Análisis de requerimiento de software

Correo profe - [jose.m.castellanos@tec.mx](mailto:jose.m.castellanos@tec.mx)

La primera reunión para discutir los procesos de creación de software se dio en 1968 el Comité de ciencia de la OTAN reconoció la brecha entre el desarrollo del hardware y el desarrollo de software.

Frase de Bauer – “El software no se hace de una manera clara ni estandarizada, lo que necesitamos es un proceso de ingeniería en el software.”

**¿Qué es el software?**

“El conjunto de instrucciones detalladas que controlan la operación de un sistema computacional. Su función es:

* Administrar los recursos computacionales para la organización
* Facilitar herramientas para que las personas aprovechan esos recursos
* Ser el intermediario entre la organización y su información” Blum (1992)

Esta definición está enfocada al software de tipo empresarial.

**Los productos de software**

**Programa** – De un desarrollador a un usuario (Por lo general es muy sencillo que solo tiene una tarea)

Si un programa aumenta el numero de desarrolladores se genera un **sistema de programas** (Programming System).

Si hay pocos desarrolladores y muchos usuarios tenemos **un producto de programa** (Programming Product).

Si hay muchos programadores y muchos usuarios se genera un **producto de sistema de programa** (Programming System Product).

**El software hoy**

- Las economías desarrolladas dependen fuertemente del software

- Cada vez más empresas se crean específicamente dirigidas a procesos de software.

- El costo del software forma parte importante del gasto de una empresa.

- Mas dispositivos son controlados por software

“Todas las empresas hoy son empresas de tecnología” – Satya Nadella (CEO Microsoft)

**Características de un buen software**

* **Mantenible** a las necesidades del usuario.
* **Confiable** con los usuarios y las personas que dependen de su ejecución.
* **Eficiente** con recursos diversos.
* **Adaptable** a las características de sus usuarios.

**¿Qué es la ingeniería de Software?**

“La aplicación práctica del conocimiento científico en el diseño y construcción de programas computacionales y la documentación necesaria para diseñar, operar y mantener el software.”

“Disciplina de la ingeniería que se ocopa de todos los procesos de producción de software.”

- Se trata de elaborar procesos de desarrollo de software efectivos en recursos.

- Sistemas grandes requieren procesos complejos.

- Mantenimiento indispensable.

- Minimización de riesgos.

**El desarrollo de software se divide en 4 partes (procesos)**

1. Especificaciones de software. -> Define qué debe de hacer el sistema

2. Desarrollo de software. -> Define cómo se organiza e implementa el sistema

3. Validación del software. -> Verifica y valida que el sistema hace que el cliente desea (necesita)

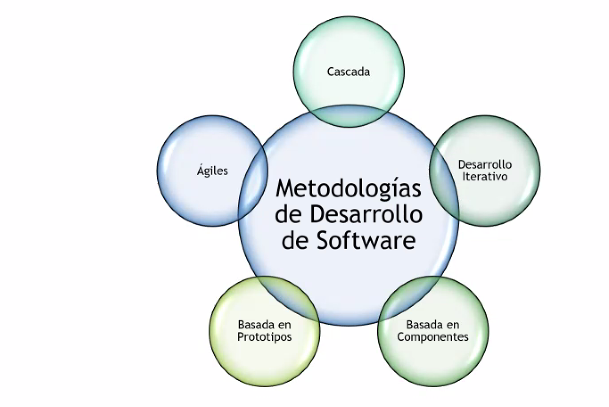
4. Evolución del software. -> Ajusta/modifica el sistema, de acuerdo con el cambio en las necesidades del cliente.

**La descripción del proceso normalmente incluye:**

* Actividades de los involucrados
* Productos (Artefactos)
* Roles y Responsabilidades
* Pre-condiciones
* Post-condiciones

**Modelos de procesos de Desarrollo de Software**

* Normalmente llamados “Metodologías de Desarrollo”

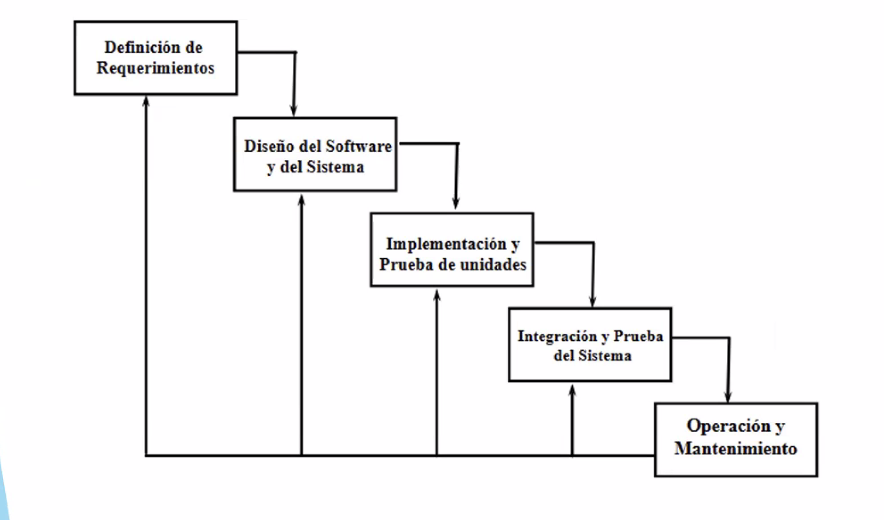


* No existen modelos buenos o malos

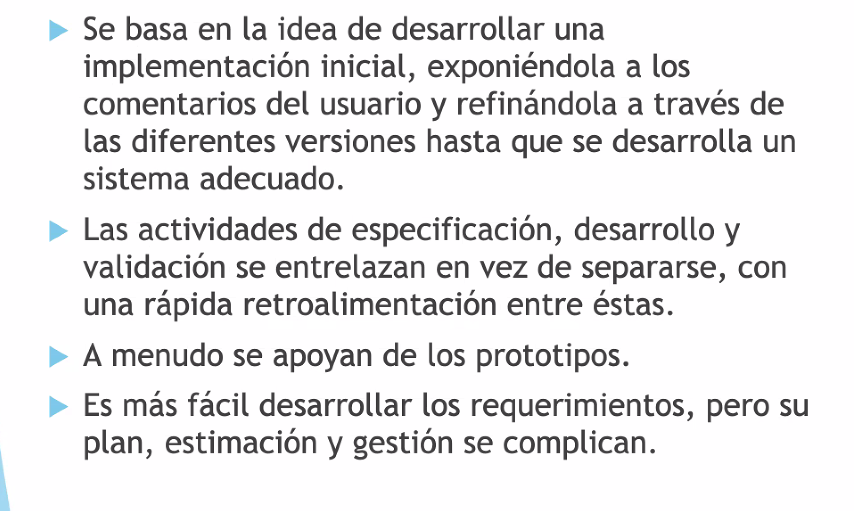
**Modelo en CASCADA**

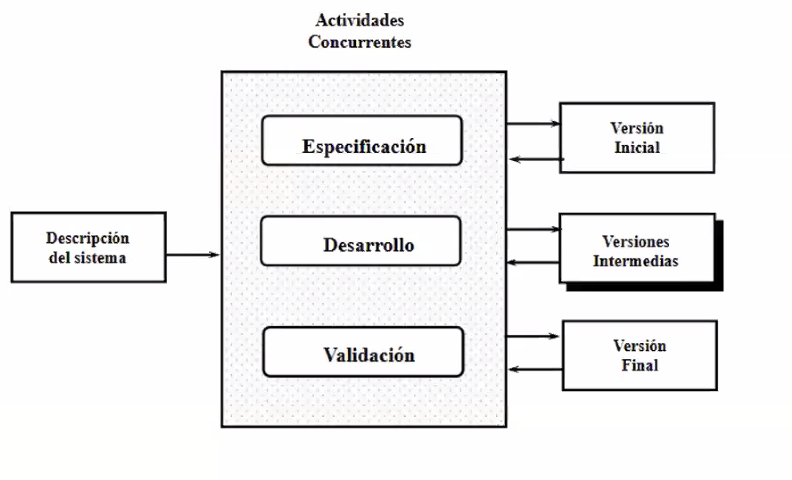
* Es un modelo guiado por un Plan (plan-driven): todas las actividades del proceso son planeadas desde el inicio
* Las fases se realizan de forma independiente y secuencial, sin poder avanzar hasta que s e termine la previa
* Las principales etapas de este modelo se transforman en actividades fundamentales de desarrollo:

1. Análisis y definición de requerimientos
2. Diseño del sistema
3. Implementación y pruebas unitarias
4. Integración y prueba del sistema
5. Operación y mantenimiento



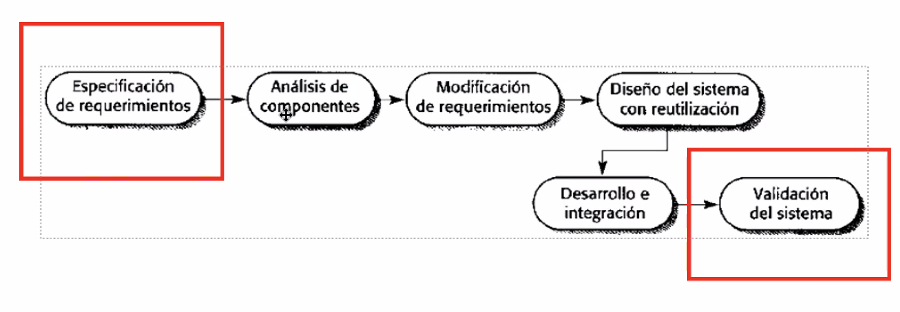
**Modelo Iterativo (Evolutivo)**





**Modelo BASADO EN COMPONENTES**

* Se basa en la reutilización de código
* Se denomina Ingeniería del Software Basada en Componentes (CBSE por sus siglas en ingles)
* Usa una gran base de componentes de software reutilizables y algunos marcos de trabajo de integración para estos
* Aunque la etapa de especificación de requerimientos y la de validación son comparables con otros procesos, las etapas intermedias son diferentes.



**Metodologías Ágiles de Desarrollo de Software**

* Estas metodologías surgieron a finales de los 90’s, intentaban reducir los tiempos de entrega de los proyectos de software.
* Se apoyan universalmente en el **enfoque incremental** para la especificación, el desarrollo y la entrega de software.
* Son mas adecuados par el diseño de aplicaciones en que los requerimientos del sistema cambian frecuentemente.
* Cuenta con una filosofía detrás de los métodos agiles se refleja en el manifiesto ágil.

**Manifiesto Ágil**

Estamos descubriendo mejores formas para desarrollar software, al hacerlo y al ayudar a otros a hacerlo. Gracias a este trabajo llegamos a valorar:

* A los individuos y las interacciones sobre los procesos y herramientas.
* Al software operativo sobre la documentación exhaustiva
* La colaboración con el cliente sobre la negociación del contrato
* La respuesta al cambio sobre el seguimiento de un plan

Esto es, aunque exista valor en los objetos de la derecha, valoraremos más de la izquierda.



**Principios del Agilismo**

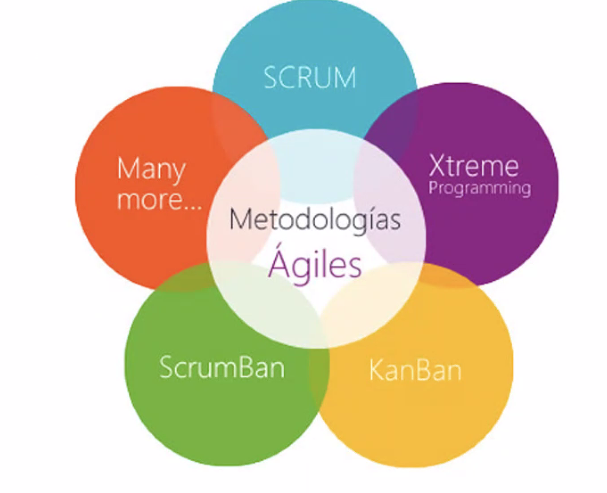
****

**Ideal para cierto tipo de desarrollos:**

Desarrollo de producto:

* Donde una compañía de software elabora un producto pequeño o mediano para su venta

Desarrollo de Sistema a Medida:



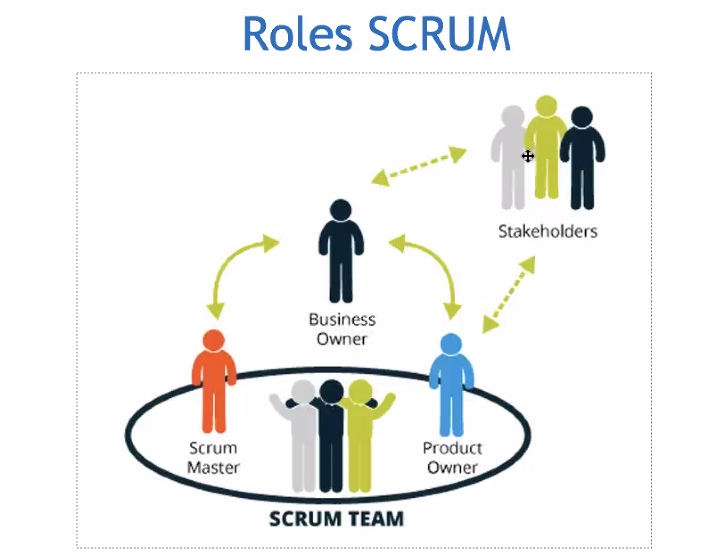
**Programación Extrema (XP)**

* Toma el nombre por su enfoque de llevar a niveles “extremos” las prácticas conocidas.
* En XP muchas versiones actuales de un sistema pueden desarrollarse mediante diferentes programadores, integrarse y ponerse a prueba en un solo día. (Solo es viable si se hace en equipo)
* Los requerimientos se expresan escenarios (historias de usuario), que se implementan directamente como una serie de tareas
* Los trabajadores trabajan en pares y antes de escribir el código, desarrollan pruebas para cada tarea.
* Los clientes intervienen estrechamente en la especificación y priorización de los requerimientos del sistema.



**SCRUM**

* Es un método de trabajo ágil en el que se aplican un conjunto de buenas prácticas para trabajar colaborativamente.
* Minimiza los riesgos derivados de los desarrollos muy largos.
* Es primordial la colaboración con el cliente, pues es quien establece las prioridades en función de sus necesidades.
* Esta indicado para proyectos en entornos complejos, donde se necesitan obtener resultados de manera inmediata uy donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.



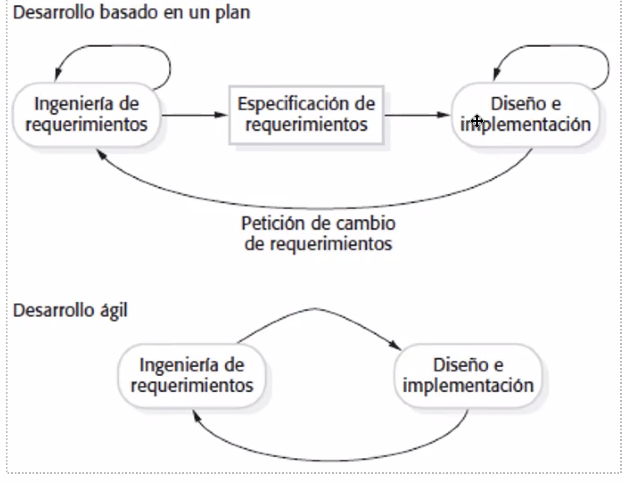
**SCRUM – BENEFICIOS**

* Gestión de las expectativas del cliente: El cliente establece prioridad y expectativas a cada requerimiento
* Reducción en tiempos de desarrollo y puesta en marcha: Al finalizar cada iteración se entregan funcionalidades completas y funcionales
* Capacidad de adaptación: El cliente está revisando el producto a final de cada iteración
* Aumento de la productividad: Constante revisión del trabajo, realizada por el propio equipo
* Estimación de esfuerzo contina: Los miembros del equipo estiman de manera conjunta

**Inconvenientes de las metodologías ágiles**

* Su éxito radica en tener un cliente que desee y pueda pasar tiempo con el equipo de desarrollo
* Quizás algunos miembros del equipo no cuenten con la personalidad adecuada para la participación intensa.
* Priorizar los cambios es difícil, sobre todo en sistemas donde existen muchos participantes.
* Mantener la simplicidad requiere trabajo adicional.
* En grandes compañías resulta difícil moverse hacia un modelo de trabajo donde los procesos son informales.

Los enfoques agiles consideran el diseño y la implementación como las actividades centrales.



**Situación Problema y el método ágil que mejor se adapta es:**

**SCRUM**

**Tipos de requerimientos**

* **Funcionales**

Describen funcionalidad o servicios del sistema

Dependen del tipo de software que se este desarrollando, de los usuarios esperados del software y del enfoque general que adopta la organización

Requerimientos funcionales del usuario -> Se describen de forma abstracta que entiendan los usuarios

Requerimientos funcionales del sistema –> Detallan las funciones del sistema

**Algunos Problemas (Requerimientos Funcionales)**

Los problemas comienzan cuando no se definen en forma precisa, es decir, cuando no son completos y/o consistentes.

**Completos –** Deben definirse todos los servicios requeridos por el usuario

**Consistentes –** Evitar definiciones contradictorias

El problema más común de la inconsistencia surge cuando se definen requerimientos ambiguos -> El enunciado se puede interpretar de varias formas.

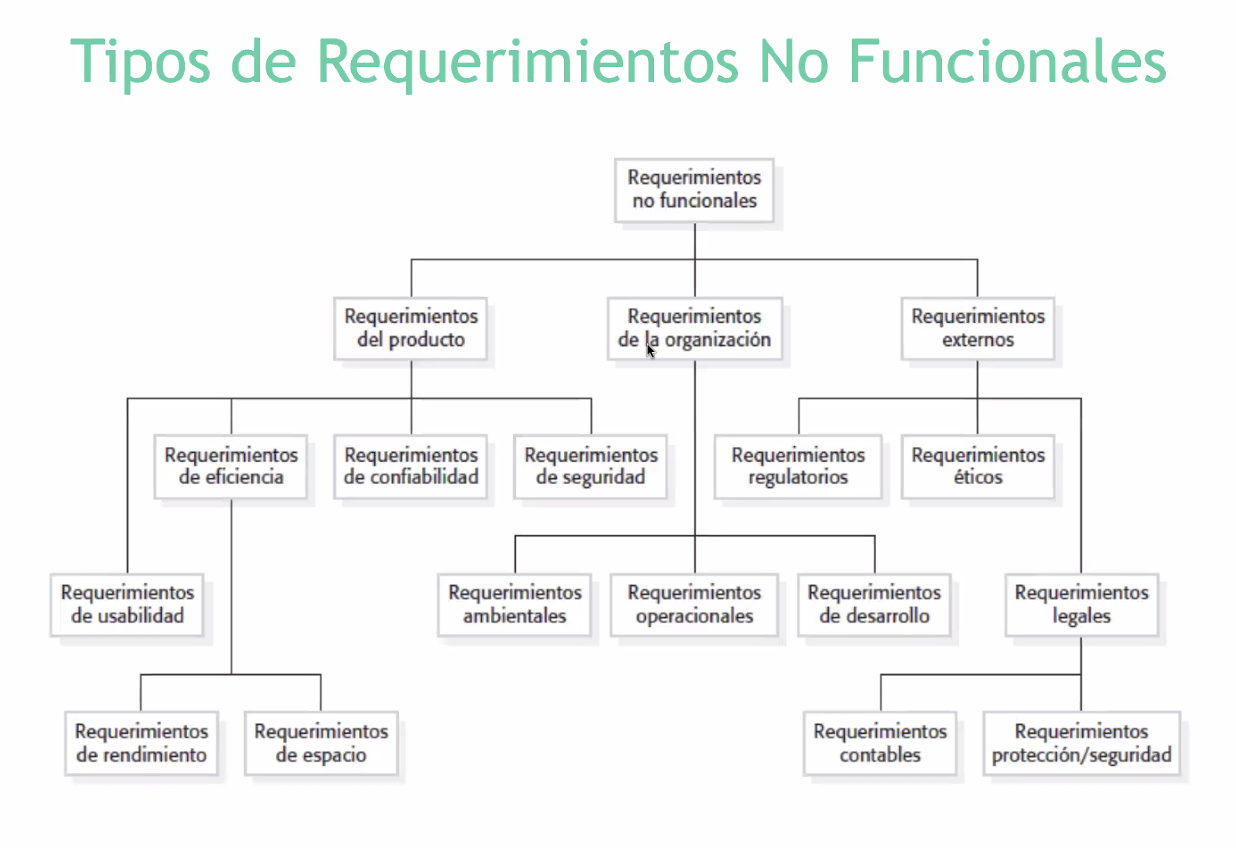
* **No funcionales**

No se relacionan directamente como los servicios específicos que el sistema entrega a los usuarios.

Se relacionan con propiedades emergentes del sistema, como confiabilidad, tiempo de respuesta y uso de almacenamiento.

Otros tipos de requerimientos No Funcionales pueden ser:

* Restricciones de capacidades de los dispositivos
* Representaciones de datos de interfaces externas
* IDE o lenguaje de programación específico
* Método de desarrollo de software
* Estándares de programación y/o documentación.



Se dividen principalmente en 3

**Requerimientos del Producto**

* Especifican o restringen el comportamiento del software
* Ejemplos:

Rendimientos, Fiabilidad, Seguridad, Usabilidad.

**Requerimientos de la Organización**

* Derivados de políticas y procedimientos de la organización (del cliente y/o desarrollador).
* Ejemplos:

Proceso operacional, proceso de desarrollo, estándares ambientales.

**Requerimientos Externos**

* Derivados de factores externos al sistema y su proceso de desarrollo
* Condiciones para ser aprobado por un regulador
* Ejemplos:

Requerimientos legislativos, requerimientos éticos.

**Proceso de Ingeniería de Requerimientos**

Este proceso depende del dominio de la aplicación., la gente involucrada y la organización que desarrolla los requerimientos.

A pesar de esto, existen ciertas actividades comunes para cualquier tipoi de proceso:

1. Elicitación de Requerimientos
2. Análisis de Requerimientos
3. Validación de Requerimientos
4. Administración de Requerimientos

**Elicitación y Análisis**

* Conocido como **Descubrimiento** o **Adquisición** de Requerimientos
* Los ingenieros de software trabajan junto con clientes y usuarios finales para descubrir el dominio de aplicación, qué servicios debe proporcionar el sistema, el desempeño requerido de éste, las restricciones de hardware, etc.
* Los participantes incluyen: usuarios finales, ingenieros de otros sistemas relacionados, administradores de negocios, expertos de dominio, representante de asociaciones sindicales. A estos participantes se les conoce como **Stakeholders.**

**Entrevistas y Cuestionarios**

* Se formulan preguntas a los participantes sobre elk sistema que actualmente usan y el sistema que se va a desarrollar
* Dos tipos de entrevistas: abiertas y cerradas

**Historias y Escenarios**

* Se usan ejemplos reales de cómo se usaría el sistema.
* Los escenarios son particularmente útiles para detallar un bosquejo de descripción de requerimientos.

**Caso de Uso**

* Identifica a los actores implicados en una interacción y nombre el tipo de interacción.
* Esto se complementa con información adicional que describe la interacción con el sistema.

**Etnografía**

* Técnica de observación que se usa para entender los procesos operacionales y ayudar a derivar requerimientos de apoyo.
* Un analista se adentra en el ambiente laboral donde se usará el sistema.