Ingeniería de Software

Ingeniería de Requisitos
Clase 1

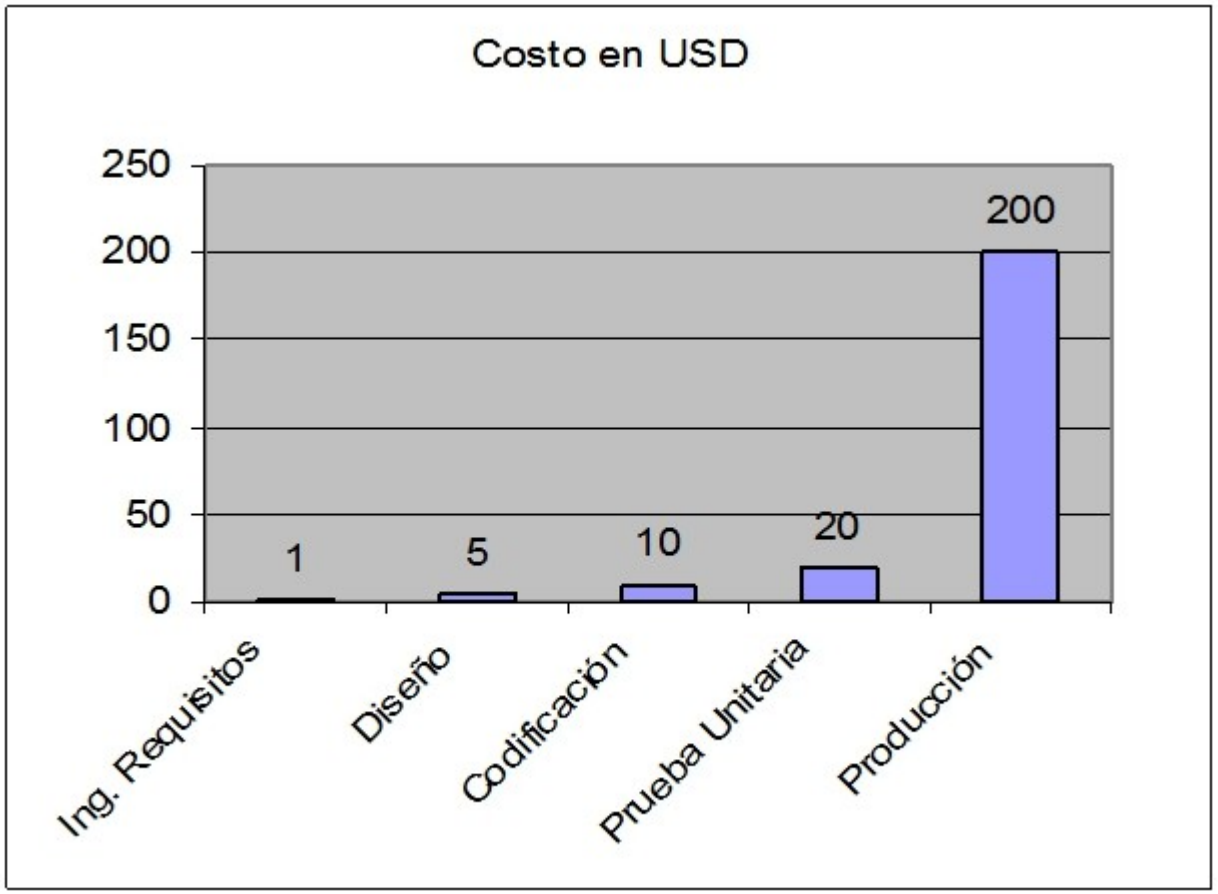
Sommerville 10 — Capítulo 4 (secciones 4.1, 4.2 y 4.3).
Wiegers — Capítulo 1 , 7 y 15

Requisitos

- Los requisitos de un sistema son descripciones de lo qué debería hacer el sistema— los servicios que provee y las restricciones en su operación.
- El proceso de descubrir, analizar, documentar y chequear esos servicios y restricciones es llamado ingeniería de requisitos (RE).

Costos de errores en los requisitos

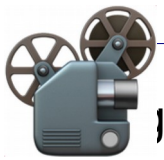
- Costo de corregir un error en los requisitos
  (Boehm-Papaccio,1988)



Requisitos – Ejercicio

Requisitos - Abstracción

- La forma de un requisito puede variar desde una oración con un alto nivel de abstracción hasta una especificación funcional matemática.
- Posibles contextos de un requisito:
  - Puede ser la base de una oferta para un contrato — por lo tanto debe estar abierto a la interpretación.
  - Puede ser la base para un contrato — por lo tanto debe ser definido detalladamente.



Tipos de requisitos - Abstracción

- **Requisitos de usuario**
  - Son declaraciones en lenguaje natural, además de diagramas de los servicios que proporciona el sistema y sus limitaciones operativas. Escritos para los clientes.

- **Requisitos del sistema**
  - Un documento estructurado que establece descripciones detalladas de las funciones, servicios y restricciones operacionales del sistema. Define lo que se debe implementar, podría ser parte de un contrato entre el cliente y el empresario.

Tipos de requisitos - Abstracción

Definición del requerimiento del usuario

1. El MHC-PMS elaborará mensualmente informes administrativos que revelen el costo de los medicamentos prescritos por cada clínica durante ese mes.

Especificación de los requerimientos del sistema

1.1 En el último día laboral de cada mes se redactará un resumen de los medicamentos prescritos, su costo y las clínicas que los prescriben.

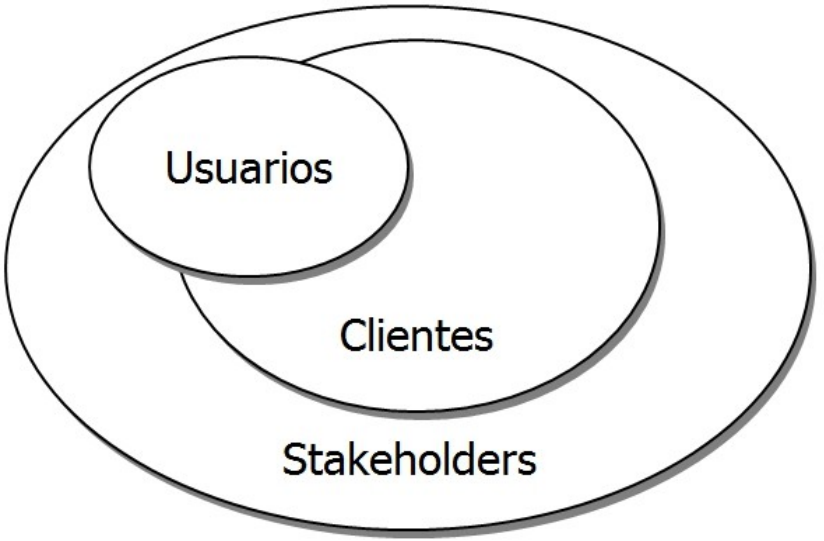
1.2 El sistema elaborará automáticamente el informe que se imprimirá después de las 17:30 del último día laboral del mes.

1.3 Se realizará un reporte para cada clínica junto con los nombres de cada medicamento, el número de prescripciones, las dosis prescritas y el costo total de los medicamentos prescritos.

1.4 Si los medicamentos están disponibles en diferentes unidades de dosis (por ejemplo, 10 mg, 20 mg) se harán informes por separado para cada unidad de dosis.

1.5 El acceso a los informes de costos se restringirá a usuarios autorizados en la lista de control de acceso administrativo.

Stakeholders - interesados



- **Cliente y Usuarios** — Requisitos adecuados a sus necesidades.
- **Diseñadores** — Comprenderlos para lograr diseño que los satisfaga.
- **Supervisores del Contrato** — Sugieren hitos de control, cronogramas.
- **Gerentes del Negocio** — Entienden impacto en la organización.
- **Verificadores** — Comprenderlos para poder verificar si el sistema los satisface.

Ejemplo de stakeholders

- Pacientes cuya información es registrada en el sistema.
- Los doctores quienes son responsables por evaluar y dar tratamiento a los pacientes.
- Las enfermeras quienes coordinan las consultas con los doctores y administran algunos tratamientos.
- Recepcionistas médicos que administran las citas de los pacientes
- El personal de TI que son responsables de la instalación y mantenimiento del sistema.
- Un administrador de ética médica que debe asegurarse de que el sistema cumple con las normas éticas vigentes para la atención del paciente.
- Gerentes de la salud quienes obtienen información para la gestión del sistema.
- Equipo de registros médicos, son responsables de asegurarse que la información del sistema se mantiene y se conserva y que los procedimientos de mantenimiento de registro se ejecutan correctamente.

Otros tipos de *información de requisitos*

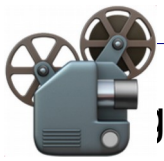
- **Requisitos del dominio**
  - Se derivan del dominio de aplicación del sistema y no de las necesidades específicas de los usuarios.
- **Requisitos del negocio**
  - Un objetivo de alto nivel de una organización que desarrolla un producto o de un cliente que lo compra.
- **Regla de negocio**
  - Una política, guía, estándar o regulación que define o restringe algún aspecto del negocio.
- **Requisito de interfaz externa**
  - Una descripción de una conexión entre un sistema de software y un usuario, otro sistema de software o un dispositivo de hardware.

Otros tipos de *información de requisitos*

- **Característica (*feature*)**
  - Una o más capacidades relacionadas de forma lógicas que proveen valor al usuario y son descriptas como un conjunto de requisitos funcionales. Ejemplo: los favoritos del navegador.
- **Requisito funcional**
  - Una descripción de lo que el sistema debe hacer bajo condiciones específicas.
- **Requisitos no funcionales**
  - Una descripción de una propiedad o característica que un sistema debe poseer o una restricción que debe respetar.
- **Atributo de calidad**
  - Un tipo de requisito no funcional que describe una característica de servicio o desempeño de un producto.

Requisitos funcionales y no funcionales

- **Requisitos funcionales**
  - Son declaraciones de los servicios que el sistema debería proveer, cómo el sistema debería reaccionar a entradas particulares y cómo debería comportarse en situaciones particulares.
  - Pueden indicar lo que el sistema no debería hacer.
- **Requisitos no funcionales**
  - Son restricciones a los servicios o funciones provistas por el sistema, como restricciones de tiempo, restricciones sobre el proceso de desarrollo, estándares, etc.
  - Generalmente son aplicables al sistema entero y no a servicios o funciones en particular.



Requisitos funcionales

- Describen funcionalidades o servicios del sistema.
- Dependen del tipo de software, de las expectativas de los usuarios y del enfoque con el cual se escriben los requisitos.

- Ejemplos:
  - Un usuario podrá buscar en las listas de citas de todas las clínicas.
  - El sistema deberá generar cada día para cada clínica una lista de pacientes que se espera asistan a la cita durante ese día.

Requisitos funcionales - Contexto

- Pueden variar entre lo que debe hacer el software (literalmente) hasta algo mucho más específico y describir cómo se hace el trabajo localmente en una organización.
- Si vamos a comprar o implantar sw → los requisitos deben indicar que información necesitan los stakeholders para hacer su trabajo o modelar la casuística particular de los procesos utilizados.



Falta de precisión en los requisitos

- La falta de precisión en los requisitos es la causa de muchos problemas en la ingeniería de software.
- Los requisitos ambiguos pueden ser interpretados de diferentes maneras por desarrolladores y usuarios.

- Ejemplo:
  - Un usuario podrá buscar en las listas de citas de todas las clínicas.

Búsqueda dentro de todas las clínicas a la vez

Búsqueda por nombre y apellido

Búsqueda dentro de cada clínica por vez

Búsqueda por hora de la cita

Completitud y consistencia de los requisitos

- Los requisitos deben ser lo más completos y consistentes posibles.

- **Completos**
  - Todos los requisitos requeridos por el usuario están definidos.

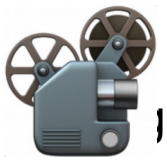
- **Consistentes**
  - En los requisitos no hay conflictos o definiciones contradictorias.

Requisitos no funcionales

- Definen las propiedades y las restricciones del sistema.

- Ejemplos:
  - Propiedades— confiabilidad, tiempo de respuesta y requisitos de almacenamiento.
  - Restricciones— capacidad de dispositivos de entrada/salida, las representaciones del sistema, etc.

- Los requisitos no funcionales pueden ser más críticos que los requisitos funcionales. Si no se cumplen el sistema puede resultar inútil.



Implementación de los requisitos no funcionales

- En general es posible identificar los componentes que implementan un requisito funcional específico. Es mucho más difícil identificar los componentes que implementan un requisito no funcional.

- Esto se debe a que:
  - Los requisitos no funcionales pueden afectar a la arquitectura general del sistema en lugar de a componentes individuales.
  - Un único requisito no funcional, por ejemplo un requisito de seguridad, puede generar varios requisitos funcionales que definen servicios requeridos para el sistema.

Tipos de requisitos no funcionales

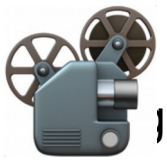
- **Requisitos del producto**
  - Especifican comportamiento del producto, por ejemplo la velocidad de ejecución, confiabilidad, etc.
- **Requisitos organizativos**
  - Son consecuencia de las políticas y procedimientos de la organización, por ejemplo estándares de procesos utilizados, requisitos de implementación, etc.
- **Requisitos externos**
  - Surgen de factores externos al sistema y su proceso de desarrollo, por ejemplo: requisitos de interoperabilidad, requisitos legislativos, etc.

Requisitos no funcionales verificables

- Los requisitos no funcionales pueden ser muy difíciles de declarar de forma precisa y los requisitos imprecisos pueden ser difíciles de verificar.
- **Objetivo**
  - Una intención general del usuario como la facilidad de uso
- **Requisito no funcional verificable**
  - Una declaración con alguna medida que pueda ser probada objetivamente.
- Los objetivos son útiles para los desarrolladores, ya que transmiten las intenciones de los usuarios del sistema.

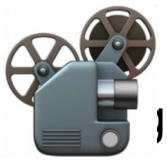
Problemas en los requisitos

- La mayor consecuencia es el **retrabajo** (en etapas más avanzadas del desarrollo o después de liberar).
- Ejemplos:
  - Poco involucramiento de los usuarios.
  - Planes inadecuados – utilizar requisitos muy vagos para crear planes.
  - Recortes en los requisitos del usuario.
  - Requisitos ambiguos.
  - Gold plating – chapado en oro: requisitos que creemos que el usuario va a amar.
  - No identificar correctamente a los usuarios correctos.

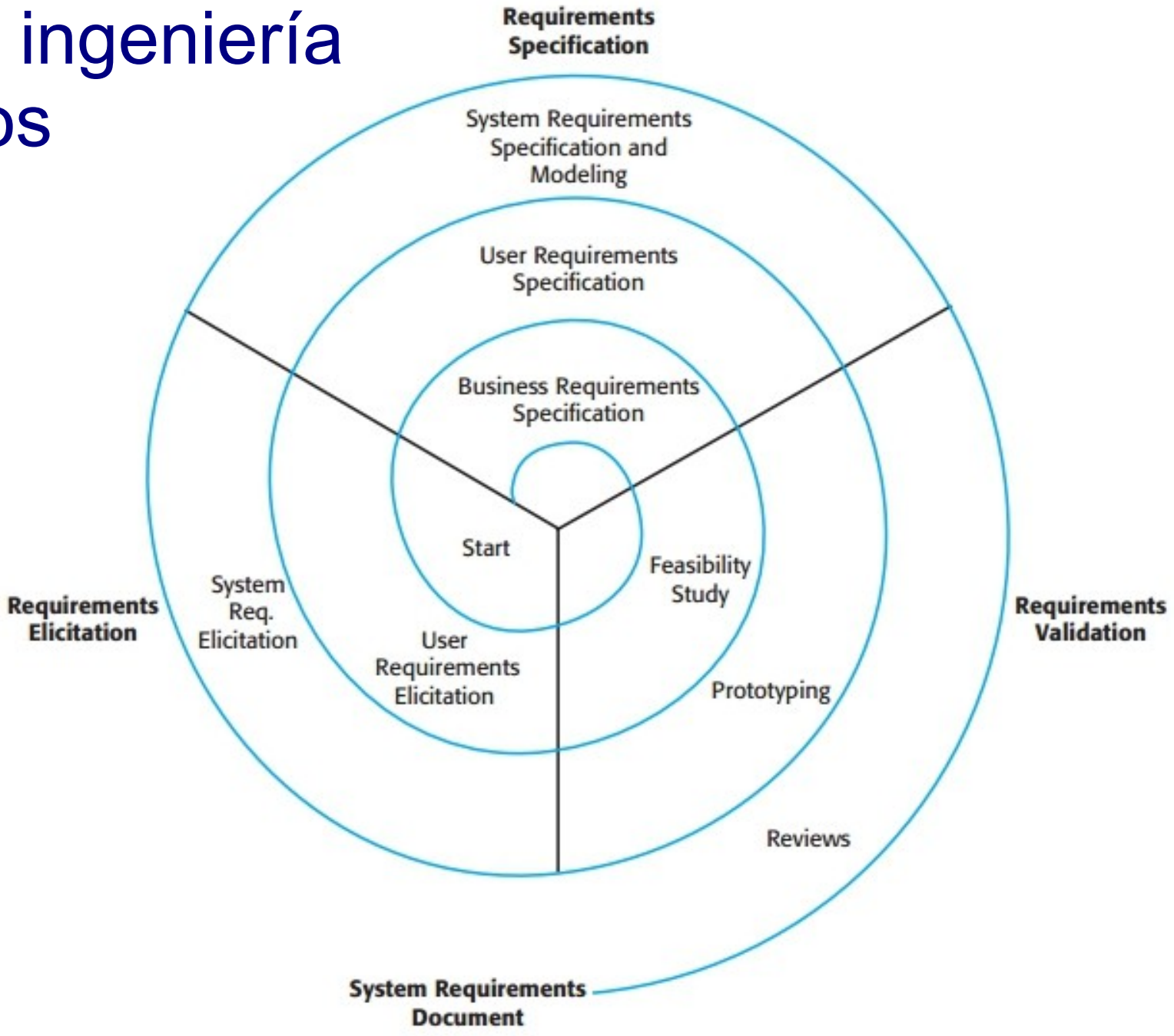


Los procesos de la ingeniería de requisitos

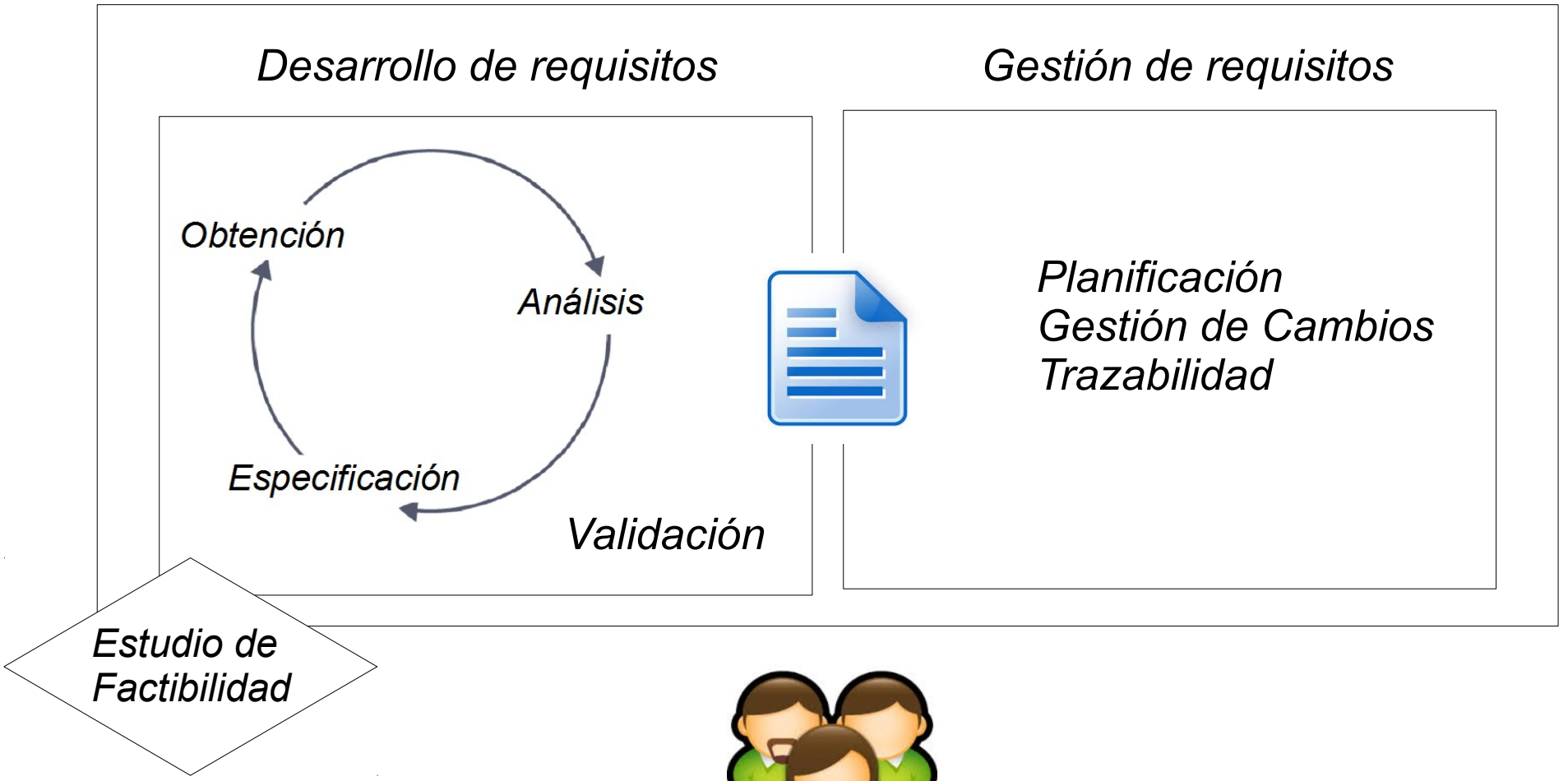
- Varían ampliamente dependiendo del dominio de la aplicación, las personas involucradas y la organización que desarrolla los requisitos.
- Sin embargo, hay una serie de actividades genéricas que son comunes a todos los procesos:
  - Estudio de factibilidad.
  - Relevamiento (u obtención) y análisis de requisitos.
  - Validación de requisitos.
  - Gestión de requisitos.
- En la práctica, la ingeniería de requisitos es una actividad iterativa en la cual se intercalan los procesos.



Una vista en espiral del proceso de ingeniería en requisitos



Actividades de la ingeniería de requisitos



Dificultades comunes en el proceso

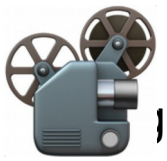
- Los stakeholders no saben lo que realmente quieren.
- Los stakeholders expresan los requisitos en sus propios términos.
- Diferentes stakeholders pueden tener conflictos con sus requisitos.
- Factores organizacionales y políticos pueden influir en los requisitos del sistema.
- Los requisitos cambian (o su prioridad) durante el proceso de análisis. Pueden surgir nuevos stakeholders y el entorno empresarial puede cambiar.
- Influyen factores políticos.

Estudio de factibilidad

- Tiene como objetivo averiguar sí vale la pena implementar el sistema y sí es posible implementarlo dadas las restricciones existentes (calendario, presupuesto, tecnología, etc).
- Recibe como entradas un conjunto de requisitos del negocio preliminares, una breve descripción del sistema y cómo será destinado a apoyar los procesos del negocio.
- Brinda como salida un informe que recomienda sí vale la pena o no realizar el proceso de desarrollo del sistema.
- Sí un sistema no da soporte a los objetivos del negocio, entonces no tiene ningún valor real para el negocio.

Proceso ingeniería de requisitos

- Los ingenieros de software trabajan con una variedad de stakeholders del sistema para obtener información acerca del dominio de la aplicación, los servicios que debe proporcionar el sistema, los requisitos de performance del sistema, restricciones de hardware, otros sistemas, etc.
- Las etapas incluyen:
  - Obtención de requisitos (descubrimiento, relevamiento).
  - Clasificación y organización de requisitos.
  - Priorización y negociación de requisitos.
  - Especificación de requisitos.



Actividades del Proceso

- **Obtención de requisitos**
  - Interactuar con los stakeholders para recolectar sus requisitos. Uno de los principios fundamentales es una efectiva comunicación entre los distintos stakeholders.
- **Clasificación y organización de los requisitos**
  - Los requisitos que están relacionados se ponen en un mismo grupo y luego se organizan en subgrupos coherentes.
- **La asignación de prioridades y la negociación**
  - Priorizar los requisitos y resolver los requisitos en conflicto.
- **Especificación de requisitos**
  - Los requisitos se documentan y se ingresa a la próxima ronda del espiral.

Requisitos – Fuentes

- Metas – Objetivos de alto nivel (*Goals*).
- Conocimiento del dominio.
- Stakeholders.
- Reglas del negocio.
- Ambiente operacional.
- Ambiente organizacional.

Técnicas para obtención de requisitos

- Entrevistas
- Investigar antecedentes
- Workshops – sesiones de trabajo
- Focus groups – grupos de enfoque
- Observaciones (incluyendo el caso de las etnografías)
- Cuestionarios
- Análisis de las interfaces del sistema
- Análisis de la interfaz de usuario
- Análisis de documentación
- Tormenta de ideas
- Escenarios
- Casos de uso
- Prototipado
- Modelado de procesos
- Historias de usuario

Entrevistas

- La manera más obvia de averiguar que necesitan los usuarios de un sistema de software es preguntarle a ellos.
- **Usar para:**
  - Entender el problema de negocio.
  - Entender el ambiente de operación
  - Evitar omisión de requisitos.
  - Mejorar las relaciones con el cliente.

- **Ventajas**
- Orientado a las personas
- Interactivo/flexible
- Rico

- **Desventajas**
- Costoso
- Depende de las habilidades interpersonales

Entrevistas

- **Tipos de entrevistas**
  - Entrevistas cerradas basadas en una lista de preguntas pre determinadas.
  - Entrevistas abiertas donde varios temas son explorados con los stakeholders.
- **Entrevista efectiva**
  - Tener la mente abierta, evitar ideas preconcebidas acerca de los requisitos y estar dispuesto a escuchar a los stakeholders.
  - Utilizando una pregunta como trampolín se consiguen discusiones con el entrevistado para lograr una propuesta de requisitos o trabajar juntos en un prototipo del sistema.

Entrevistas - ¿Qué preguntar?

- En general, las preguntas evolucionan naturalmente mientras transcurre las preguntas.
- Algunas preguntas genéricas:
  - ¿Dónde se inicia el proceso?
  - ¿Cómo se va a utilizar la funcionalidad?
  - ¿Qué es necesario que cumpla la funcionalidad?
  - ¿A quién le puedo preguntar para aprender más sobre esto?
  - ¿En qué casos se puede utilizar la funcionalidad? ¿En qué casos no?
  - ¿Hay otras maneras de realizar lo mismo?
  - ¿La funcionalidad cubre todas las necesidades de negocio relacionadas?

Entrevistas

- **Recomendaciones**
  - Establecer una buena relación: presentarse, revisar agenda, recordar los objetivos de la sesión y responder dudas.
  - Mantener el foco: mantener el foco de la discusión en los objetivos.
  - Preparar preguntas y maquetas previamente: en general a las personas les cuesta menos criticar contenido que crearlo.
  - Sugerir ideas: muchas veces los interesados no se dan cuenta de las capacidades de lo que es posible desarrollar.
  - Escuchar activamente: mostrar paciencia, dar feedback, consultar puntos confusos.

Entrevistas en la práctica

- Generalmente son una mezcla de entrevistas cerradas y abiertas.
- Las entrevistas son buenas para conseguir una comprensión global de que quieren los stakeholders y como podrían interactuar con el sistema.
- Las entrevistas no son buenas para entender los requisitos de dominio:
  - Los ingenieros de requisitos no pueden entender la terminología especifica del dominio.
  - Para algunas personas algunos de los conocimientos del dominio son tan familiares que les resulta difícil explicarlos o piensan que no vale la pena.

Investigar antecedentes

- Estudio, muestreo, visitas,…
- Buena forma de comenzar un proyecto
- Interna: estructura de la organización, políticas y procedimientos, formularios e informes, documentación de sistemas
- Externa: publicaciones de la industria y comercio, encuentros profesionales, visitas, literatura y presentaciones de vendedores

- **Ventajas**
- Ahorra tiempo de otros
- Prepara para otros enfoques
- Puede llevarse a cabo fuera de la organización

- **Desventajas**
- Perspectiva limitada
- Desactualizado
- Demasiado genérico

Workshops – Sesiones de trabajo

- Reuniones estructuradas en las cuales un selecto grupo de interesados y expertos trabajan en conjunto para definir, crear, refinar y acordar documentos y modelos que representen los requisitos de usuario.
- En general es necesario el rol de facilitador el cual explica las reglas y los objetivos al comienzo y luego mantiene la dinámica de trabajo y el foco en los objetivos. Debe también mantener la motivación de los participantes.
- Recomendaciones: mantener grupos pequeños, utilizar tiempos fijos para discutir cada tema, manejar una lista de temas que surjan para tratarlos después y no perder el foco.

- **Ventajas**
- Son muy efectivos para resolver desacuerdos que las entrevistas individuales.

- **Desventajas**
- Costosos en tiempo y recursos (a veces días completos).

Focus groups – Grupo focal

- Son grupos de usuarios que participan en una actividad de obtención de requisitos para generar contribuciones e ideas sobre los requisitos funcionales y de calidad de un producto.
- Es necesario contar con el rol de facilitador así como hacer una muy buena selección de los participantes del grupo.

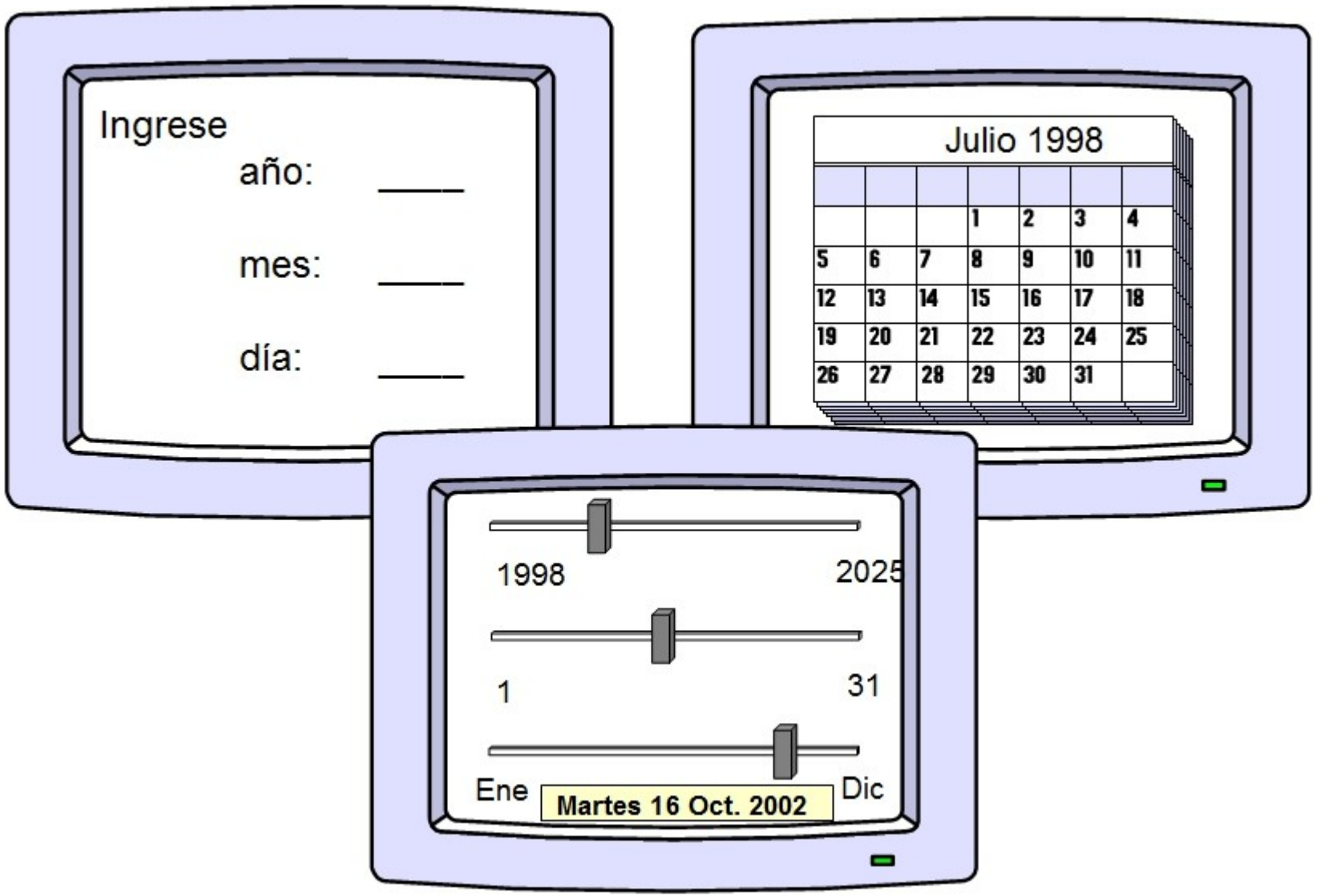
- **Ventajas**
- Son útiles para explorar actitudes, impresiones, preferencias y necesidades de los usuarios.
- Son en particular valiosos cuando no hay fácil acceso a los usuarios finales.

- **Desventajas**
- No hay que esperar resultados cuantitativos sino información subjetiva que sirve para evaluar y priorizar requisitos.

Prototipado

- Implementación parcial, permite a los desarrolladores y usuarios:
  - Entender mejor los requisitos.
  - Cuáles son necesarios, deseables.
  - Acotar riesgos.
- Aspectos para los que es frecuente construir prototipos:
  - Apariencia y percepción de la interfaz de usuario
  - Arquitectura (riesgos tecnológicos, tiempos de respuesta)
  - Otros aspectos riesgosos

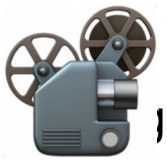
Prototipado



Prototipado

Técnica en detalle

- “If a picture is worth 1000 words, then a prototype is worth a 1000 meetings” – @ideo
- Las personas tienen dificultad para describir sus necesidades sin tener nada tangible adelante; criticar es mucho más fácil que crear.
- Los prototipos toman un paso tentativo en el espacio de soluciones.
- Un feedback temprano (utilizando prototipos) permite compartir con entendimiento con los stakeholders sobre los requisitos del sistema, lo cual reduce el riesgo de su insatisfacción.
- Los usuarios en general prefieren explorar un prototipo que leer un documento de requisitos.
- Los prototipos de software pueden ser diseños estáticos o modelos de trabajo; bocetos rápidos o pantallas detalladas; presentaciones visuales o porciones completas de funcionalidad; o simulaciones.



Prototipos – Alcance

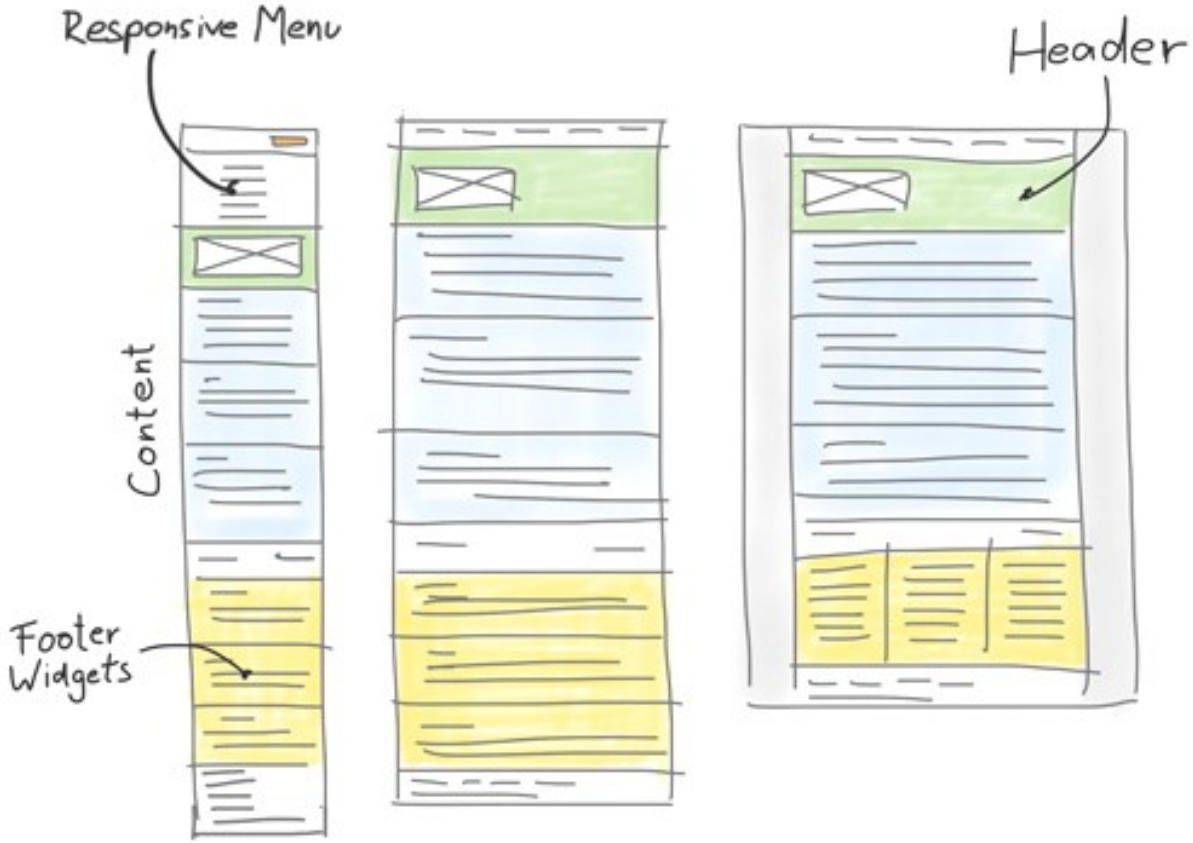
- **Mock-ups – Bosquejos**
  - Prototipos horizontales.
  - Se enfocan en porciones de la interfaz de usuario.
  - Permiten explorar comportamientos específicos del producto.
  - No realizan ningún trabajo útil, sólo lucen como si lo hicieran.

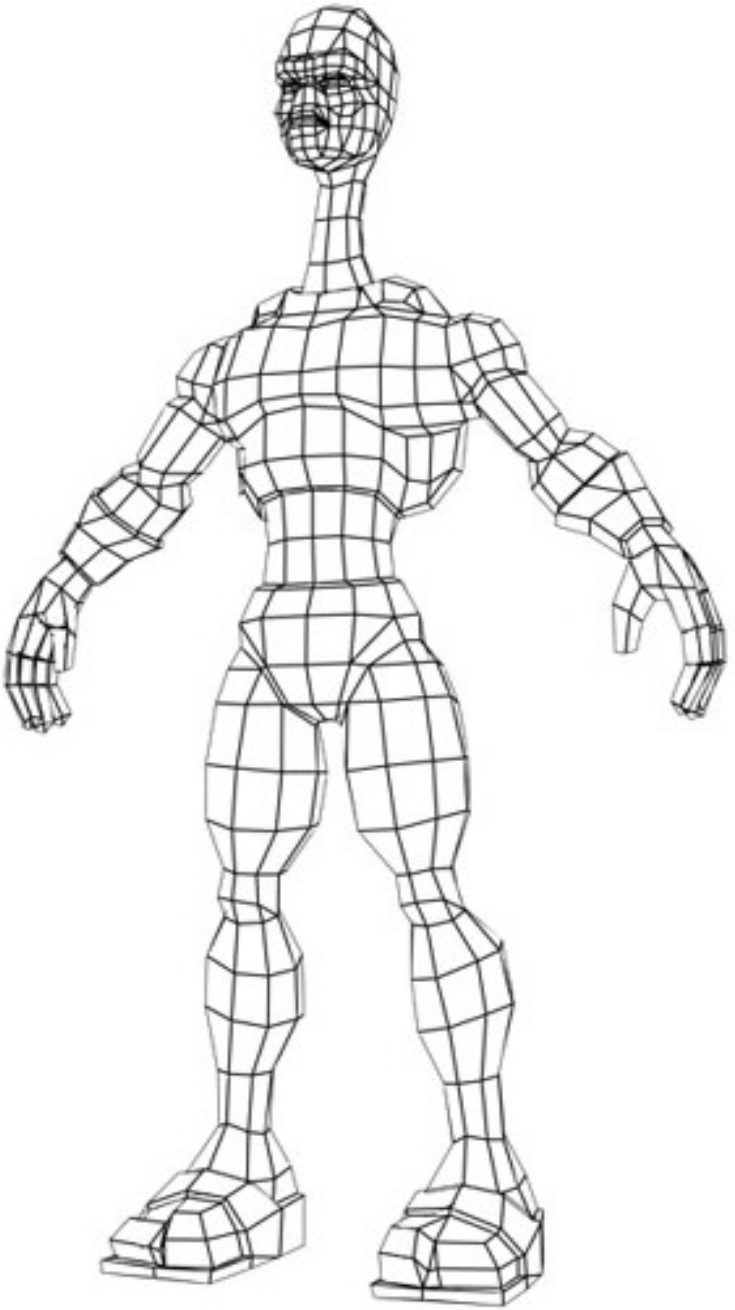
- **Pruebas de concepto**
  - Prototipos verticales.
  - Implementan una porción funcional de la aplicación.
  - Permiten resolver incertidumbres sobre la factibilidad de la arquitectura propuesta u otros riesgos técnicos.
  - Funcionan como el sistema real porque toca todos los niveles de la implementación.

Prototipos – Uso futuro

- **Desechables**
  - Sirven para responder preguntas, resolver incertidumbres y mejorar la calidad de los requisitos.
  - Conviene hacerlos lo más rápido y baratos posibles. Resistir la tentación de elaborarlos más de lo necesario.
  - Mucho cuidado con la calidad al incorporar algo de un prototipo desechable al producto final.
  - Ejemplo: wireframes.

Prototipos – Wireframes





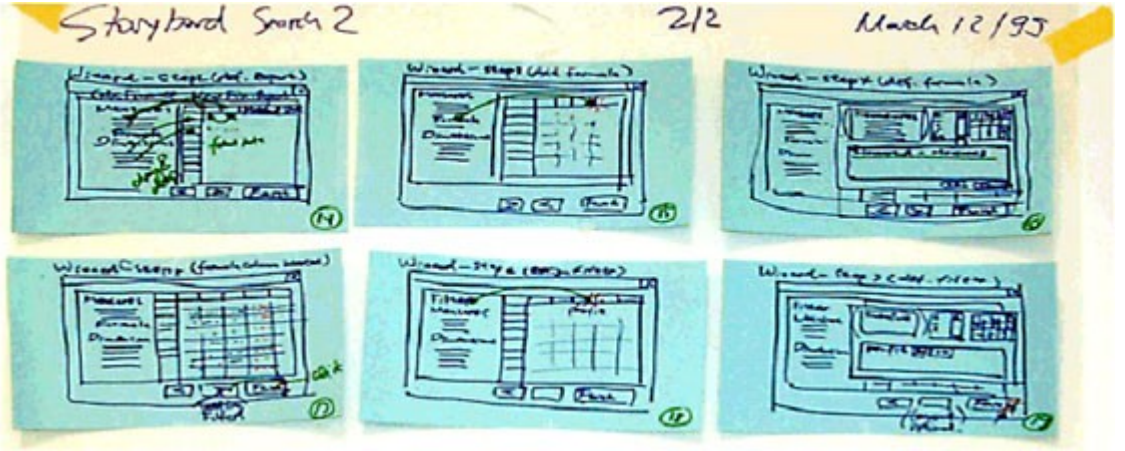
Prototipos – Uso futuro

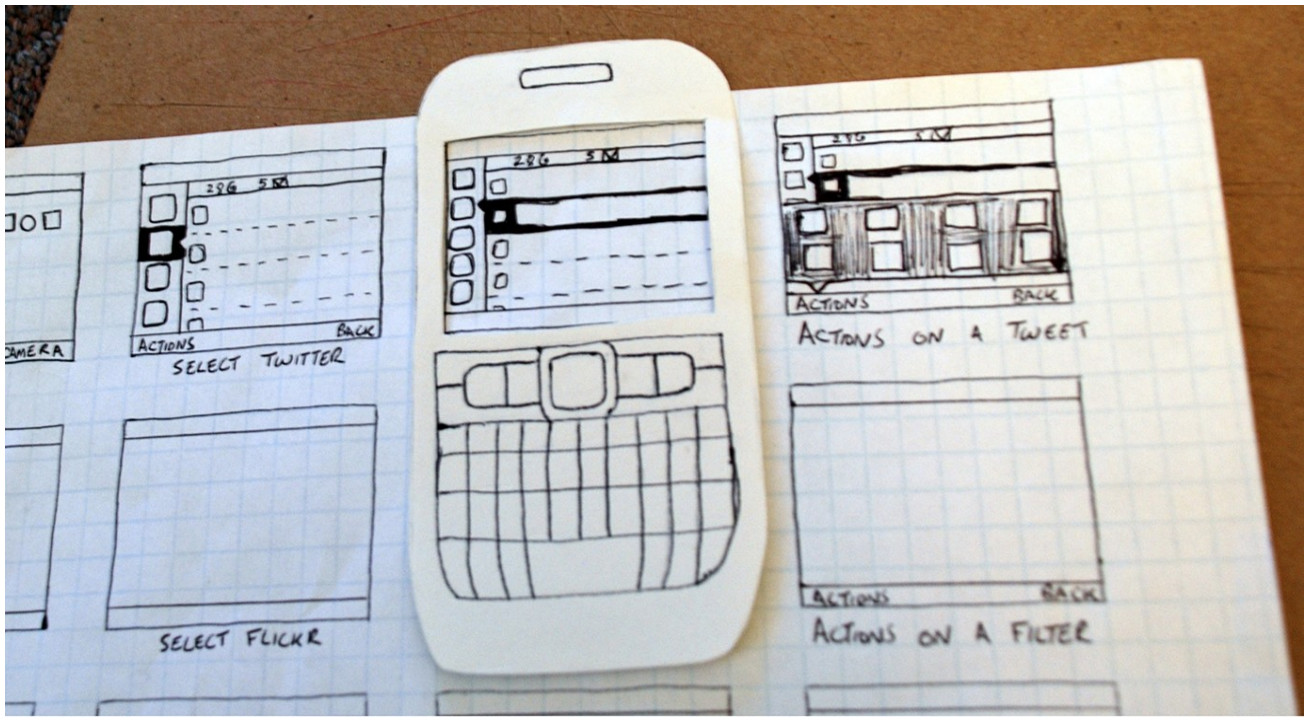
- **Evolutivos**
  - Proveen una base arquitectónica solida para desarrollar el producto de forma incremental mientras los requisitos se vuelven más claro con el tiempo.
  - Las metodologías ágiles proveen un ejemplo de prototipación evolutiva. Los equipos ágiles construyen el producto a lo largo de una serie de iteraciones, utilizando feedback para ajustar los siguientes ciclos del desarrollo.
  - Deben ser construidos con calidad en el código y robustez desde el principio.
  - Además deben ser diseñados para contemplar un rápido crecimiento y una mejora frecuente.
  - Son una buena elección para aplicaciones que sabemos que crecerán a lo largo del tiempo (por ejemplo, sitios web).

Prototipos – Forma

- **Prototipos en papel**
  - Prototipos de baja fidelidad.
  - Permiten de forma barata, rápida y de baja tecnología explorar cómo va a lucir una parte del producto.
  - Involucran herramientas simples (papel, tarjetas, post-its).
  - La idea es explorar posibles alternativas de comportamiento y estética del producto sin perderse mucho en los detalles.
  - Facilitan una rápida interacción y se utilizan para refinar los requisitos antes de diseñar la interfaz de usuario.
  - Se pueden incluir, por ejemplo, bocetos de las pantallas en los casos de uso.
  - Ejemplo: storyboards.

Prototipos – en papel

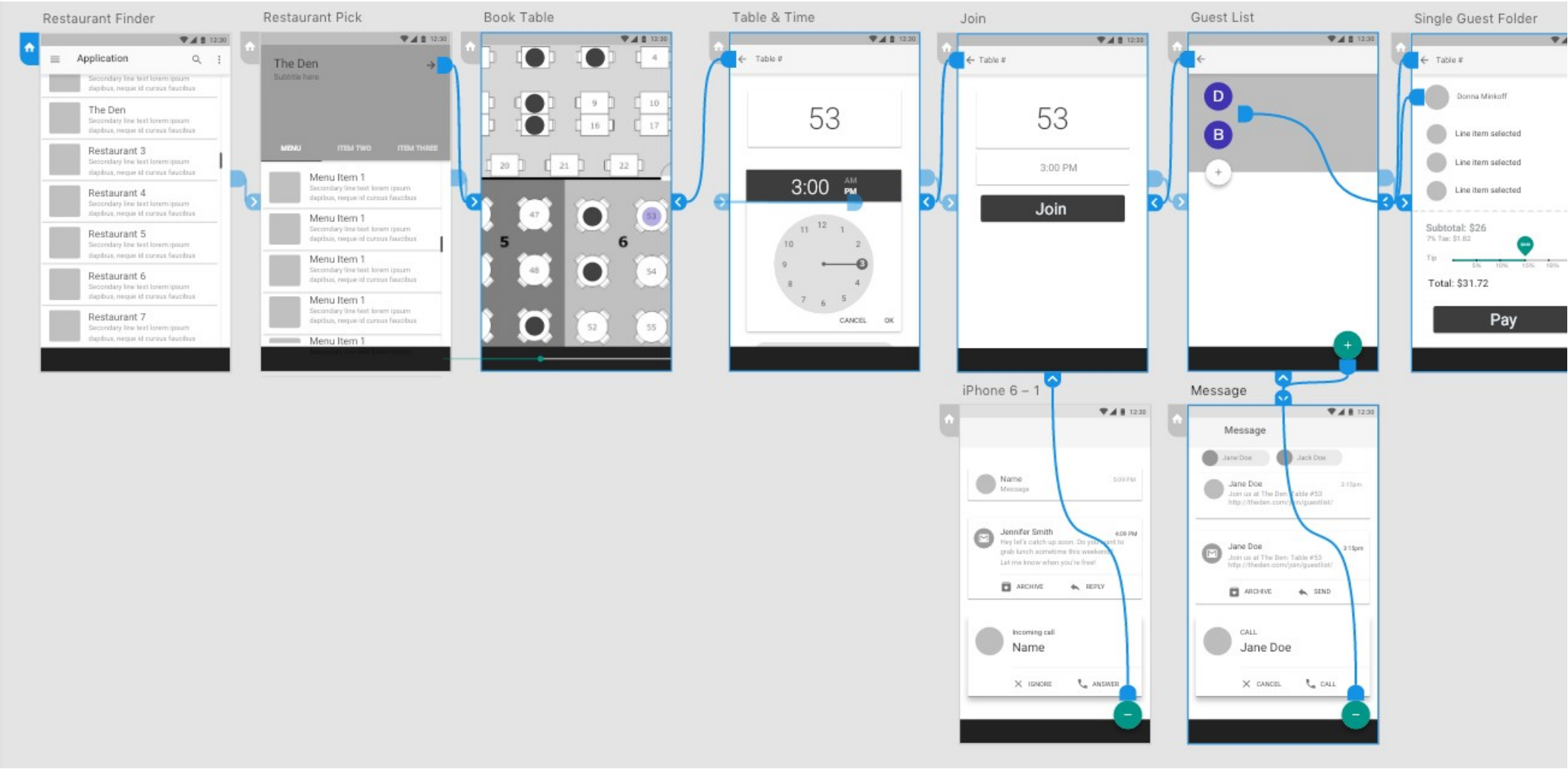




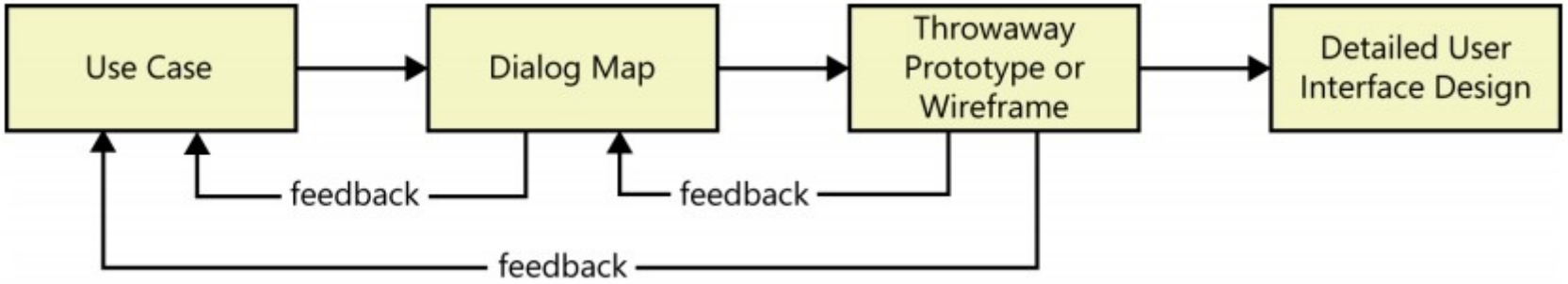
Prototipos – Forma

- **Prototipos electrónicos**
  - Prototipos de alta fidelidad.
  - Hay numerosas herramientas para realizarlos: desde Ms Visio y Ms Powerpoint a herramientas de prototipado y simulación.
  - La interacción con una versión muy similar en su estética o comportamiento provee un mecanismo muy valioso para clarificar los requisitos y estudiar el comportamiento de los usuarios.

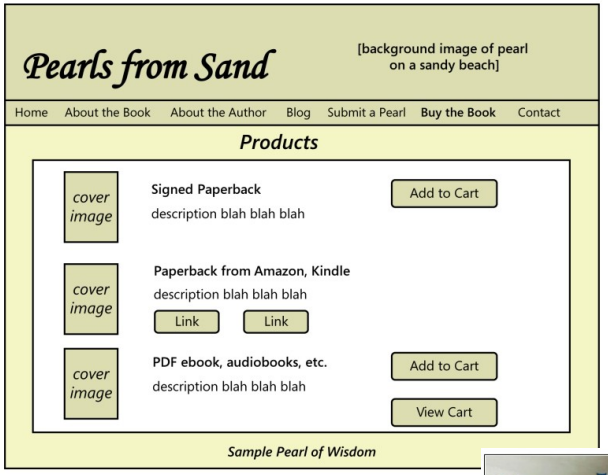
Prototipos – electrónicos

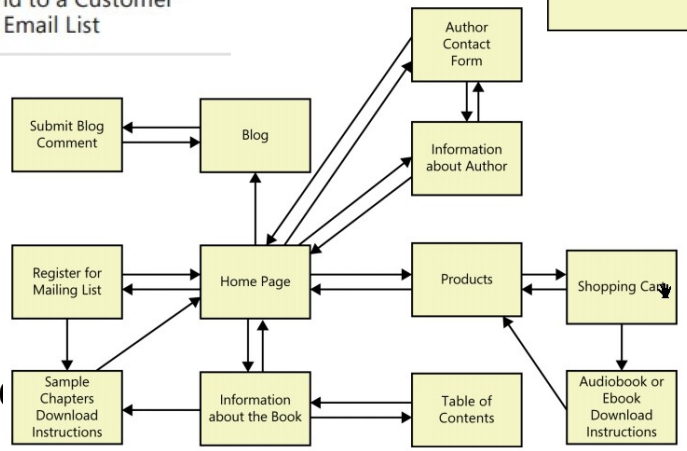


Prototipos – Un ejemplo de uso



User class	Use case
Visitor	Get Information about the Book Get Information about the Author Read Sample Chapters Read the Blog Contact the Author
Customer	Order a Product Download an Electronic Product Request Assistance with a Problem
Administrator	Manage the Product List Issue a Refund to a Customer Manage the Email List





Prototipos – Evaluación

- La evaluación de los prototipos está relacionada a las pruebas de usabilidad. Se aprende más observando a los usuarios trabajando con el prototipo que preguntado qué piensan de él.
- Hay que validar con los prototipos con un público objetivo adecuado.
- Para mejorar la evaluación se puede utilizar guías, por ejemplo indicando que realicen una serie de operaciones específicas.
- Hay que evitar guiar demasiado e indicar la forma “correcta” de usar el prototipo.
- Documentar todo lo ocurrido en la validación para futuras actividades.

Prototipos – Riesgos

- Que el cliente piense que el producto ya está pronto. O que quiera una versión para probarlo fuera de la instancia de validación.
- Perderse en los detalles.
- Generar expectativas irreales con respecto al producto final. El prototipo es más simple y seguramente tiene menos lógica o menos detalles.
- Invertir demasiado esfuerzo en los prototipos.

Prototipos – Factores de éxito

- Incluirlos en el plan de trabajo.
- Establecer los objetivos previo a su construcción.
- Planificar la construcción de varios. A veces es necesario hacer más versiones de un prototipo.
- Crear prototipos desechables tan rápido y barato como sea posible.
- No incluir muchos detalles como validaciones, manejo de errores en prototipos desechables.
- No prototipar requisitos que no se entendieron todavía.
- Usar datos cercanos a la realidad.
- No esperar realizar prototipos en lugar de escribir requisitos.