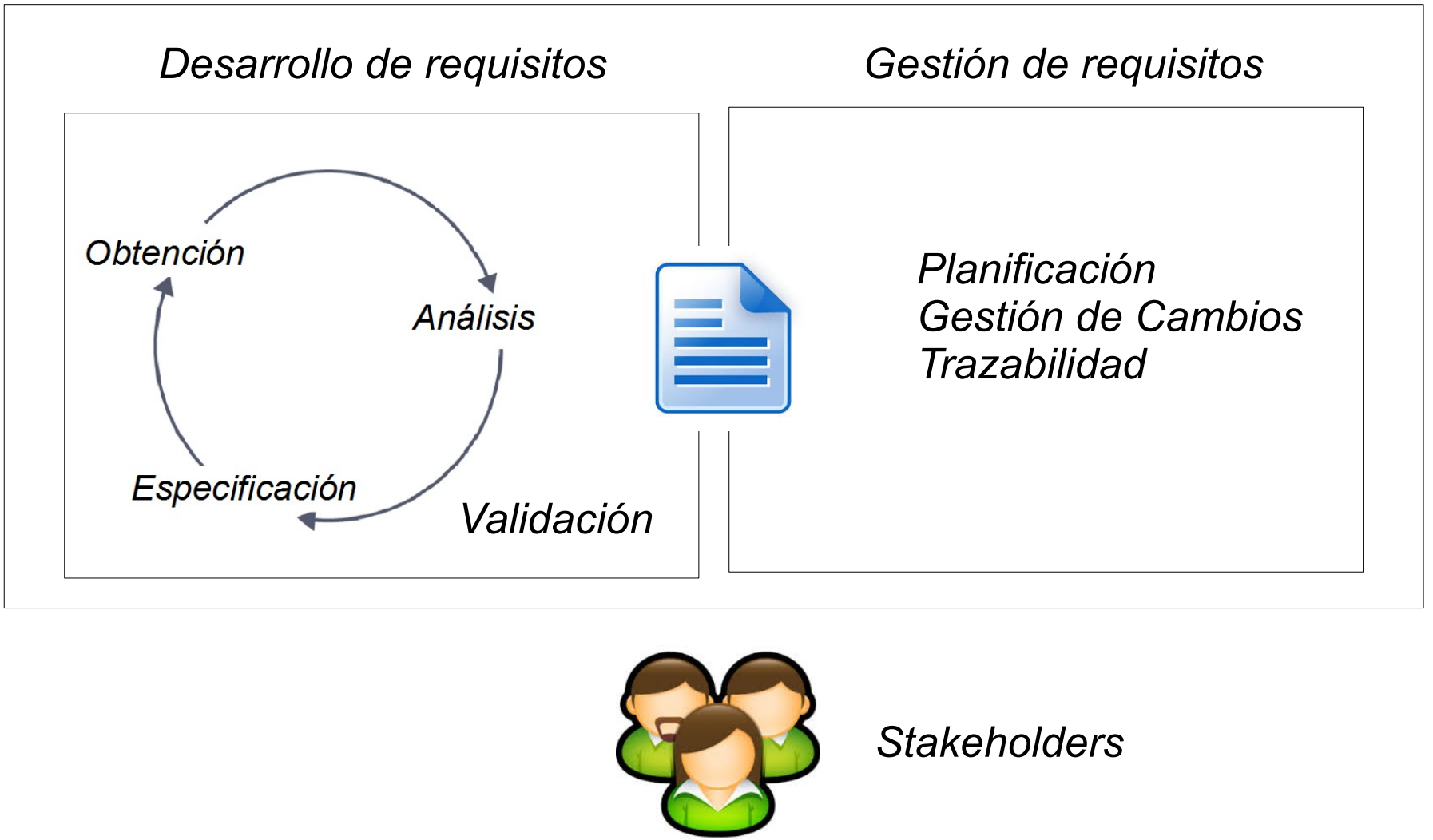
FACULTAD DE INGENIERIA

Ingeniería de Software

Ingeniería de Requisitos
Clase 2

Sommerville capítulo 4 – sección 4.5
Wiegers capítulos 7 y 15

Actividades de la ingeniería de requisitos



Obtención de requisitos y análisis

- 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.

Observaciones

- Implican observar a los usuarios mientras realizan sus actividades. Pueden ser silenciosas o interactivas.
- En general los usuarios no son muy precisos al describir sus tareas. Esto puede ser porque son tareas complejas y difíciles de explicar. En otros casos, están tan familiarizados que omiten detalles de forma inconsciente.

- **Ventajas**
- Confiable
- Muy rico
- Desarrolla empatía

- **Desventajas**
- Muy costosas
- Efecto Hawthorne
- Hay que tener cuidado en generalizar (sesgo particular/local)

Etnografías

- Una etnografía es una técnica observacional.
  - Las personas no tienen que explicar o articular su trabajo.
  - Se pueden observar los factores sociales y organizacionales de importancia.
- Los estudios etnográficos han demostrado que el trabajo suele ser más rico y complejo que lo que sugieren los modelos de sistemas simples.
- Los requisitos son derivados de la forma en la cual realmente trabajan las personas en lugar de las definiciones de procesos que indican cómo se debe trabajar.
- La etnografía es efectiva para entender proceso existentes pero no para identificar nuevas características que deben agregarse al sistema.

Cuestionarios / Encuestas

- Son una manera de estudiar a grandes grupos de usuarios para entender sus necesidades.
- Antes de usar el enfoque:
  - Determinar la información que se precisa.
  - Determinar el enfoque más adecuado:
    - Abierto, cerrado, combinado.
    - Múltiple opción, valor en escala, orden relativo.
  - Desarrollar cuestionario.
  - Probarlo con perfil típico.
  - Analizar resultado de las pruebas.
- Su principal uso es para validar y obtener datos estadísticos sobre preferencias.

Cuestionarios / Encuestas

- Preparar preguntas bien escritas es el mayor desafío.
- Recomendaciones
  - Proveer opciones para todas las posibles respuestas.
  - Hacer que las opciones sean mutuoexcluyentes.
  - Utilizar preguntas cerradas para análisis estadístico y abiertas para recolectar ideas o necesidades nuevas.
  - Siempre probar el cuestionario antes de usarlo.
  - No incluir demasiadas preguntas.

- **Ventajas**
- Economía de escala.
- Conveniente para quien contesta.
- Respuestas anónimas.

- **Desventajas**
- Menos rico.
- Problemas por no-respuesta.
- Esfuerzo de desarrollo.

Análisis de las interfaces del sistema

- Implica examinar los otros sistemas con los que se conecta el sistema.
- Revela requisitos funcionales relativas al intercambio de datos y servicios entre sistemas.
- Para cada sistema que se deba comunicar con el nuestro se identifican las funcionalidades que nos puedan generar requisitos. Esos requisitos pueden describir los datos a pasar a otros sistemas, los datos a recibir de otros sistemas y las reglas sobre los datos (por ejemplo criterios de validación).
- Es útil para revisar las validaciones de la información a comunica o recibir (quizás, por ejemplo, se puedan omitir validaciones)

Análisis de la interfaz de usuario

- Implica estudiar sistemas existentes para determinar requisitos de usuario y funcionales.
- Los mejor es utilizar sistemas existentes, pero sino se pueden utilizar screenshots (por ejemplo de manuales de usuario).
- Puede ayudar a identificar una lista completa de pantallas y a descubrir características potenciales del nuevo sistema. Se utiliza para identificar pasos comunes de los usuarios al realizar tareas así como para crear borradores de casos de uso.
- No hay que asumir que una funcionalidad es necesaria porque se encuentra en un sistema existente. O mantener la interfaz de usuario parecida o que se comporte de forma similar a la estudiada.

Análisis de documentación

- Contempla examinar toda la documentación existente en busca de requisitos potenciales del software.
- La documentación más útil incluye especificación de requisitos, procesos de negocio, lecciones aprendidas y manuales de usuario de aplicaciones existentes o similares.
- Es a una forma rápida de introducirse rápidamente en un nuevo dominio o en un sistema existente.

Tormenta de ideas

- Ayuda a la participación de todos los involucrados.
- Reglas: no se permite criticar ni debatir, generar tantas ideas como sea posible, mutar y combinar ideas.
- **Fase de generación**
  - Los principales stakeholders se reúnen y se establecen objetivos.
  - Se pide que cada uno escriba sus ideas.
- **Fase de reducción**
  - Se leen las ideas y se establece si es válida.
  - Se agrupan ideas y se hacen las definiciones necesarias.
  - Se priorizan (opcional).

Escenarios

- Los escenarios son ejemplos de la vida real donde se especifica como el sistema debe ser usado.
- Debe incluir:
  - Una descripción de la situación de partida.
  - Una descripción del flujo normal de los eventos
  - Una descripción de lo que puede salir mal.
  - Información sobre otras actividades concurrentes.
  - Una descripción del estado cuando finaliza el escenario.

Escenarios - ejemplo

INITIAL ASSUMPTION:
The patient has seen a medical receptionist who has created a record in the system and collected the patient's personal information (name, address, age, etc.). A nurse is logged on to the system and is collecting medical history.

NORMAL:
The nurse searches for the patient by family name. If there is more than one patient with the same surname, the given name (first name in English) and date of birth are used to identify the patient.

The nurse chooses the menu option to add medical history.

The nurse then follows a series of prompts from the system to enter information about consultations elsewhere on mental health problems (free text input), existing medical conditions (nurse selects conditions from menu), medication currently taken (selected from menu), allergies (free text), and home life (form).

WHAT CAN GO WRONG:
The patient's record does not exist or cannot be found. The nurse should create a new record and record personal information.

Patient conditions or medication are not entered in the menu. The nurse should choose the 'other' option and enter free text describing the condition/medication.

Patient cannot/will not provide information on medical history. The nurse should enter free text recording the patient's inability/unwillingness to provide information. The system should print the standard exclusion form stating that the lack of information may mean that treatment will be limited or delayed. This should be signed and handed to the patient.

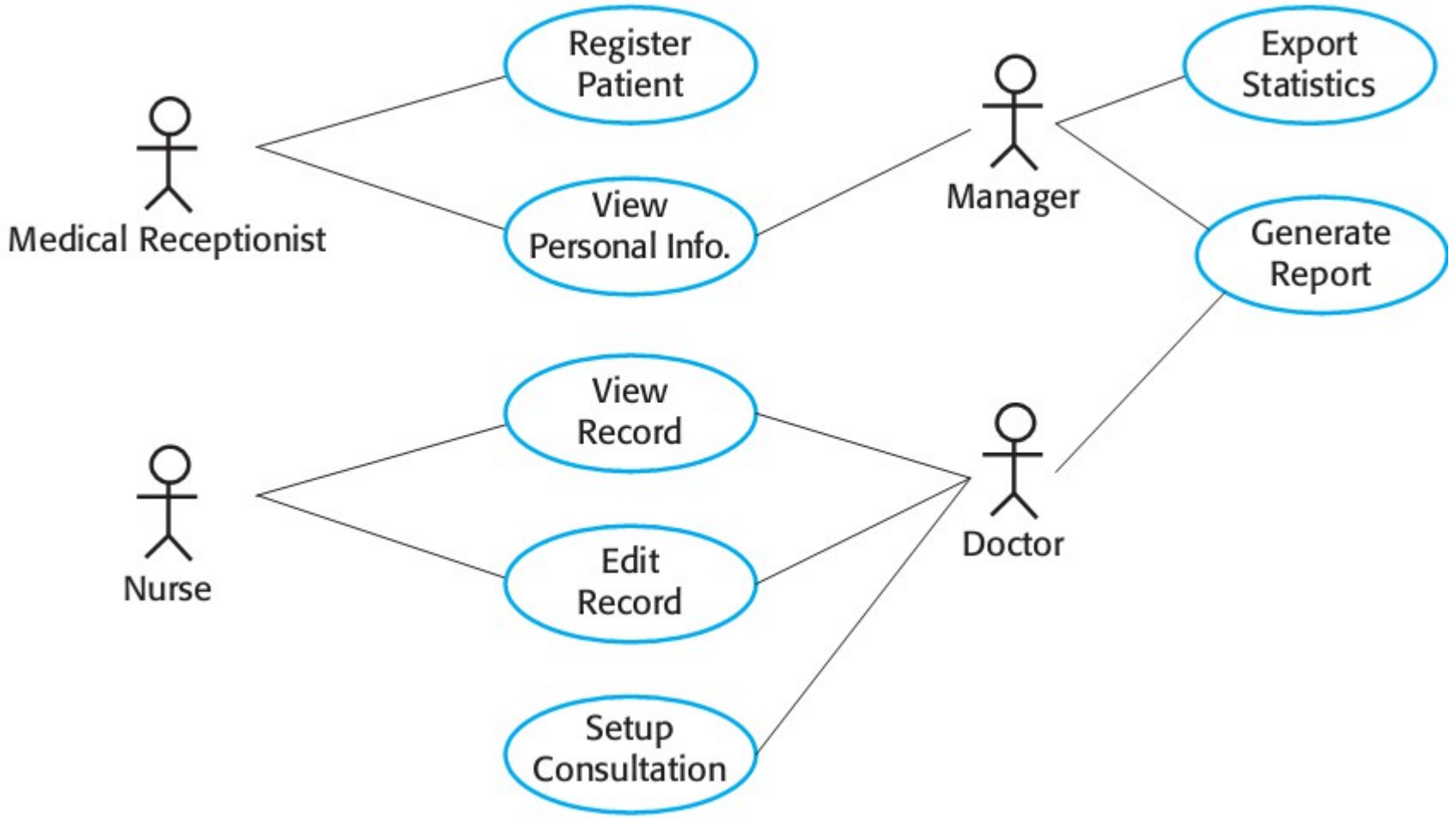
OTHER ACTIVITIES:
Record may be consulted but not edited by other staff while information is being entered.

SYSTEM STATE ON COMPLETION:
User is logged on. The patient record including medical history is entered in the database, a record is added to the system log showing the start and end time of the session and the nurse involved.

Casos de uso

- Los casos de uso son una técnica incluida en UML que se basa en el concepto de los escenarios.
- Un conjunto de casos de uso debe describir todas las posibles interacciones con el sistema. Se identifica cada tipo de interacción entre los distintos actores y el sistema y se les da un nombre.
- Constan de un modelo gráfico de alto nivel complementado con una descripción más detallada de cada caso de uso.

Casos de uso


Register Patient
View Personal Info.
Medical Receptionist
Manager
Export Statistics
Generate Report
View Record
Edit Record
Setup Consultation
Nurse
Doctor

Casos de uso

- Formato simple y estructurado donde los usuarios y desarrolladores pueden trabajar juntos.
- No son de gran ayuda para identificar aspectos no funcionales.
- Mientras se definen los casos de uso, puede ser un buen momento para definir pantallas u otros objetos con los que el usuario interactúa.
- Pueden ser usados en el diseño y en el testing del sistema.

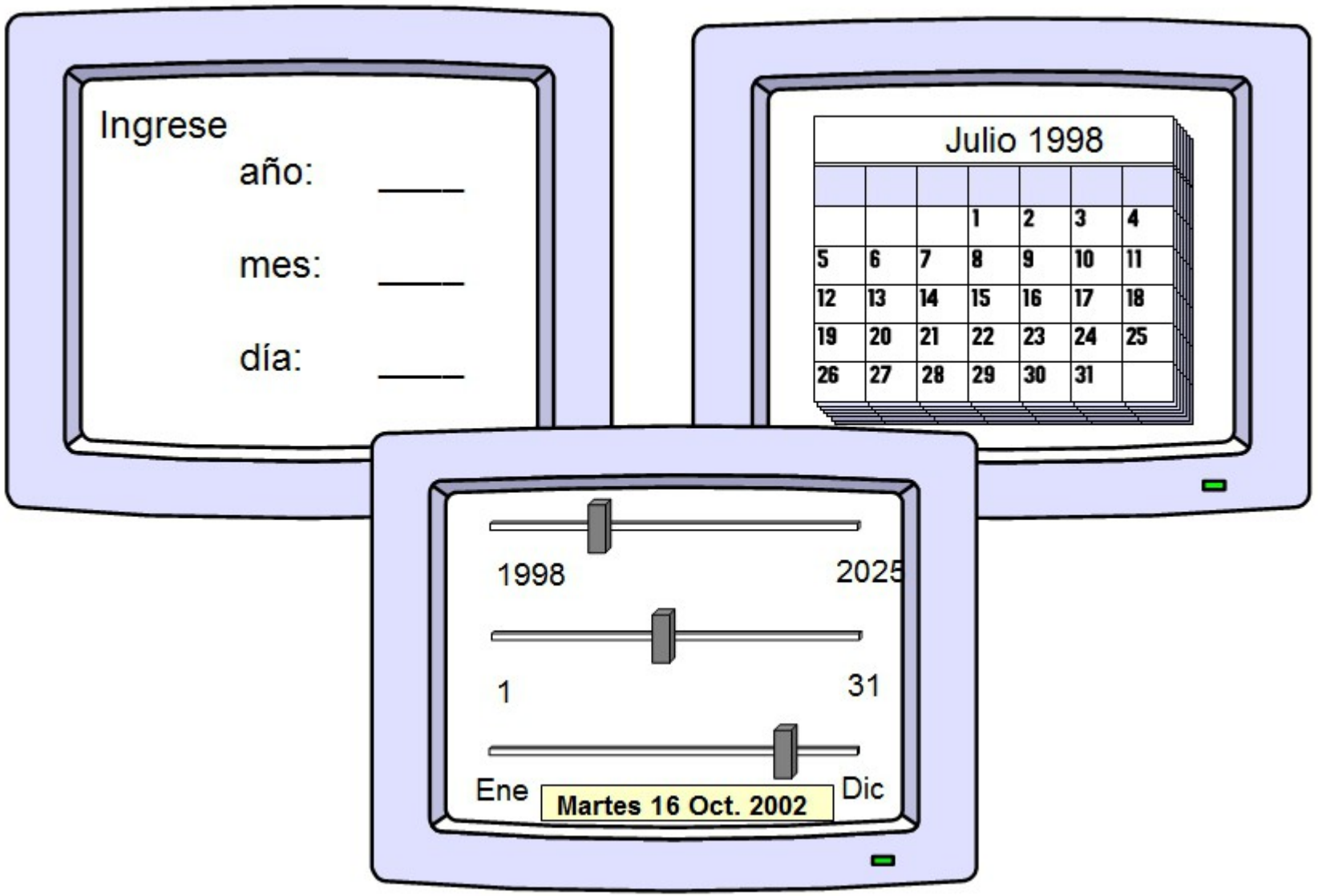
- **Ventajas**
- Permiten ver fácilmente la interacción del sistema con los demás actores (usuarios u otros sistemas)
- Muy útiles cuando la implementación se va a hacer OO y con UML.

- **Desventajas**
- No son demasiado útiles para describir sistemas sin usuarios y/o con pocas interfaces.
- No modelan bien requisitos no funcionales y restricciones de diseño.

Prototipado

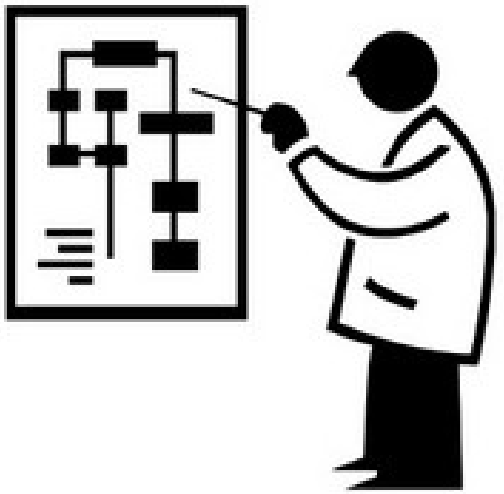
- Implementación parcial, permite a los desarrolladores y usuarios:
  - Entender mejor los requisitos.
  - Cuáles son necesarios, deseables.
  - Acotar riesgos.
- Aspectos para los que es frecuente construir prototipos:
  - Apariencia y percepción de la interfaz de usuario
  - Arquitectura (riesgos tecnológicos,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

Prototipado

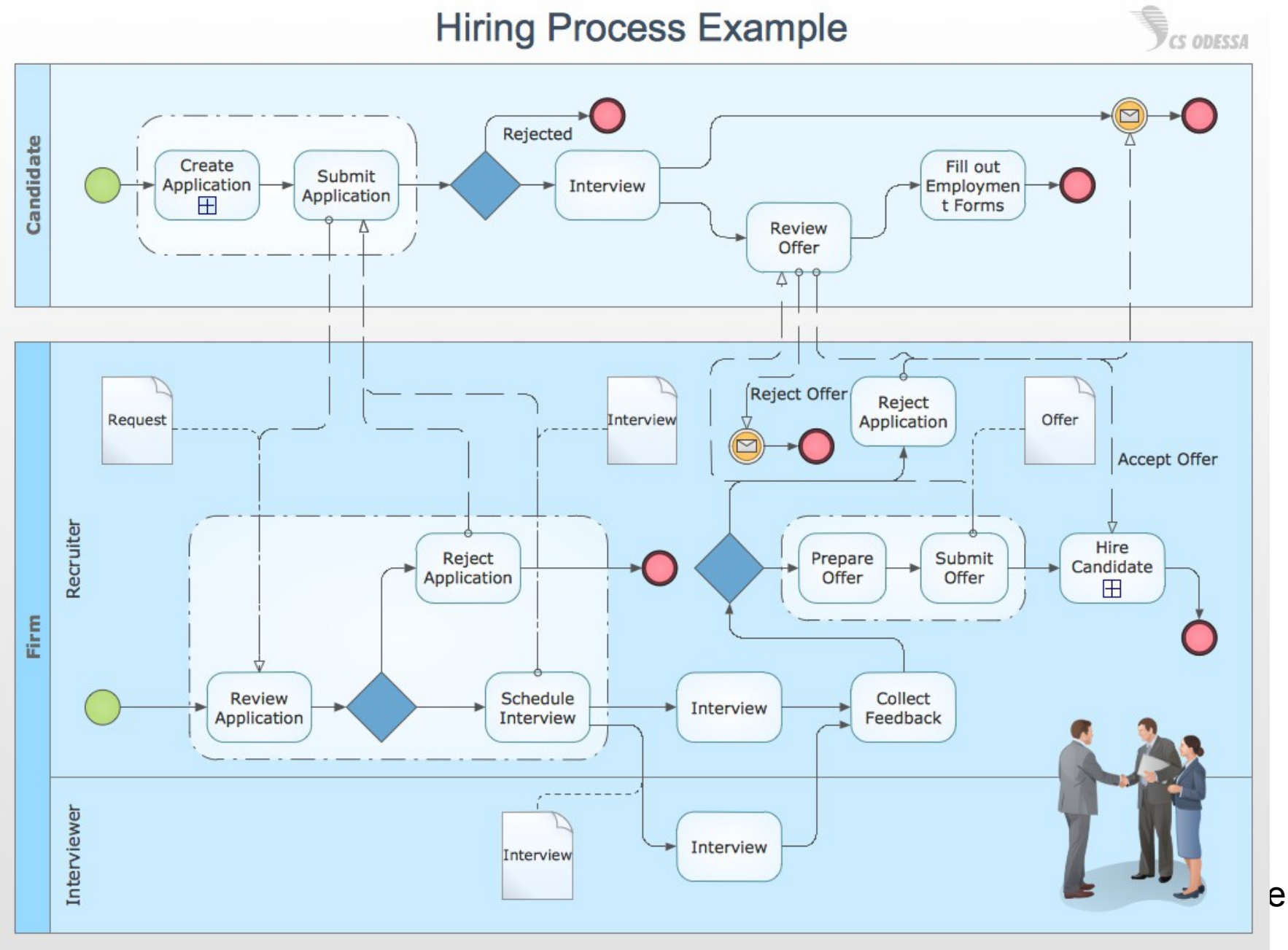


Modelado de Procesos

- Permiten entender el trabajo con múltiples pasos, roles o departamentos.
  - Iniciados por un evento.
  - Incluyen actividades manuales, automáticas o combinaciones de ambos tipos.
- Pueden volverse complejos y pesados si no se estructuran con cuidado.
- Los procesos complejos se pueden descomponer para ayudar su entendimiento.



Modelado de Procesos - Ejemplo



Historias de usuario

- Refieren a descripciones cortas y de alto nivel de las funcionalidades expresadas en los términos del cliente.
- Una historia de usuario típica tiene la forma:
  - *"As a <role>, I want <goal/desire> so that <benefit>."*
- Pretenden contener justo la información necesaria para que los desarrolladores puedan producir una estimación razonable del esfuerzo para su implementación.
- La idea es evitar perder demasiado tiempo relevando detalles de requisitos que luego cambian demasiado o son desestimados.

Historias de usuario - ejemplo

ID	Theme	As a/an	I want to...	so that...	Notes	Priority	Status
2	Game	moderator	create a new game by entering a name and an optional description	I can start inviting estimators	If games cannot be saved and returned to, the description is unnecessary	Required	done
2	Game	moderator	invite estimators by giving them a url where they can access the game	we can start the game	The url should be formatted so that it's easy to give it by phone.		done
5	Game	estimator	join a game by entering my name on the page I received the url for	I can participate			done
6	Game	moderator	start a round by entering an item in a single multi-line text field	we can estimate it			done
8	Game	estimator	see the item we're estimating	I know what I'm giving an estimate for			done
~~40~~	~~Game~~	~~participant~~	~~always have the cards in the same order across multiple draws~~	~~it's easy to compare estimates~~	-	Replaced with A08 because I didn't want the story to talk about "the same order" as that might be a UI implementation detail	~~todo~~
35	Non-functional	user	have the application respond quickly to my actions	I don't get bored			done
36	Non-functional	user	have nice error pages when something goes wrong	I can trust the system and it's developers			done
A11	Non-functional	Researcher	results to be stored in a non-identifiable way	I can study the data to see things like whether estimates converged around the first opinion given by "estimator A" for example	No names or story text should be stored but we should store each card of each hand, know who played it, and know the final accepted estimate		
A05	Game	moderator	edit an item in the list of items to be estimated	so that I can make it better reflect the team's understanding of the item			
22	Archive	moderator	export a transcript of a game as a CSV file	I can further process the stories and estimates	Exported file should be directly importable back into the system.		done

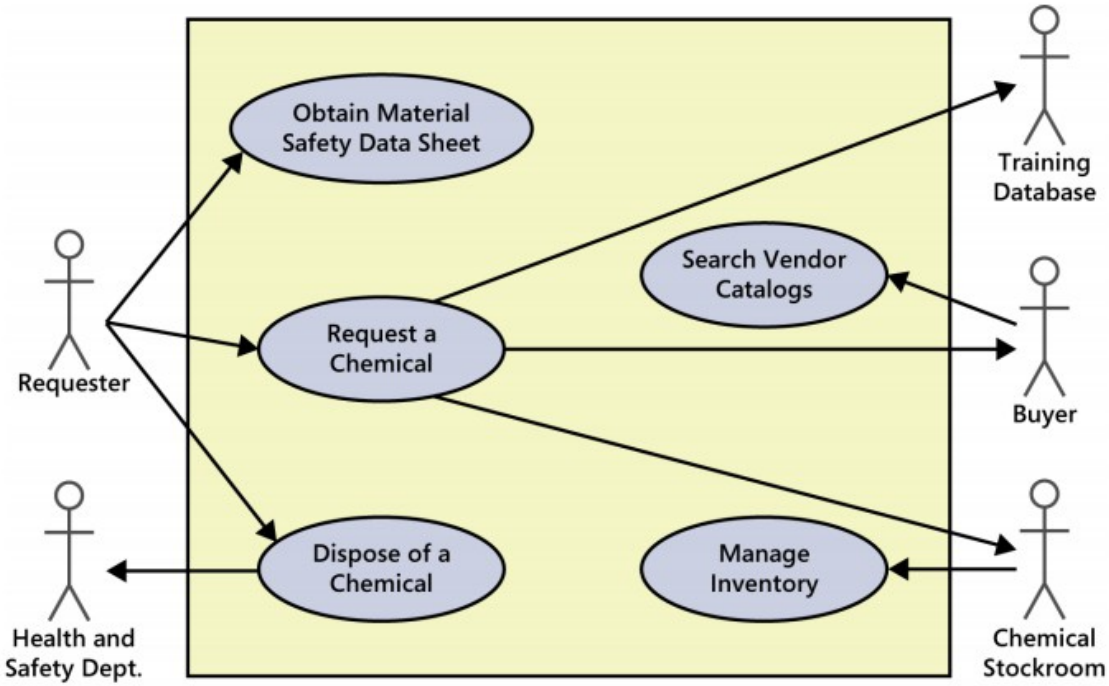
Selección de técnicas a utilizar

- En general es necesario utilizar más de una técnica y la elección está muy relacionada al tipo de proyecto.
- Wiegers hace algunas sugerencias de ejemplo (incluye sólo las técnicas presentadas en el libro):

	Interviews	Workshops	Focus groups	Observations	Questionnaires	System interface analysis	User interface analysis	Document analysis
Mass-market software	x		x		x			
Internal corporate software	x	x	x	x		x		x
Replacing existing system	x	x		x		x	x	x
Enhancing existing system	x	x				x	x	x
New application	x	x				x		
Packaged software implementation	x	x		x		x		x
Embedded systems	x	x				x		x
Geographically distributed stakeholders	x	x			x			

Análisis de requisitos

- El análisis se da naturalmente junto a las actividades de obtener y especificar los requisitos, conceptualmente implica:
  - Detectar y resolver conflictos entre los requisitos.
  - Descubrir las fronteras del software y cómo debe interactuar con los ambientes organizacional y operacional.
  - Elaborar los requisitos del sistema para derivar los requisitos del software.



Análisis de requisitos

- Las actividades incluyen:
  - **Clasificación de requisitos**
    - Utilizando, por ejemplo, las categorías que vimos la clase pasada.
    - Prioridad, alcance y estabilidad de los requisitos.
  - **Modelado conceptual**
    - Realizar modelos resulta clave para el análisis de requisitos. Ayudan a entender la situación en la cual ocurren los problemas o necesidades, así como representar soluciones.
  - **Diseño de la arquitectura y asignación de requisitos**
  - **Negociación de requisitos**
    - Resolución de conflictos. Priorización.
  - **Análisis Formal**
    - Muy recomendado para ciertos dominios de aplicación.

Prototipado — Técnica en detalle

- Las personas tienen dificultad para describir sus necesidades sin tener nada tangible adelante de ellos; 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

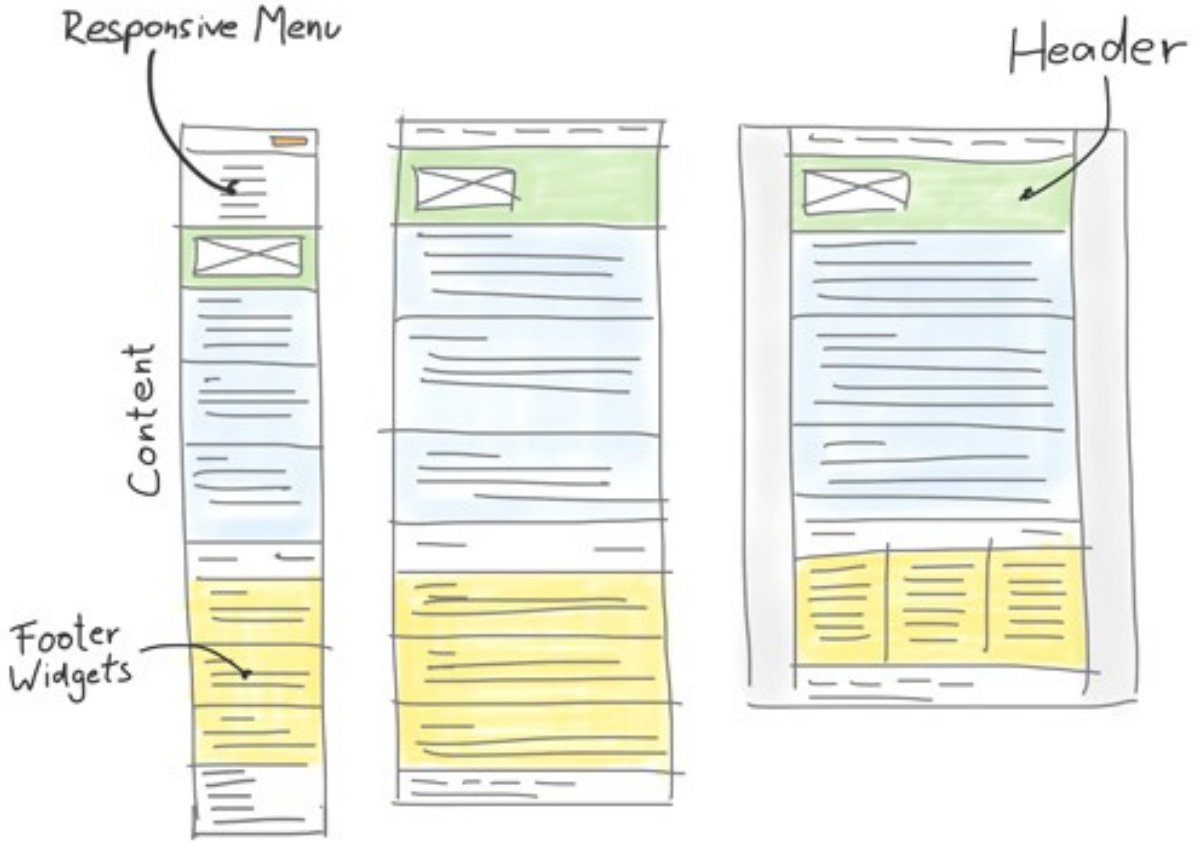
- **Mock-ups – Bosquejos**
  - Prototipos horizontales.
  - Se enfocan en porciones de la interfaz de usuario.
  - Permiten explorar comportamientos específicos del producto.
  - No realizan ningún trabajo útil, sólo lucen como si lo hicieran.

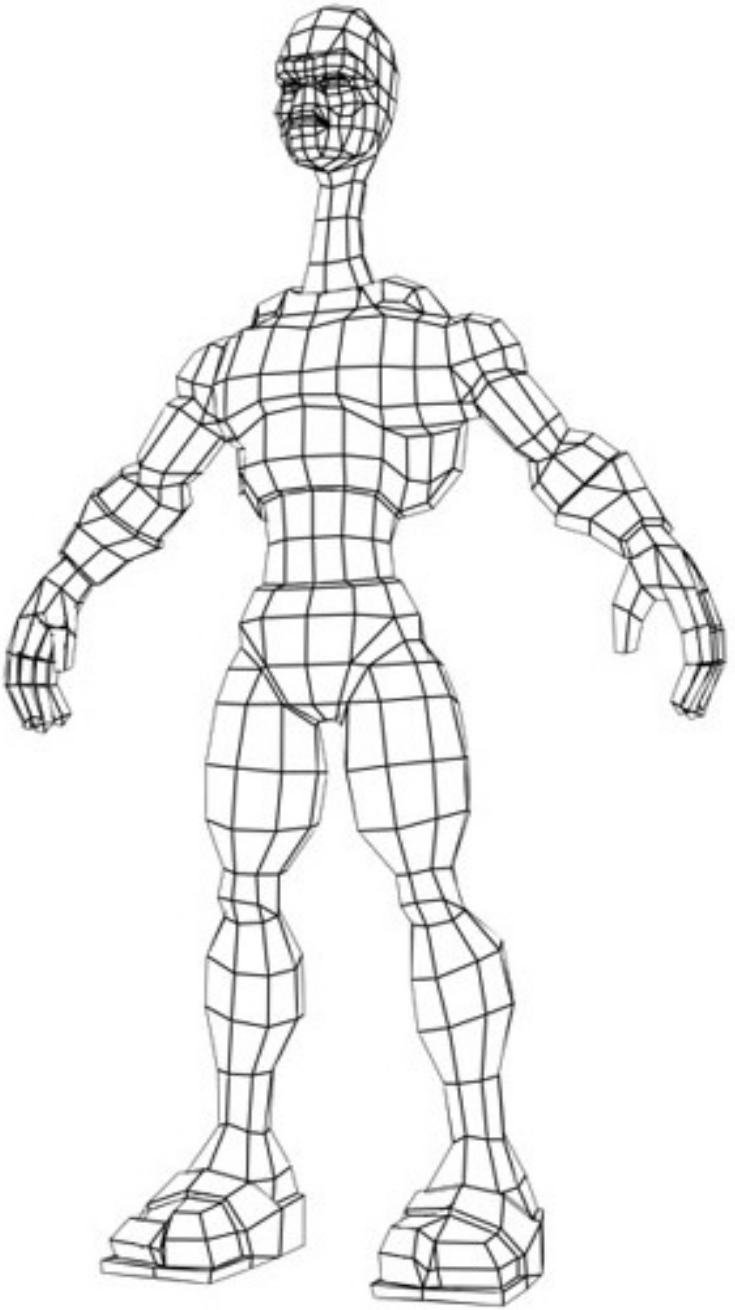
- **Pruebas de concepto**
  - Prototipos verticales.
  - Implementan una porción funcional de la aplicación.
  - Permiten resolver incertidumbres sobre la factibilidad de la arquitectura propuesta u otros riesgos técnicos.
  - Funcionan como el sistema real porque toca todos los niveles de la implementación.

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





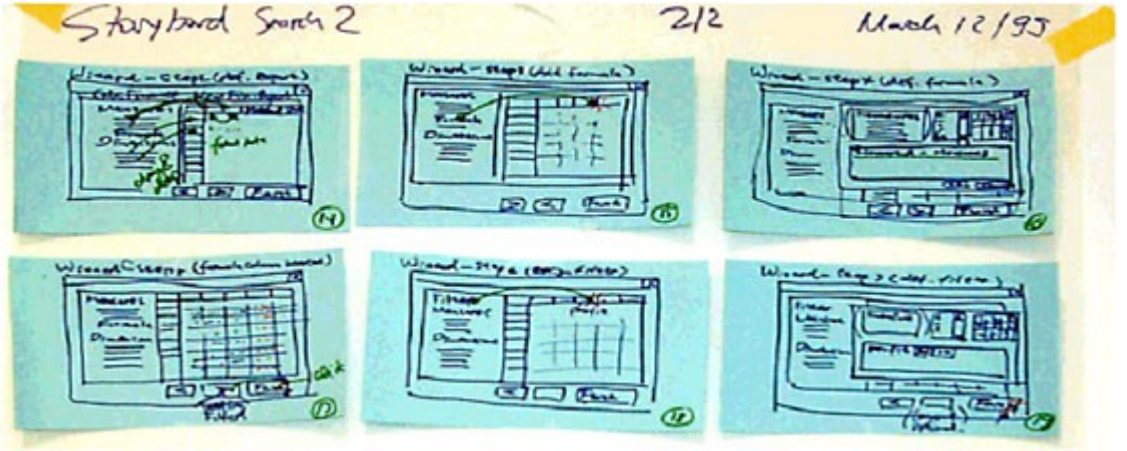
Prototipos – Uso futuro

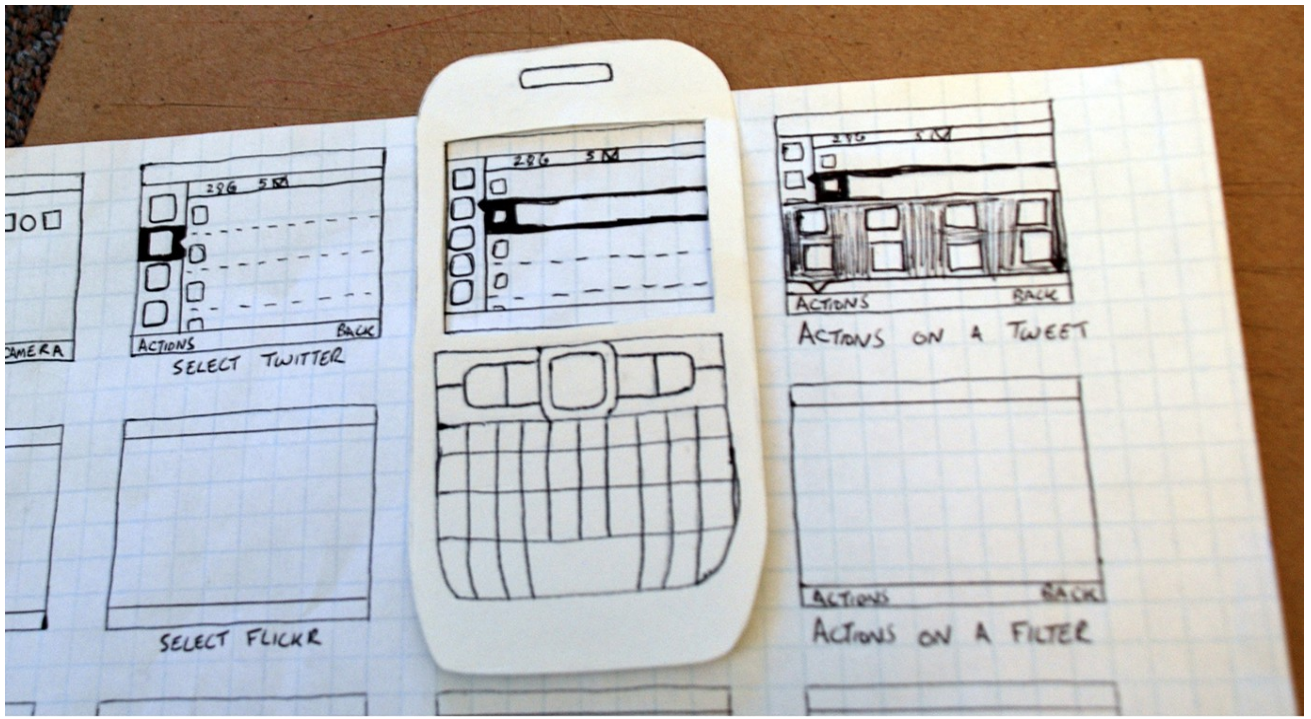
- **Evolutivos**
  - Proveen una base arquitectónica solida para desarrollar el producto de forma incremental mientras los requisitos se vuelven más claro con el tiempo.
  - Las metodologías ágiles proveen un ejemplo de prototipación evolutiva. Los equipos ágiles construyen el producto a lo largo de 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

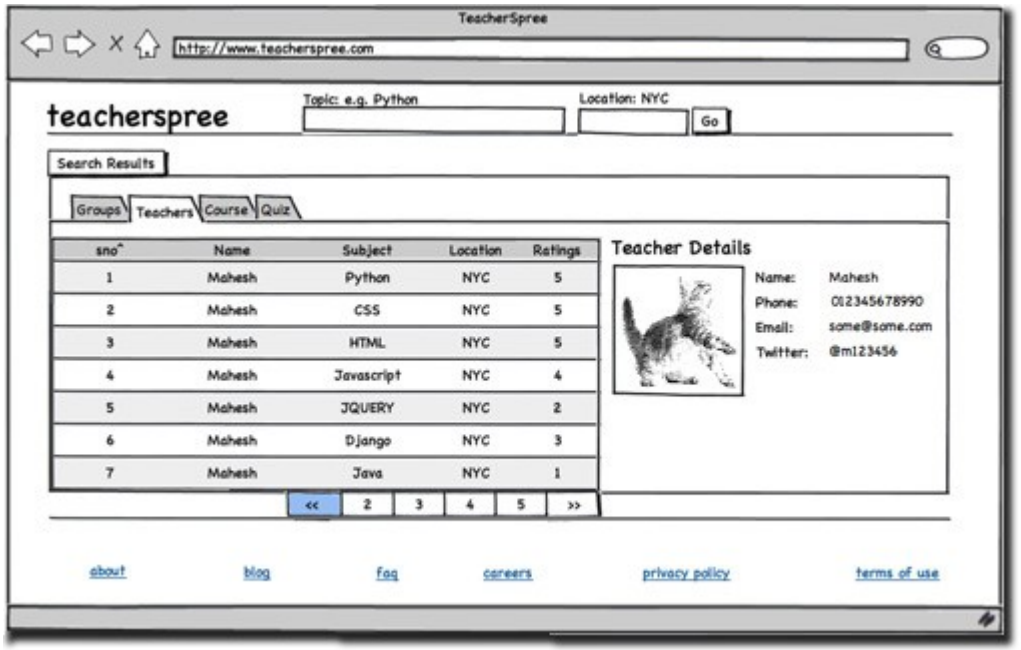


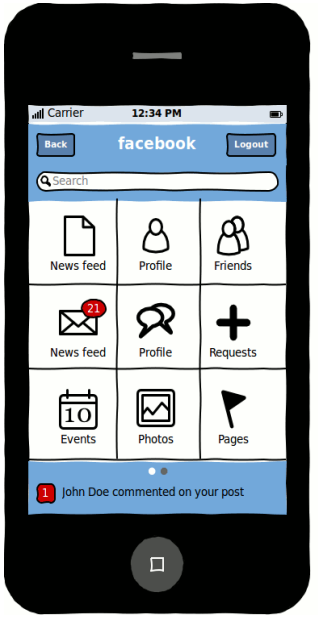


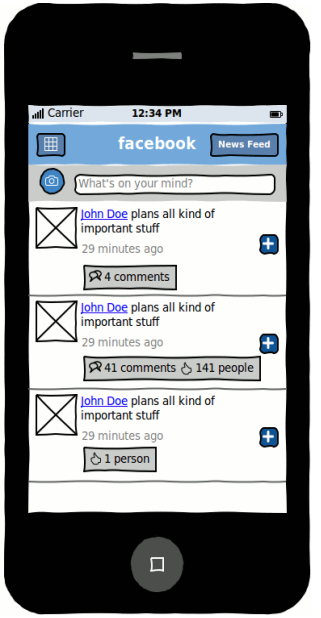
Prototipos – Forma

- **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.

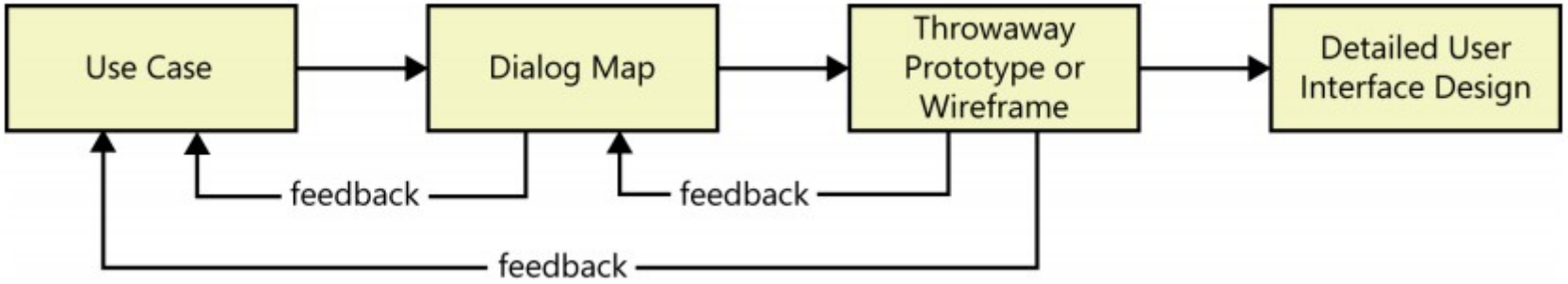
Prototipos – electrónicos



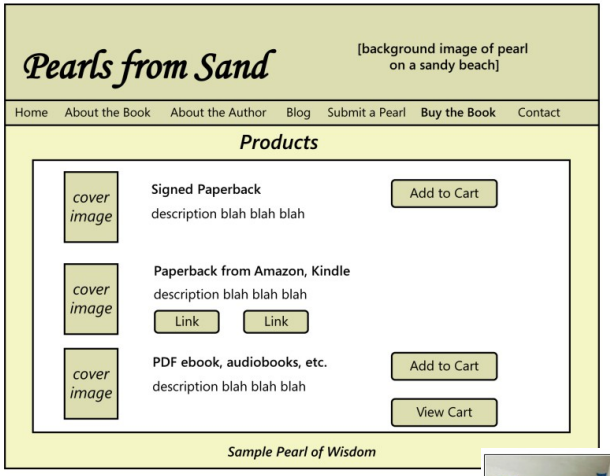


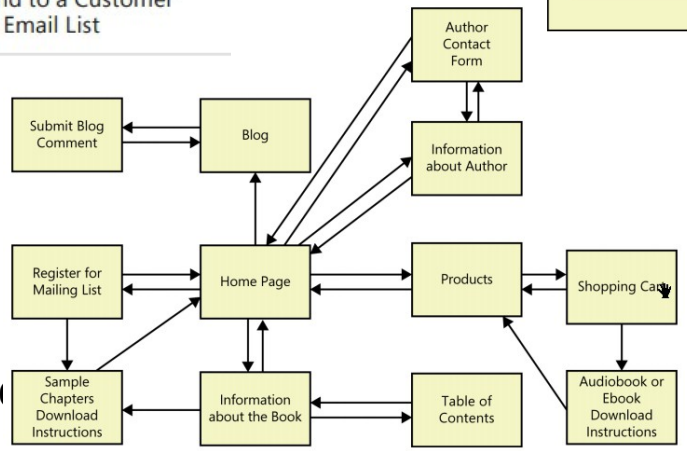


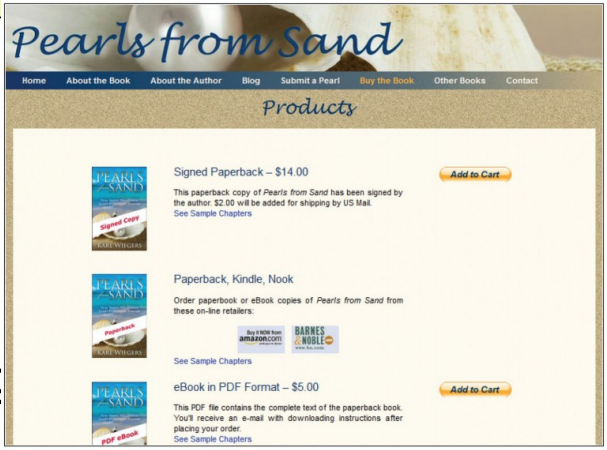
Prototipos – Un ejemplo de uso



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.

Ingeniería de Software

Próxima clase: Casos de uso
y modelado del sistema

Qué tengo que hacer?
Leer Sommerville capítulo 5
Leer Wiegers capítulos 8 y 12