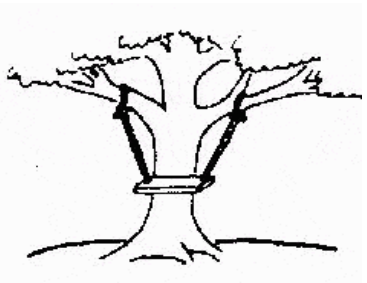
Asi fueron definidos los requerimientos



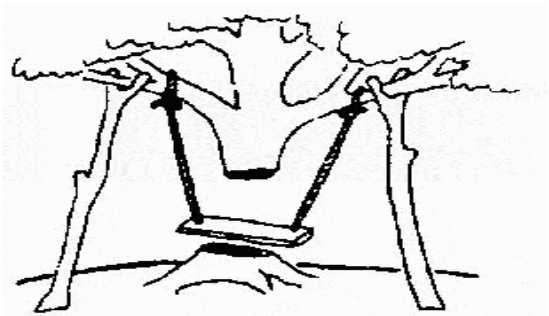
Asi lo entendieron los desarrolladores



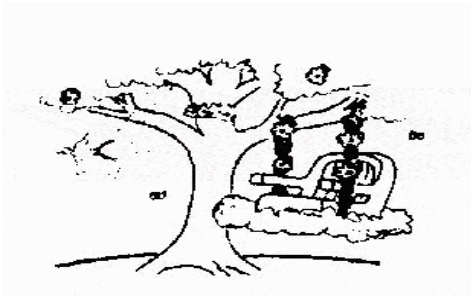
Asi fué resuelto el problema anteriormente



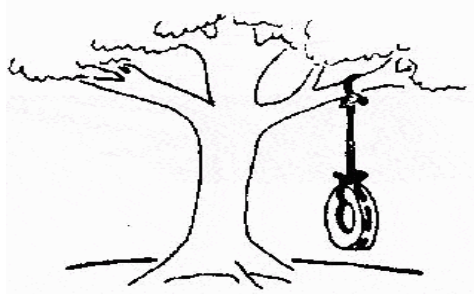
Asi fué resuelto el problema ahora



Este es el problema despues de la depuración



Así es como el problema es descrito por ventas



Esto es lo que el cliente realmente quería.

Conceptos

Mitos del software

(Desde la perspectiva del usuario)

– Una descripción general de los objetivos es suficiente para iniciar el desarrollo del software. Detalles de los requerimientos pueden ser incorporados después.

– Los requerimientos del software son estables, y el software tiene que ser echo tan flexible que permita incorporar cambios en medida que se requiera.

Conceptos

Mitos del software

(Desde la perspectiva del Desarrollador)

- Una vez el software es demostrado, el trabajo esta terminado.
- Hasta que el software no está codificado no es posible determinar su calidad.
- Lo único que se entrega del software es el código probado.

Conceptos

Mitos del software

(Desde la perspectiva del Administrador)

- Si estamos certificados por un estándar no hay que preocuparse por los procedimientos.
- Si tenemos el hardware(computadores) mas sofisticados y las herramientas estado del arte todo saldrá bien.
- Si estoy retrasado en un proyecto contratando más desarrolladores y con jornadas mas largas me pondré al día.

Conceptos

Asunciones mal interpretadas

- Todos los requerimientos pueden ser pre-especificados.
- Los usuarios son expertos para especificar sus necesidades.
- Usuarios y desarrolladores son buenos para definir las interfaces.
- El equipo de desarrollo son capaces de comunicarse sin problemas.

Conceptos

Programación Vs Desarrollo de software

La programación es el proceso de trasladar un problema de su ambiente físico a un lenguaje que el computador pueda entender y obedecer.

- Uno o dos desarrolladores
- Aplicaciones pequeñas
- Vida corta
- Uno o pocos participantes
- Normalmente construido desde cero
- Mantenimiento mínimo

Conceptos

Programación Vs Ingeniería de software

- Equipos de desarrolladores con múltiples roles.
- Sistemas complejos.
- Infinito crecimiento.
- Muchos participantes.
- Familias de sistemas.
- Reutilización para reducir costos.
- Requieren mantenimiento.

Conceptos

¿Qué es software ?

- I — Instrucciones (programas de computador) que cuando se ejecutan proveen la funcionalidad y rendimiento requerido.
- D — Documentos que describen la operación y uso de los programas.
- E — Estructuras de datos que permiten a los programas manipular la información adecuadamente.

Desarrollo de Software I

Conceptos

Conceptos

Qué es Ingeniería de Software ?

Los libros tienen diferentes definiciones:

- La ingeniería de software está relacionada con las teorías, métodos y herramientas de desarrollo, mantenimiento y evolución de los productos de software. (Sommerville).

- Es una disciplina donde la idea es la producción de software de calidad, entregable a tiempo, dentro de un presupuesto definido y satisfaciendo las necesidad del usuario (Stephen R. Schach).

- The practical application of scientific knowledge in the design and construction of computer programs and the associated documentation required to develop, operate and maintain them (B.W. Boehm)

Conceptos

En Resumen Ingeniería de Software es ...

- Alcance
  - –Estudio de los procesos de software, desarrollo, principios, técnicas y documentación.
- Objetivos
  - –Producción de software de calidad,
  - –Entregable a tiempo,
  - –Dentro de un presupuesto estimado,
  - –Satisfaciendo los requerimientos del cliente y las necesidades de los usuarios.

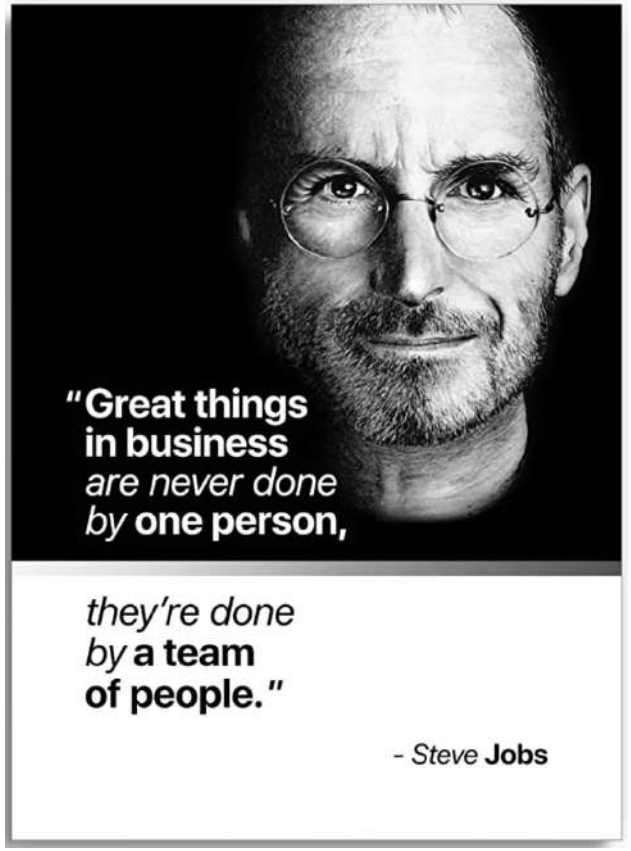
Conceptos

El desarrollo de software software hoy día

- Las organizaciones: “nos vamos con quien hemos trabajado en el pasado o lo que usa la mayoría”.
- Todo el mudo esta muy ocupado tratando de terminar sus productos para gastar tiempo en educación o entrenamiento para tratar los problemas efectivamente.
- “Terminar tarde las cosas” se está convirtiendo en una práctica institucionalizada y causa de muchos problemas.
- Pocas personas conocen o pueden integrar las mejores prácticas.
- Imposibilidad de adoptar y utilizar metodologías probadas de manera oportuna.
- A pesar de esto se han echo significativos avances en áreas específicas que rápidamente impactan la industria del software.

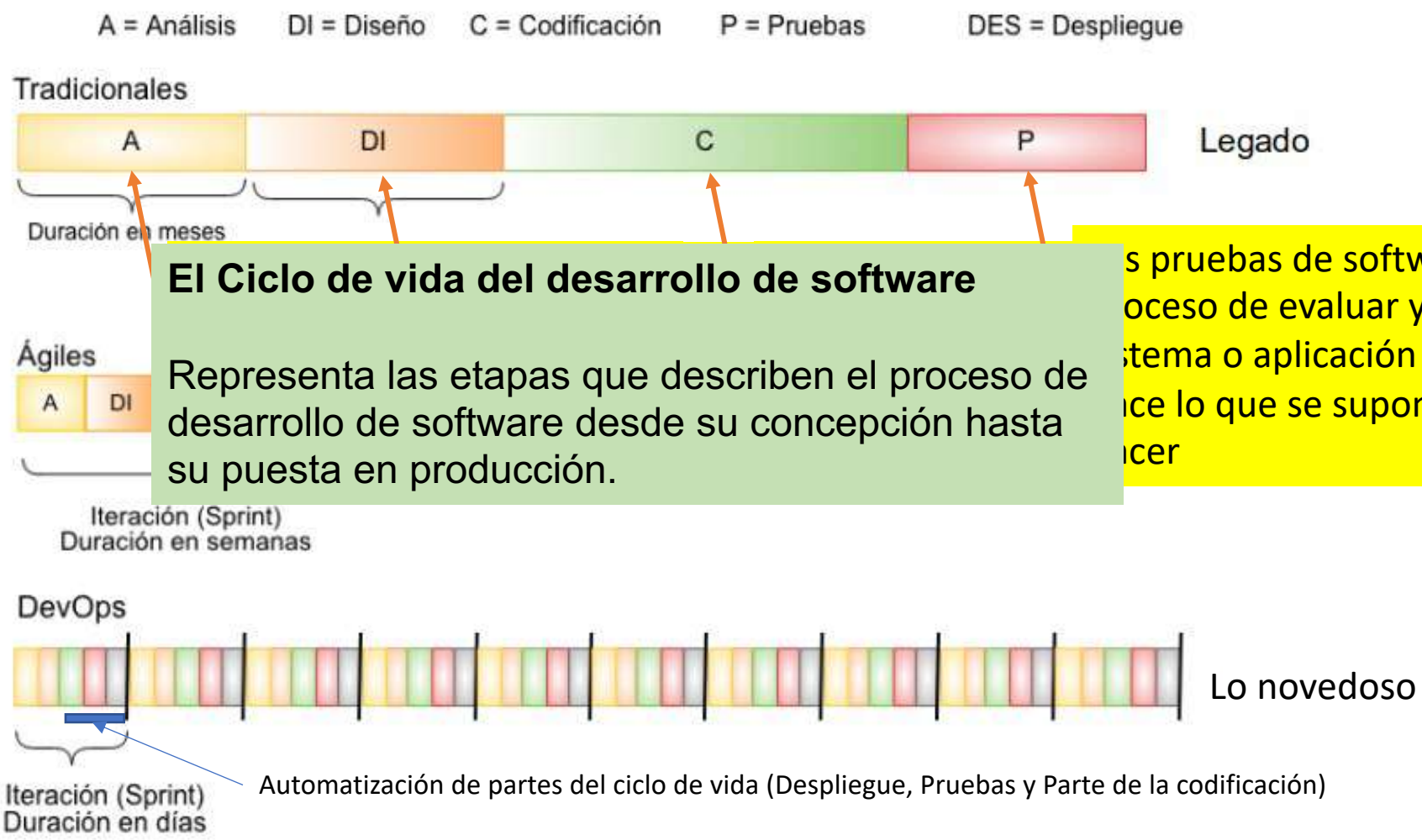
Conceptos

Conceptos Básicos



Conceptos

Ciclo de vida del desarrollo de software evoluciona



El Ciclo de vida del desarrollo de software

Representa las etapas que describen el proceso de desarrollo de software desde su concepción hasta su puesta en producción.

s pruebas de software es el
oceso de evaluar y verificar que un
tema o aplicación de software
ce lo que se supone que debe
cer

Desarrollo de Software I

Ciclo de vida del software hoy día

IDEACIÓN

Lluvia de ideas que resuelven un problema particular que enfrentan los usuarios.

REQUERIMIENTOS

Interacturar con los stakeholdes y usuarios para entender los procesos y recolectar los requerimientos del Proyecto.

DISEÑO

Crear la arquitectura de un sistema de software es decir sus componentes, sus interfaces y su forma de interactuar.

DESARROLLO

Crear el sistema de software usando una metodología de desarrollo y herramientas técnicas.

PRUEBAS

Evaluar la calidad del sistema para encontrar y corregir posibles errores.

DESPLIEGUE

Colocar el sistema en los servidores o ambiente donde va a funcionar.

MONITOREO

Conocer el estado del sistema y su infraestructura por si se requiere actualizaciones y soporte del software que esta en producción.

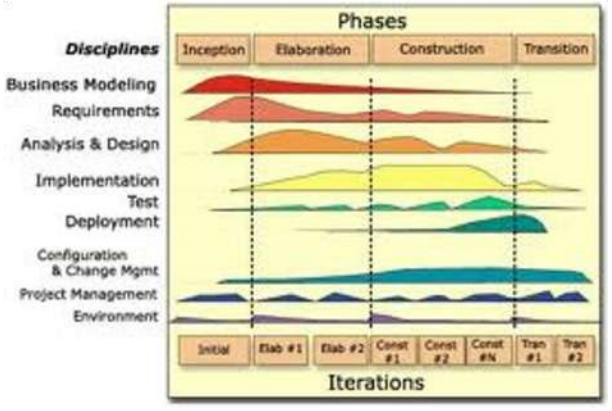
Conceptos

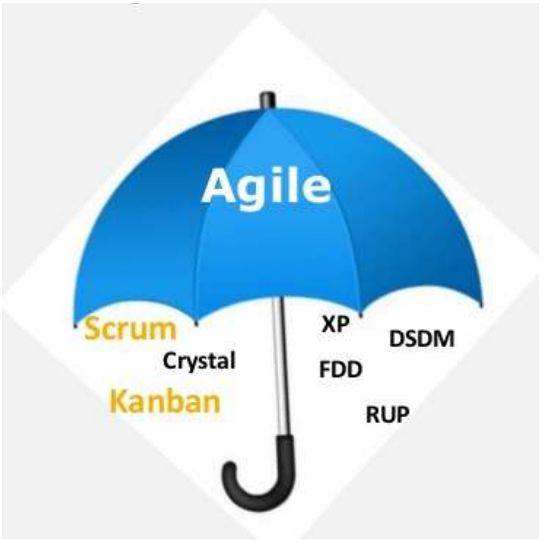
Metodologías

- Son una colección de métodos aplicados a lo largo del ciclo de vida de desarrollo de software, coherentes entre sí y que siguen una filosofía o enfoque de desarrollo de software.

- Es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo de un sistema de software.

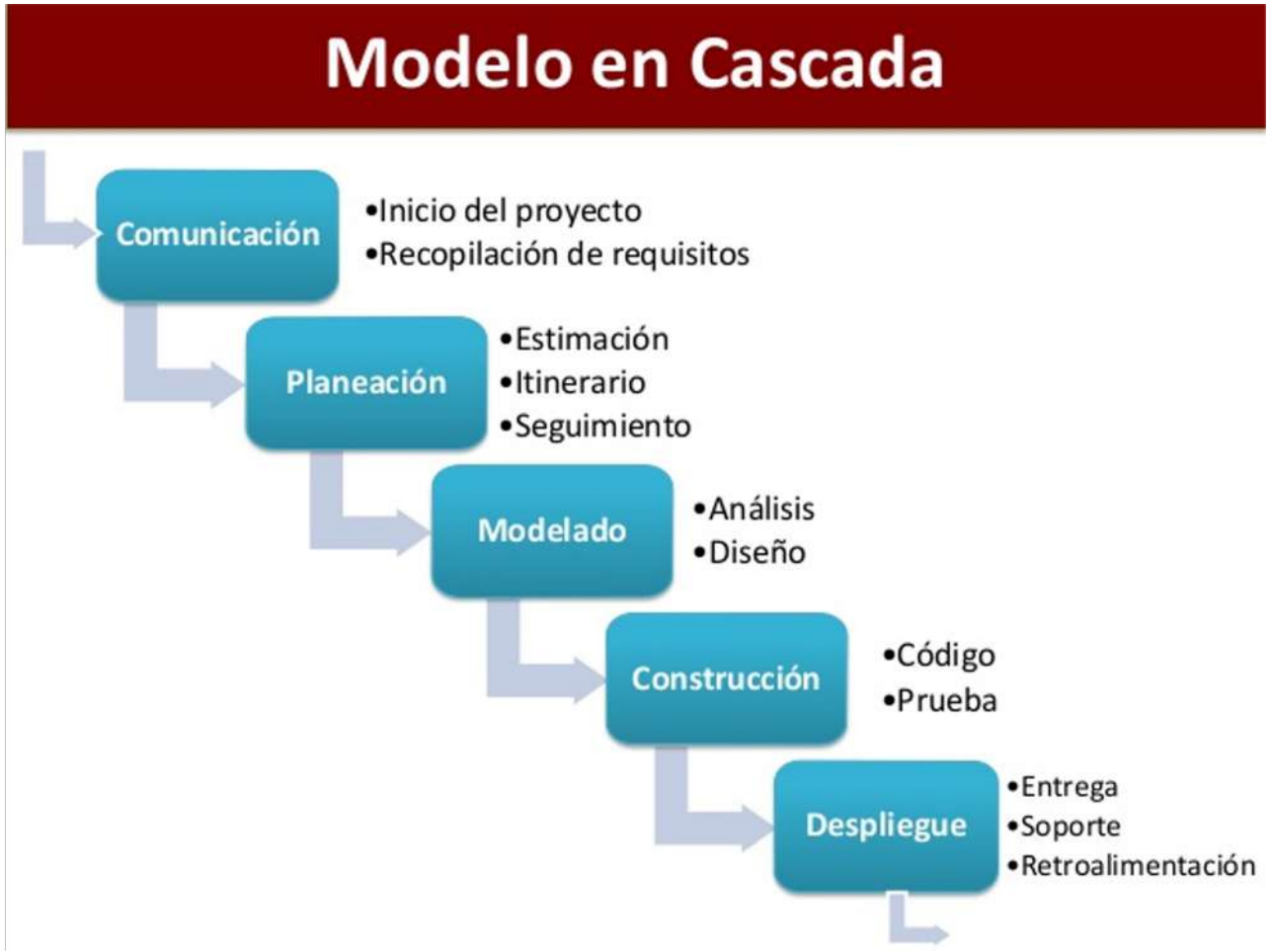
- Ejemplo: Metodologías Tradicionales RUP, MSF, ... Metodologías Ágiles SCRUM, XP, Kanban ...





Conceptos

Metodologías tradicionales



Desarrollo en cascada

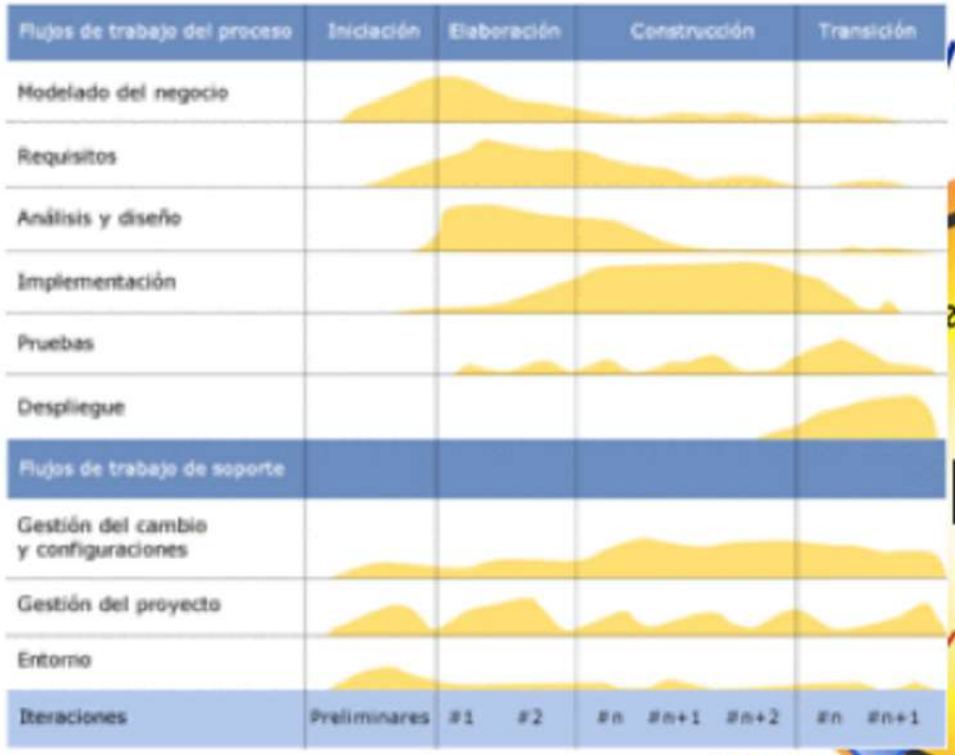
Dependía de una planeación muy grande al inicio de los proyectos y una captura muy detalla de los requerimientos.

- Requiere procesos bien definidos
- Dificultades para incluir cambios
- Todo se entrega al final del proyecto

Conceptos

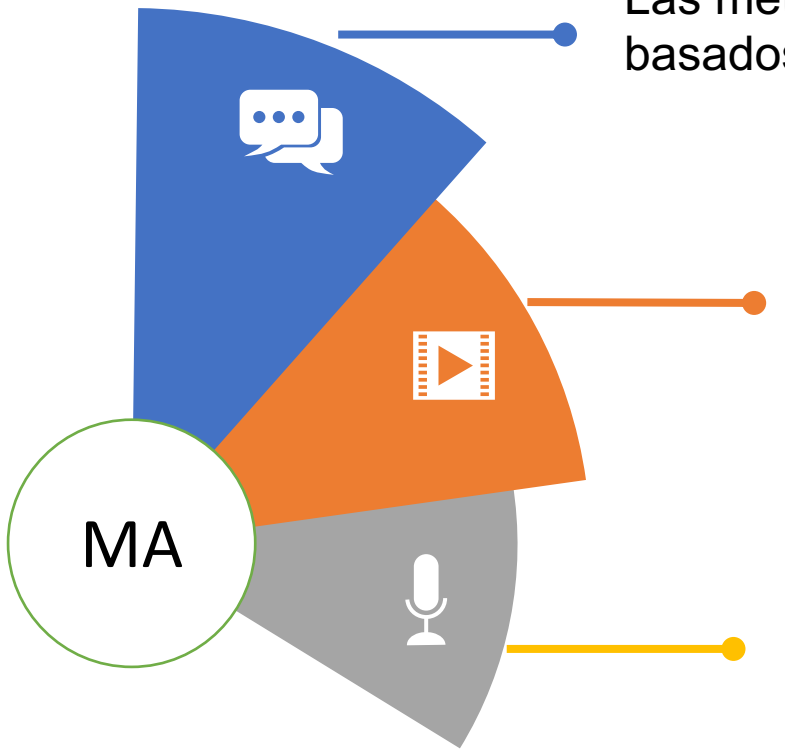
Metodologías tradicionales

RUP



Conceptos: Crisis del software

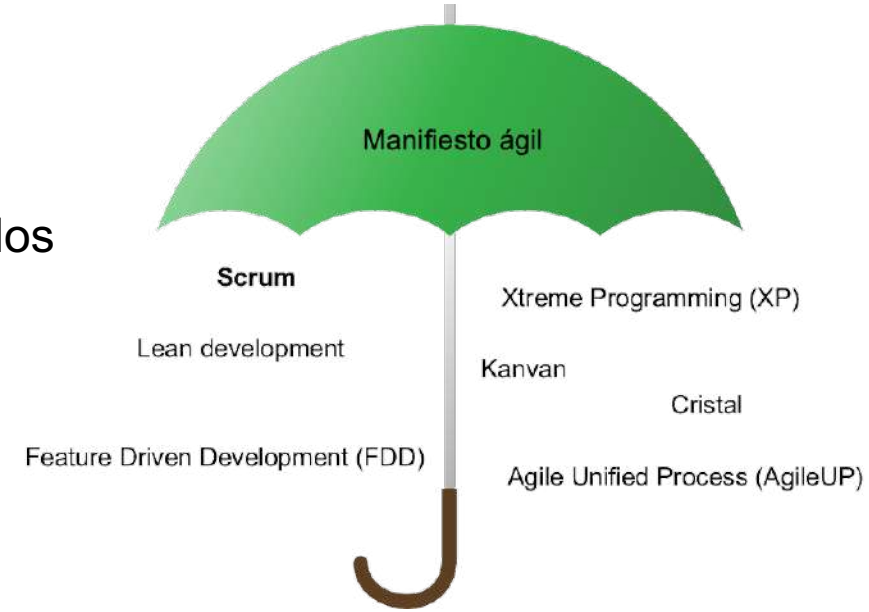
Metodologías ágiles



Las metodologías ágiles se refieren a métodos de ingeniería del software basados en el desarrollo iterativo e incremental.

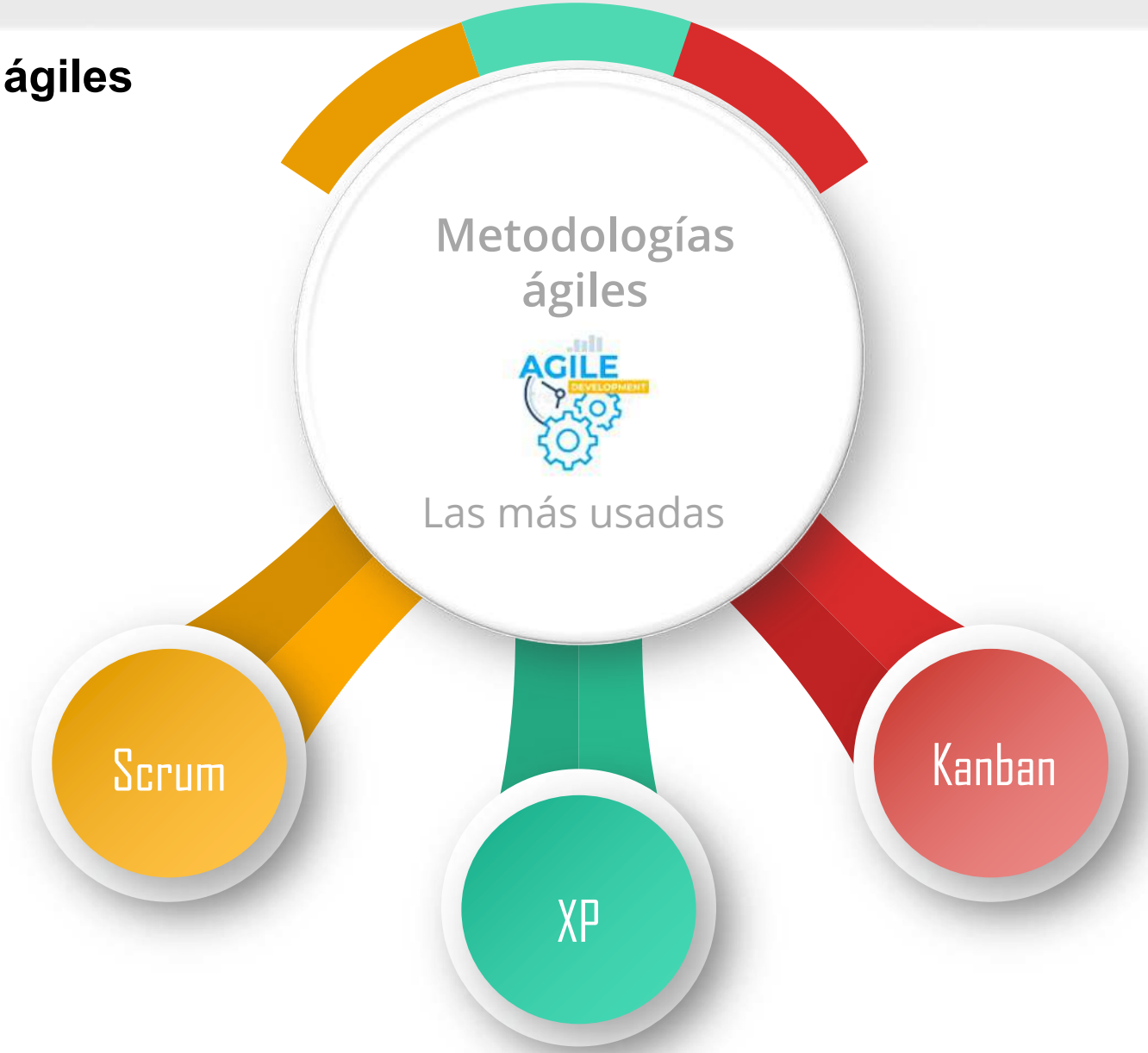
Las metodologías ágiles se basan en **prácticas ágiles** donde una práctica ágil es una actividad que se hace durante el proceso de desarrollo de software

Ejemplos



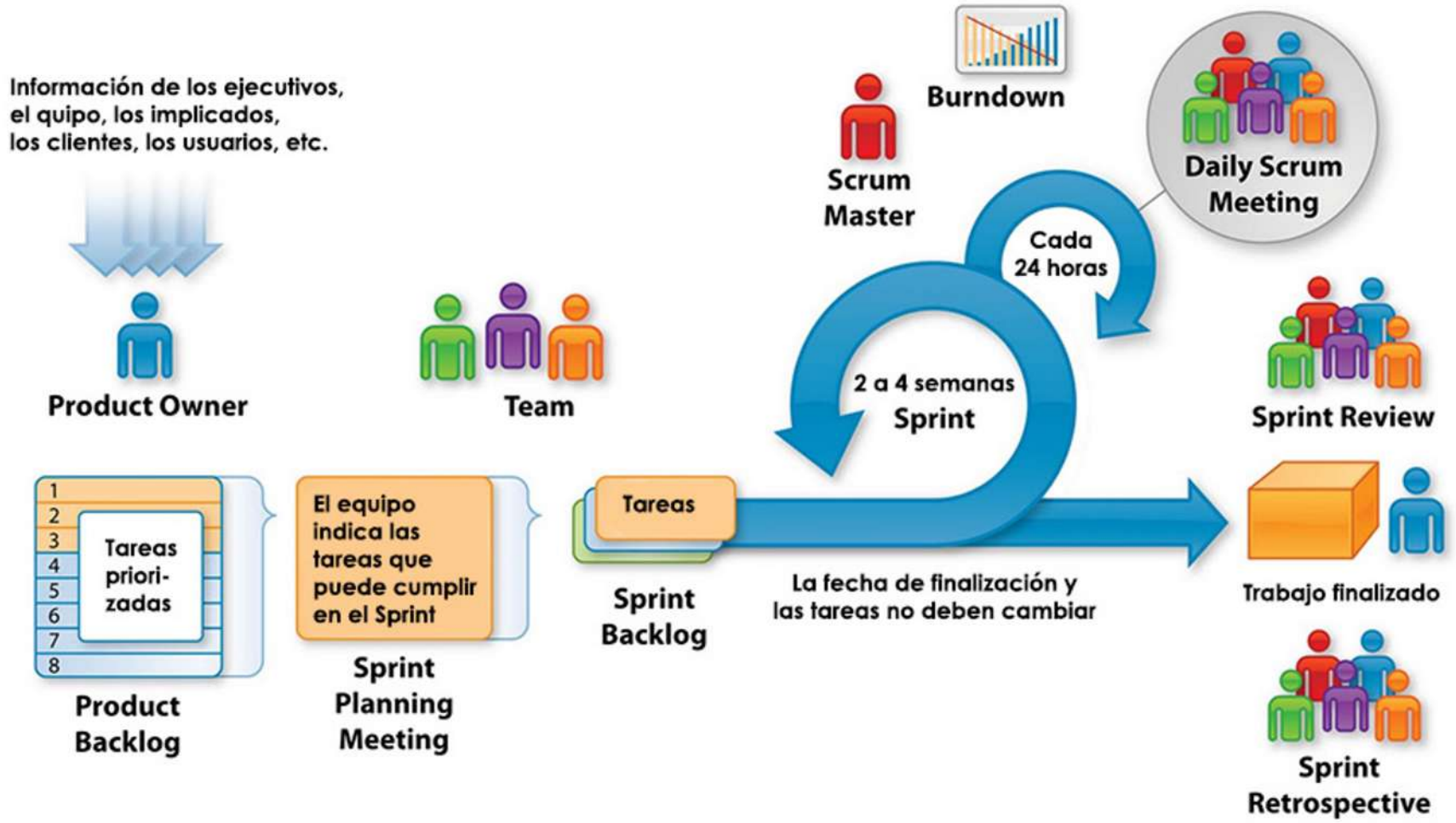
Conceptos

Metodologías ágiles



Conceptos

Metodología ágil Scrum



Desarrollo de Software I

Ideas para recordar de la clase de hoy

1

Crisis del software

Problemas que presentan los proyectos de desarrollo de software

2

Mitos Del Desarrollo de software

Malas interpretaciones de conceptos o aspectos del desarrollo de software

3

¿Qúe es ingeniería de software?

Aplicar conceptos, métodos y herramientas para crear aplicaciones de software.

4

Diferencias

Escribir programas
vs
Desarrollar software

5

Ciclo de vida

Proceso de creación de software

6

Metodologías

Proceso o marco estructurado para el desarrollo de software

7

Metodologías tradicionales

RUP, …

8

Metodologías ágiles

SCRUM, …

Preguntas ?

Próxima clase requerimientos de software ?







Gracias

Universidad del Valle