

## Técnicas de Elicitación de Requerimientos

Proceso que se realiza para adquirir el conocimiento necesario para producir un modelo de requerimientos de un dominio de un problema, conocer el sistema actual e identificar las necesidades explícitas e implícitas del mismo.

### Técnicas Discretas

Insuficientes para recopilar información por si solos, deben combinarse con otros métodos

#### Muestreo de documentación

Se hace en base a documentos existentes (bases de datos, sistemas actuales, diagramas, etc)

#### Investigaciones y visitas al sitio

Investigar el dominio, consultar otras organizaciones, etc.

#### Observación del ambiente del trabajo

El analista observa a los trabajadores y las actividades manteniendo bajo perfil.

## Técnicas Interactivas

Se basan en hablar con las personas en la organización y escuchar para comprender

### Planeación Conjunta de Requerimientos (JRP)

Reuniones estructuradas que se realizan para analizar problemas y definir requerimientos. Ahoran tiempo e involucran a los usuarios y la gerencia activamente pero es complicado de organizar. Además requieren entrenamiento y equipo específico como por ejm. proyectores.

## • Brainstorming

Se ofrecen ideas sin análisis hasta que no haya más ideas, se buscan ideas creativas y en gran cantidad.

Ayudan a entender el dominio y resolver la falta de consenso.

## • Cuestionarios

Se usan para obtener actitudes, creencias, comportamientos y características de un grupo grande o disperso de personas.

Son rápidos, económicos, anónimos y fáciles de analizar.

Son rígidos, difíciles de hacer y tienen pocas preguntas

## • Tipos de Preguntas:

### Abiertas

Abiertas a todas las opciones de respuesta

### Cerradas

Limitan o cierran todas las opciones de respuesta.

## • Entrevistas

Recaudan información de las personas mediante una conversación. Tienen propósito específico y generalmente se basan en preguntas y respuestas.

Generan inclusión con el entrevistado y adaptabilidad en las preguntas para una mejor retroalimentación.

Son costosas y no aplicables a distancia.

## • Tipos de entrevistas:

### Estructuradas

Usan un conjunto de preguntas sobre algo puntual y no permite adquirir un gran conocimiento del dominio

### No estructuradas

Hay un tema general sin preguntas específicas

### Tipos de preguntas:

#### Abiertas

Preguntas a responder como quiera el entrevistado  
Espontáneas. Además, revelan más preguntas  
Dan datos irrelevantes y pueden hacer que se pierda el  
control de la entrevista.

#### Cerradas

Preguntas directas o cortas

Ahorran tiempo, son fáciles de controlar y solo dan  
datos relevantes

No obtienen detalles

#### Sondeo

Preguntas enfocadas en un tema puntual

#### Organizaciones:

##### Piramidal

preguntas cerradas → abiertas



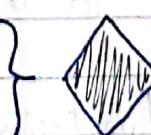
##### Embudo

preguntas abiertas → cerradas



##### Diamante

preguntas cerradas → abiertas → cerradas



## Técnicas de Especificación de Requerimientos

Las especificaciones de requerimientos tienen como objetivo que los devs le expliquen al cliente como ven el sistema y sus funcionalidades y que demostraciones llevar a cabo para que el cliente vea que el sistema resultante es lo que pidió.

### Técnicas Estáticas

Describen el sistema a través de las entidades, sus atributos y sus relaciones con otros enfocándose en el estado actual del sistema y no los cambios que pueden afrontar.

### Técnicas Dinámicas

Consideran el sistema en función de los cambios que puede afrontar a futuro.

### Historios de Usuario

Descripción corta y simple de un requerimiento de un sistema.

Se escriben en lenguaje común del usuario y desde su perspectiva.

Son independientes unas de otras, negociables, valoradas por los usuarios, son estimables en tiempo, pequeñas y verificables.

### Beneficios

cortas, fáciles de mantener, ideales para proyectos volátiles o para dividir proyectos en pequeñas partes.

### Limitaciones

Difíciles de escalar, requieren contacto constante con los usuarios.

## Casos de Uso

Modelado de las funcionalidades del sistema en eventos entre los usuarios y el sistema

### Beneficios

Captura requerimientos funcionales, permiten la comunicación con el usuario, descompone el alcance del mismo en piezas manejables.

### Componentes

#### Diagrama

Interacciones entre el sistema y los actores

#### Escenarios

Interacciones entre el actor y el sistema

#### Caso de Uso

Funcionalidad individual del sistema

#### Actor

Papel desempeñado por los usuarios

#### Relaciones

Entre ellos : Asociaciones, Extensiones, Uso e inclusión, Dependencia y Herencia.

## Diagrama de Transición de Estados

### Máquinas de estado finito

Conjunto de estados donde el sistema reacciona a ciertos eventos posibles (externos o internos)

## Redes de Petri

Utilizadas para especificar sistemas de tiempo real en los que son necesarios representar aspectos de concurrencia. Es un multigráfico, bipartito, dirigido.

Son asincrónicas y el orden en que ocurren los eventos es uno de los permitidos.

La ejecución es no determinística.

Pueden ser aplicadas para la modelación de sistemas de eventos discretos.

## Tablas de decisión

Herramienta que representa las reglas lógicas para decidir acciones a ejecutar en función de las condiciones y la lógica de decisión de un problema específico.

### Componentes

Condiciones (V o F)

Acciones simples

$Z^N$  Reglas donde  $N$  es la cantidad de condiciones

### Componentes

#### Sitios

Son los estados del modelo

### Transiciones

Son los eventos de nuestro sistema.

### Arcos

Indican la relación entre sitios y transiciones

### Tokens

Sirven para habilitar o deshabilitar transiciones

### Analisis Estructurado

Representación gráfica para lograr una comprensión más profunda del sistema a construir y comunicar a los usuarios lo comprendido enfocándose en el procesamiento de los datos en los procesos.

Busca reconocer como se mueven los datos, sus procesos o transformaciones y sus resultados.

### Stakeholders

Cualquier persona o grupo que se verá afectado por el sistema, directa o indirectamente, entre ellos, usuarios finales, ingenieros, gerentes, expertos del dominio, etc.

### Puntos de vista

#### Interactuadores

Personas u otros sistemas que interactúan directamente con el sistema. Pueden influir en los requerimientos.

#### Indirectos

Stakeholders que no usan el sistema pero influyen en los requerimientos.

#### Dominio

Características o restricciones del dominio que influyen en los requerimientos.

### Requerimientos

Característica del sistema o descripción de algo que el mismo es capaz de hacer en pos de satisfacer el propósito del sistema.

## ● Requerimiento Funcional

Describen lo que el sistema debe hacer o incluso cómo no debe comportarse.

Descripción detallada de una función o capacidad que un sistema de software debe cumplir o ser capaz de resolver

Ver

## ● Requerimiento No Funcional

Restricciones que limitan las elecciones de construcción de soluciones al problema.

Relacionados con la calidad, el rendimiento, la seguridad, la privacidad, aspectos legales, la eficiencia, la fiabilidad, la usabilidad, etc.

## ● Modelos de Proceso de Software

Representación simplificadora de un proceso de software que presenta una visión del mismo. Establece todas las actividades y pueden estar compuestos por subprocesos.

## ● Modelos Prescriptivos

Prescriben un conjunto de elementos del proceso: actividades del marco de trabajo, acciones de la ingeniería de software, tareas, aseguramiento de la calidad y mecanismos de control y están interrelacionadas.

## ● Modelos Descriptivos

Describen como se realizan en la realidad

## ● Modelos Tradicionales

Conjunto de fases o actividades que no toman en cuenta la evolución del software

### Modelo en Cascada

Es simple y secuencial, se debe terminar la parte anterior para pasar a la siguiente

### Desventajas

Necesita terminar para ver resultados

Difícil de manejar

Necesita más pruebas al final ya que ahí es donde ocurren más errores/fallos.

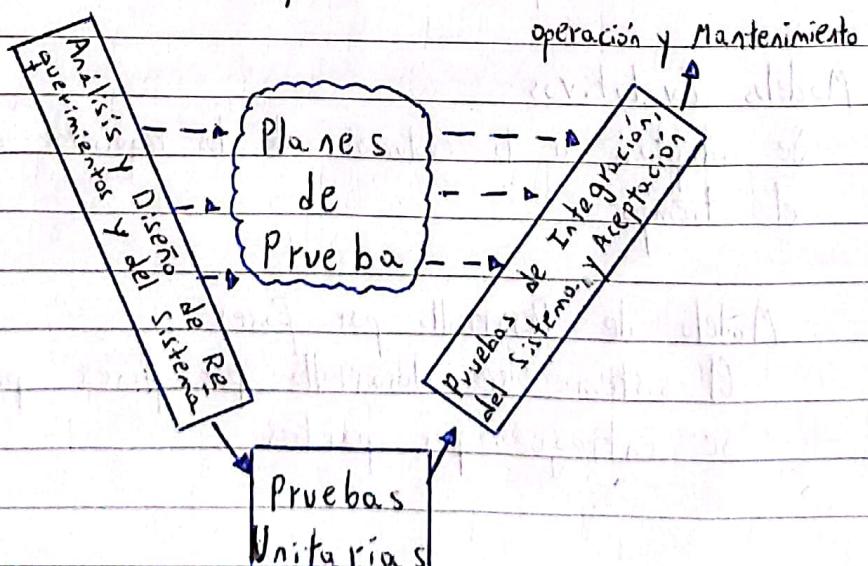
No se adapta bien a los cambios

### Modelo en V

Demuestra cómo se relacionan las actividades de prueba con las de análisis y diseño.

Sugiere que la prueba unitaria y de integración también sea utilizada para verificar el diseño del programa.

Al vincularse el lado derecho con el izquierdo se implica que, si se encuentran problemas en las verificaciones y validaciones, entonces el lado izquierdo de la V es el que se ejecuta devuelta para solucionar el problema.



## Modelo de Prototipos

### Prototipo

Producto parcialmente desarrollado que permite que se examinen algunos aspectos del sistema.

### Características recomendadas para su uso

El sistema ha de ser experimental.

El costo ha de ser bajo ( $< 10\%$ )

Debe ser de desarrollo rápido.

Enfoquado en la interfaz de usuario.

Equipo de devs reducido

Herramientas y lenguajes adecuados

### Tipos

#### Evolutivo

Busca obtener el sistema final y permite construir rápidamente partes del sistema para analizar y tener una comprensión de lo que se necesita y lo propuesto como solución.

#### Descartable

Sin funcionalidad

## Modelos Evolutivos

Se adaptan a la evolución de los requisitos del sistema en el tiempo

### Modelo de Desarrollo por Fases

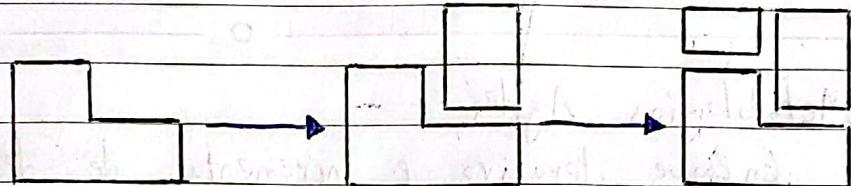
El sistema se desarrolla por fases permitiendo que se entregue por partes

### Tipos

#### Incremental

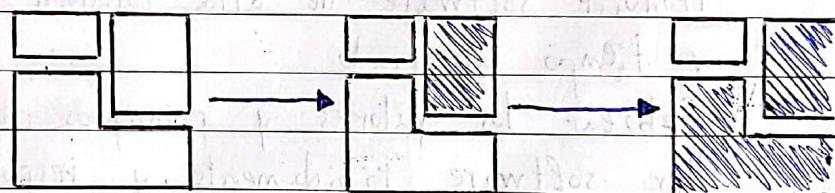
Se subdivide en subsistemas por funcionalidad.

Se entrega por subsistemas



#### Iterativo

Entrega un sistema completo y va aumentando las funcionalidades de los subsistemas con nuevas versiones



### Modelo en Espiral

#### Características

- Combina actividades de desarrollo con la gestión del riesgo

- Incorpora objetivos de calidad

- Trata de mejorar los ciclos de vida clásicos y prototipos

- Elimina errores y alternativas poco atractivas al inicio

- Permite iteraciones

- En cada ciclo se identifican primero los objetivos de su porción y sus alternativas

## Restricciones

Cada ciclo se completa con una revisión que incluye todo el ciclo anterior y el plan para el siguiente ciclo a desarrollar.

## Metodologías Agiles

Enfoque iterativo e incremental de desarrollo de software que emerge como una posible solución a las metodologías con énfasis en el control del proceso, definiendo roles, actividades, herramientas y documentación.

## Objetivos

Producir software de alta calidad con un costo efectivo y tiempo apropiado

Esbozar los valores y principios que permiten desarrollar software rápidamente y responder a situaciones cambiantes

Ser una alternativa a procesos de desarrollo de software rígidos

Dan prioridad a las tareas con resultados directos y reducen la burocracia lo más posible

Metodología Agil	Metodología No Agil
Pocos Artefactos	Más Artefactos
Pocos Roles	Más roles
Sin contrato o uno muy flexible	Existe un contrato prefijado
El cliente es parte del equipo de devs	El cliente interactúa mediante reuniones
Grupos pequeños y en el mismo sitio	Grupos grandes
Menos énfasis en la arquitectura	La arquitectura es esencial

### ● Desventajas

Los clientes no intervienen por completo en el desarrollo del software.

Priorizar los cambios es difícil en sistemas grandes.

Possiblemente no haya tiempo para las simplificaciones que hay que realizarle al sistema.

El cambio a otro modelo de trabajo es duro.

Complejidad a la hora de reglamentar los documentos de requerimientos.

### ● Extreme Programming (XP)

Consiste en llevar a todo el equipo reunido en la presencia de prácticas simples, con información suficiente para ver donde están y ajustar las prácticas a su situación particular.

### ● Prácticas / Características

● Testing.

● Refactoring del código.

● Programación de Pares.

● Propiedad colectiva del código.

● Integración continua de piezas de código.

● Semana de 40 horas laborales.

● Cliente en el lugar de desarrollo.

● Estándares de codificación.

● Simplicidad de código.

Los elementos claves son Historias de Usuario, Roles, Proceso y Prácticas

### ● Roles

Conformados por Programador, Jefe de Proyecto, Cliente, Entrenador, Encargado de pruebas y Rastreador

Asamblea

## Scrum

Marco de trabajo para el desarrollo colaborativo para productos complejos

Proporciona un paradigma de trabajo que soporta la innovación, permite la auto organización del equipo y garantiza entregas parciales y regulares, priorizadas por su beneficio al receptor del proyecto. Es usado en proyectos con requerimientos dinámicos y tecnología de punta.

## Principios

- Eliminar el desperdicio
- Construir la calidad con el producto
- Crear conocimiento
- Diferir las decisiones
- Entregar rápido
- Respetar a las personas
- Optimizar el todo

## Roles

- Propietario (conoce y marca las prioridades)
- Jefe (garantiza el seguimiento de la metodología)
- Equipo (implementan las funcionalidades)
- Usuario / Cliente (beneficiarios finales)

## Artefactos

- Product Backlog (lista maestra de todas las funcionalidades)
- Sprint Backlog (lista de funcionalidades a desarrollar en un Sprint)
- Burndown Chart (acumulativo del trabajo hecho día a día)

## Proceso

Iterativo e incremental con fases de desarrollo solapadas en ejecución hasta que se

## ● Software

Instrucciones, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación

## ● Características

Es un elemento lógico

Se desarrolla

No se desgasta

No sigue una curva de envejecimiento normal porque el problema está en los nuevos cambios y no en el tiempo de operación

## ● Calidad

Conjunto de propiedades o características de un producto o servicio que le confieren aptitud para satisfacer necesidades expresadas o implícitas

## ● Calidad de Software

De gran importancia en la industria por la gestión de calidad y las técnicas de gestión de la misma

Compuesta por 2 calidades dependientes entre ellas

## ● Calidad del Producto

La estandarización del producto define las propiedades que debe satisfacer el producto resultante

## ● Calidad del Proceso

La estandarización del proceso define la manera de desarrollar el producto.

Calidad de Producto de Software	Se evalúa mediante	ISO/IEC 25000	Compuesto por distintos modelos. Define características que pueden estar o no en el producto. La norma permite evaluar si están o no y de qué manera.
Calidad de proceso de desarrollo de software	Se evalúa mediante	ISO/IEC 12207  ISO/IEC 15504 (reemplazada por ISO 33000)	Establece un modelo de procesos para el ciclo de vida del software. Define como tiene que ser el modelo de proceso para ser completo y con calidad.  Norma internacional para establecer y mejorar la capacidad y madurez de los procesos. Define lo que hay que tener en cuenta para evaluar el modelo y si este es completo y con calidad.
		ISO/IEC 90003	Guía sobre como aplicar la ISO 9001 en un proceso de software
Calidad de procesos/servicios en general	Se evalúa mediante	CMMI	Marco estructurado para evaluar los procesos actuales, establecer propiedades de mejora e implementar las mismas. Es utilizada en organizaciones de medianas a grandes dimensiones
		ISO 9001	Determina los requisitos para establecer un Sistema de Gestión de Calidad. Parte de la familia ISO 9000, que es un conjunto de normas de "gestión de la calidad" aplicables a cualquier organización y, eventualmente obtener una certificación