

Definición de requisitos

Proceso

- **Extracción o determinación de requisitos.** Proceso mediante el cual los clientes o futuros usuarios del software descubren, revelan, articulan y comprenden los requisitos que desean.
- **Análisis de requisitos.** Proceso de razonamiento sobre los requisitos obtenidos en la etapa anterior, detectando y resolviendo posibles inconsistencias o conflictos, coordinando los requisitos relacionados entre sí, etc.
- **Especificación de requisitos.** Proceso de redacción o registro de los requisitos. Suele recurrirse a un lenguaje natural, lenguajes formales, modelos, gráficos, etc.
- **Validación de los requisitos.** Confirmación, por parte del usuario o el cliente de que los requisitos especificados son válidos, consistentes, completos, etc.

Aunque estas actividades no tienen por qué realizarse en secuencia, ya que hay muchas iteraciones y solapamientos entre ellas, sí marcan un proceso general para la fase de definición de requisitos.

¿Una mala captura de requisitos?



Técnicas de recogida de información

- **Entrevistas**
- **Reuniones**
- **Prototipado**
- **Observación**
- **Cuestionarios**
- **Estudio de documentación**
- **Tormenta de ideas (brainstorming)**

METRICA V3 cuenta con guías para algunas de estas técnicas

Cuestionarios

- Para usuarios dispersos, numerosos, poco disponibles, etc.
 - Distinto de encuesta: no hay entrevistador presente
- Algunas indicaciones de diseño:
 - Tener claros objetivos de información que deseamos obtener
 - Preguntas muy claras y precisas (no presentes para aclarar) y una sola cuestión en cada pregunta
 - Cortos y sencillos (evitar aburrimiento)
 - Preguntas cerradas con unas pocas abiertas
 - Introducción con propósito y que agradezca colaboración
 - Alguna pregunta final de sugerencias, olvidos, etc.

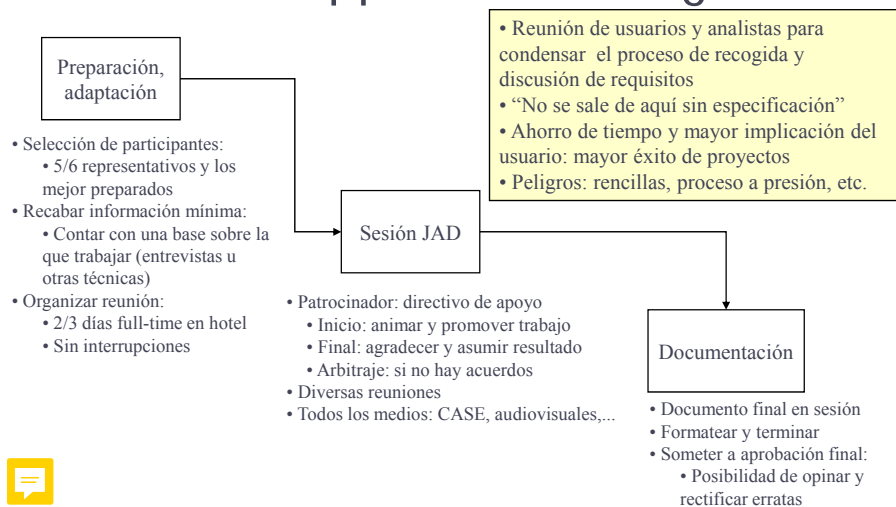
Entrevistas - Resumen

- Entrevistas
 - Obtener información de forma individual
 - Preparar un guion previo que se remite al entrevistado
 - Obtener la máxima información sin provocar rechazo
 - Al final resumir las conclusiones para aclarar malentendidos
 - Enviar el acta al entrevistado para fijar ideas por escrito



Reuniones

- Procedimiento
 - Obtener información dispersa entre varios usuarios, comunicar información, tomar decisiones
 - Preparar el orden del día y convocar la reunión
 - Al inicio resumir objetivos y método de trabajo
 - Importancia de la persona que dirige la reunión
 - Al final resumir las conclusiones para aclarar malentendidos, destacar puntos pendientes y fijar siguiente reunión o actividades/responsables
 - Enviar el acta a los participantes para fijar ideas por escrito
- Diversos tipos

JAD: Joint Application Design



JRP (Joint Requirements Planning)

- Potenciar la participación activa de la alta dirección. Muy útil en el desarrollo del Plan Estratégico de SI 
- Las características de las sesiones JRP y JAD son comunes en cuanto a la dinámica del desarrollo de las sesiones y la obtención de los modelos con el soporte de las herramientas adecuadas.
- Diferencias: nivel más alto en la organización en cuanto a visión global del negocio y capacidad de decisión. Tipo de información de salida :
 - Modelos de procesos de la organización.
 - Modelo de información.
 - Modelo de sistemas de información, etc.

Prototipado

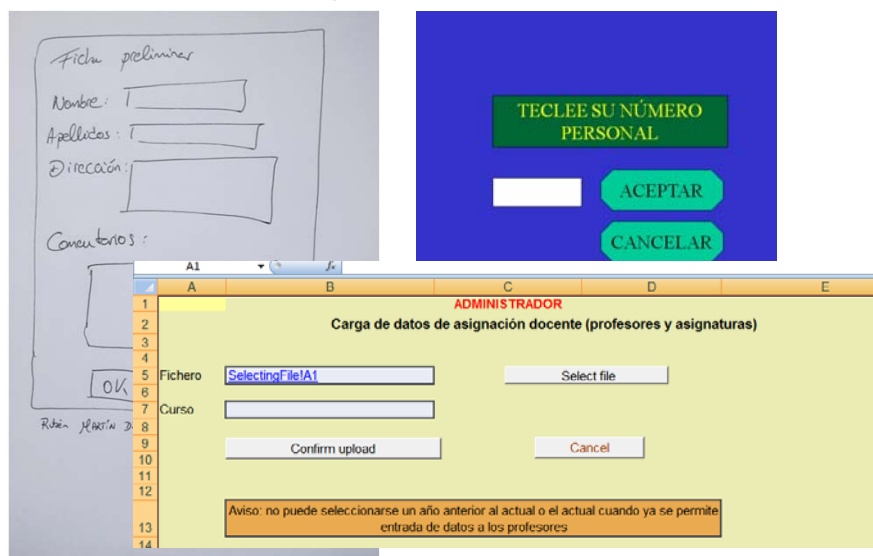
- Creación de “maquetas” de sistemas
 - para evaluación de usuario y/o desarrolladores
 - según objetivo, requiere herramientas distintas (Office, entornos visuales, programación, etc.)
 - coste/esfuerzo alto (¿10% de proyecto?)
- Aplicación:
 - área poco definida: dificultad o sin tradición
 - coste alto de rechazo
 - evaluar impacto previamente
- Tipos:
 - interfaz de usuario: más habitual
 - prototipado funcional evolutivo
 - reaprovecha código (ciclo de vida evolutivo)

Prototipado y métodos afines

- Maquetas (Mock Ups)
 - Visión simplificada de la aplicación a desarrollar
 - Muestra para el usuario de como quedará y a que se parecerá la interfaz del software sin haber programado aplicación ni funcionalidad
 - Representa la interfaz gráfica del sistema con trazos sobre papel o con herramienta visual.
- Storyboards*
 - Se encargan de presentar secuencia de interfaz gráfica con las actividades del sistema
 - Papel o herramientas como Visio, Excel, PowerPoint,...

* [Guía para la creación de storyboards en el diseño de aplicaciones](https://www.researchgate.net/profile/Monica_Forero3/publication/316845657_Guia_para_crear_storyboard_en_el_proceso_del_diseno_de_interfaz/links/591358e9a6fdcc963e7ee05e/Guia-para-crear-storyboard-en-el-proceso-del-diseno-de-interfaz)
 (https://www.researchgate.net/profile/Monica_Forero3/publication/316845657_Guia_para_crear_storyboard_en_el_proceso_del_diseno_de_interfaz/links/591358e9a6fdcc963e7ee05e/Guia-para-crear-storyboard-en-el-proceso-del-diseno-de-interfaz)

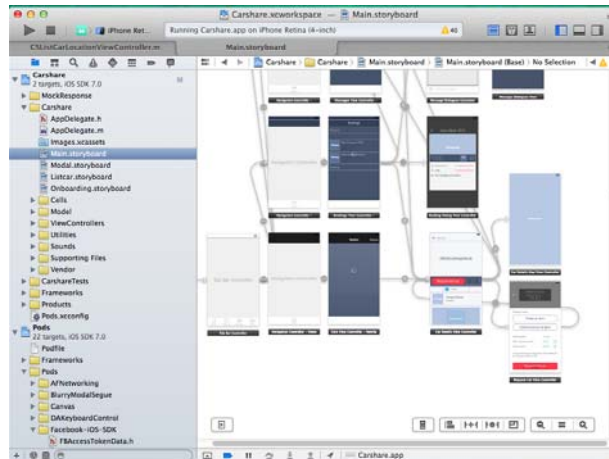
Ejemplos (I)



Ejemplos (II)

Tutorial para crear interfaces usando storyboard para desarrolladores IOS (Apple) :

<https://www.efectoapple.com/introduccion-los-storyboards-parte-1/>



Otras técnicas

- **Observación:**
 - Captar mejor la realidad del trabajo diario
 - “Pasar una mañana en el puesto”, “pasar por cliente”, *mystery shopping*, etc.
- **Estudio de documentación:**
 - Siempre necesario: no se necesita interlocutor
 - Imprescindible como fase previa a otras técnicas
 - Análisis: procesos formales/documentados, normas, impresos, formularios, legislación, etc.
 - Mejor tener siempre muestras de documentos rellenos: apreciar uso real
- **Brainstorming:**
 - Crear nuevas ideas cuando no hay tradición del problema o de software existente
 - Técnica creativa muy usada en publicidad, etc.
 - Ronda de ideas en grupo sin evaluar su bondad
 - Al parar, leer lo dicho previamente y empezar otra ronda



Dificultad de comunicación



Resistencia al cambio

- Ser humano: animal de costumbres
 - Se resiste incluso a cambios buenos
- Causas / Soluciones:
 - Resistencia al personal informático
 - Equipos conjuntos, coordinación de directivos, comprensión del trabajo informático, formación de usuarios
 - Percepción de que “el proyecto no es bueno”
 - Involucrar en el proyecto, análisis de coste/beneficio
 - No percibir necesidad de cambio (de forma de trabajo, de sistema, ...)
 - Lista de ventajas del nuevo sistema frente al antiguo
 - Miedo a perder poder o puesto. Computerfobia: miedo tecnológico
 - Formación, integración en el proyecto
 - Sistemas mal diseñados para usuarios
 - Tener en cuenta la usabilidad y accesibilidad en el desarrollo

Análisis de requisitos

- **Compleitud.**
 - No hay omisiones: no faltan requisitos (propiedad global) y no faltan detalles en la especificación de cada requisito (propiedad individual).
 - Difícil de determinar: contrastar con el cliente, comparar con proyectos semejantes, buscar la visión de conjunto, detectar huecos o partes infra-especificadas,...
- **Detección de Conflictos e Inconsistencias**
 - Los requisitos se agrupan por categorías y se organizan en sub-conjuntos, se estudia cada requisito en relación con el resto y se clasifican en base a las necesidades de los clientes/usuarios.
 - Es corriente en clientes y usuarios solicitar más de lo que puede realizarse o proponer requisitos contradictorios.

Preguntas planteadas en una Especificación de requisitos del software (ERS)

- **La funcionalidad.**
 - ¿Qué tiene que hacer el software ?
- **Las interfaces externas.**
 - ¿Cómo el software actúa recíprocamente con las personas, el hardware y otros sistemas hardware o software?
- **La actuación.**
 - ¿Cuál es la velocidad, la disponibilidad, tiempo de respuesta, tiempo de la recuperación, etc.?
- **Los atributos de calidad.**
 - ¿Que fiabilidad, mantenibilidad, portabilidad, seguridad, etc. necesita el sistema?
- **Las restricciones del diseño.**
 - ¿Hay alguna restricción de idioma, hardware, recursos, etc.?

Características de una buena ERS

- IEEE Std. 830:
 - No ambigua
 - Completa
 - Fácil de verificar
 - Consistente
 - Fácil de modificar
 - Ordenación por prioridades
 - Facilidad para identificar fuente y efectos de cada requisito (trazabilidad)
 - Facilidad de uso en explotación y mantenimiento



Requisitos funcionales y no funcionales (IEEE std. 610)

- Requisito funcional:
 - Requisito que especifica una función que un sistema o componente debe ser capaz de realizar
 - Ejemplos de un sistema de gestión de notas:
 - Obtener estadísticas de notas
 - Introducir nota de prácticas
 - Gestionar datos anagráficos del alumno
- Requisito no funcional:
 - Requisito que especifica una característica del sistema
 - Seguridad, rendimiento, facilidad de uso, capacidad, etc.
 - Ejemplos de un sistema de gestión de notas:
 - Capaz de gestionar 100 alumnos y hasta 10 exámenes y 5 prácticas por alumno
 - Seguridad de acceso a las notas basada en clave de 128 bits

Categorías requisitos no funcionales



Estándar ERS: IEEE std. 830

- **Introducción**
 - Objetivo, ámbito, definiciones y siglas, referencias
- **Descripción general**
 - Visión general de producto, funciones, usuarios, limitaciones generales, supuestos y dependencias
- **Requisitos específicos numerados**
 - Funcionales
 - Interfaz
 - Rendimiento, restricciones de diseño, calidad, otros
- **Apéndice**
 - No obligatorio, por ejemplo: ejemplos de formato de entrada/salida

Técnicas de especificación

- **Formales:**
 - Basadas en lógica formal: ej., notación Z
 - $usados \cap libres = \emptyset$
- **Semiformales:**
 - Lenguaje natural claro y preciso
 - Puede acompañarse de:
 - Gráficos (diagrama de flujo de datos, diagrama de estados,...)
 - Texto basado en gramáticas (pseudocódigo de proceso, BNF,...)
 - Plantillas (descripción de datos,...)
 - Matrices: para comprobar más que definir, por ejemplo Matriz de trazabilidad (requisito/quien lo genera, quien lo aprueba)

Validación de requisitos

- **Objetivo:**
 - Comprobar que los requisitos definidos en la ERS son correctos.
- **Los parámetros a validar en los requisitos son :**
 - Validez: Todos los usuarios involucrados conocen y están de acuerdo con los requisitos definidos.
 - Consistencia: No debe haber contradicciones entre unos requisitos y otros.
 - Completitud: Deben estar todos los requisitos. Esto es imposible en un desarrollo iterativo, pero, al menos, deben estar disponibles todos los requisitos de la iteración en curso.
 - Realismo: Se pueden implementar con la tecnología actual.
 - Verificabilidad: Tiene que existir alguna forma de comprobar que cada requisito se cumple.