Ingeniería de Software I
Requisitos de Software

Jocelyn Simmonds

Departamento de Ciencias de la Computación

Temario

- Requerimientos
  - Tipos de Requerimientos
- Análisis de Requerimientos de Software
- Gestión de Requerimientos
- Objetivos del Sistema, Proyecto, Negocio
- Objetivos v/s Requerimientos
- Desafíos en el proceso de elicitación

Requerimientos

- ¿Qué es un "requerimiento" (o "requisito")?
  - Algo que el sistema debe ser capaz de hacer para cumplir su propósito

Requerimientos

- ¿Qué es un "requerimiento" (o "requisito")?
  - Algo que el sistema debe ser capaz de hacer para cumplir su propósito
- Tipos de Requerimientos
  - Requerimientos Funcionales: Tareas que el sistema debe poder realizar
  - Requerimientos No Funcionales: Propiedades que el sistema debe cumplir

Requerimientos

- ¿Qué es un "requerimiento" (o "requisito")?
  - Algo que el sistema debe ser capaz de hacer para cumplir su propósito
- Tipos de Requerimientos
  - Requerimientos Funcionales: Tareas que el sistema debe poder realizar
  - Requerimientos No Funcionales: Propiedades que el sistema debe cumplir
- ¿Cómo distinguirlos?
  - Funcionales: Verbos (X-ear)
    - Listar, Imprimir, Buscar, Ingresar, etc.
  - No Funcionales: Adverbios (X-mente)
    - Eficientemente, establemente, integradamente

Ejemplos de Requisitos Funcionales

- "El sistema requiere que los usuarios ingresen al sistema para que tengan acceso a toda la funcionalidad"

Ejemplos de Requisitos Funcionales

- "El sistema requiere que los usuarios ingresen al sistema para que tengan acceso a toda la funcionalidad"
- Requisitos pueden ser de alto o bajo nivel:
  - alto nivel: "El sistema procesará compras usando tarjetas de crédito"
  - bajo nivel: "El sistema validará que todas las contraseñas contengan letras mayúsculas y minúsculas, y al menos un número"
- Usualmente empezamos con requisitos a un alto nivel (requisitos de usuario)
  - y despues continuamos con los más detallados (requisitos de sistema)

Más ejemplos

- "Un usuario puede buscar un medico en el listado de los medicos de la clinica"
- "El sistema debe generar un listado de las citas de cada medico, todos los dias"
- "Cada empleado será identificado en el sistema usando un codigo unico"

Más ejemplos

- "Un usuario puede buscar un medico en el listado de los medicos de la clinica"
- "El sistema debe generar un listado de las citas de cada medico, todos los dias"
- "Cada empleado será identificado en el sistema usando un codigo unico"

Hay que tratar de evitar imprecisiones . . . ¿el usuario busca sobre la lista completa de los médicos? ¿o primero filtra por especialidad?

Checklist para Requisitos Funcionales

- Un requisito funcional debería ser testeable: necesitamos saber si el sistema cumple o no el requisito
- Sólo deberían especificar lo que sistema que están desarrollando debe hacer, no lo que podría hacer, ni lo que hace otra sistema en el mercado, etc.
- Cada requisito debería:
  - describir una sola cosa: hace mas fácil el testing y seguimiento
  - tener una fuente: ¿quién/qué decidió que el sistema debía hacer esto?
  - tener un identificador único, p.ej. "REQ-1"

Cuidado!

Los requisitos listan las cosas que el sistema hace, no como las hace (eso es parte del diseño).

Cuidado!

Los requisitos listan las cosas que el sistema hace, no como las hace (eso es parte del diseño).
¿Cual versión es "mejor"?

- "El sistema guardara los datos de los usuarios, incluyendo nombre, fnac, direccion y rut"
- "El sistema guardara los datos de los usuarios en una BD Oracle, incluyendo nombre, fnac, direccion y rut"

Cuidado!

Los requisitos listan las cosas que el sistema hace, no como las hace (eso es parte del diseño).
¿Cual versión es "mejor"?

- "El sistema guardara los datos de los usuarios, incluyendo nombre, fnac, direccion y rut"
- "El sistema guardara los datos de los usuarios en una BD Oracle, incluyendo nombre, fnac, direccion y rut"
  - ¿Es obligatorio usar Oracle? Al mejor si, al mejor no . . . pero es una decisión a nivel de implementación, no de análisis

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

1. El sistema validará y aceptará tarjetas de crédito y cheques. Alta prioridad.

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

1. El sistema validará y aceptará tarjetas de crédito y cheques. Alta prioridad.
    - Problema: dos requisitos en vez de uno

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

1. El sistema validará y aceptará tarjetas de crédito y cheques. Alta prioridad.
   - Problema: dos requisitos en vez de uno
   - ¿Qué pasa si el procesamiento de tarjetas de crédito funciona, pero no la validación de cheques? en ese caso, ¿el sistema cumple o no cumple con este requisito? No cumple, pero esto no siempre es claro en la practica.

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

1. El sistema validará y aceptará tarjetas de crédito y cheques. Alta prioridad.
   - Problema: dos requisitos en vez de uno
   - ¿Qué pasa si el procesamiento de tarjetas de crédito funciona, pero no la validación de cheques? en ese caso, ¿el sistema cumple o no cumple con este requisito? No cumple, pero esto no siempre es claro en la practica.
   - Al mejor solo el procesamiento de tarjetas de crédito es de alta prioridad

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

2. El sistema procesara los clic de mouse suficientemente rápido como para asegurar que los usuarios no tendrán que esperar.
3. El usuario debe tener instalado Adobe Acrobat

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

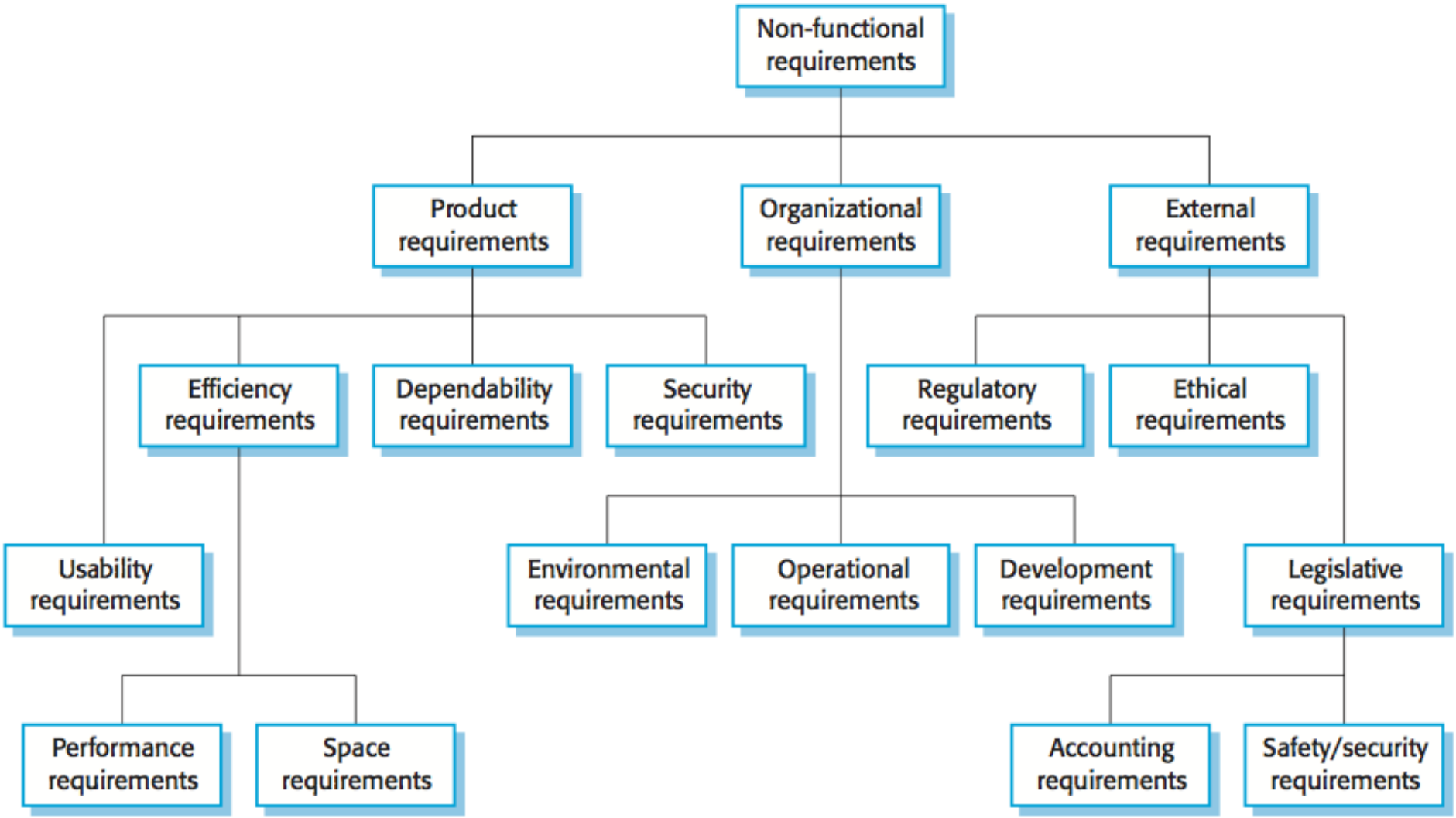
2. El sistema procesara los clic de mouse suficientemente rápido como para asegurar que los usuarios no tendrán que esperar.
   - Problema: esto no es testeable $\rightarrow$ ¿cuán rápido es aceptable?
3. El usuario debe tener instalado Adobe Acrobat

Ejemplos de requisitos malos

Explique porque los siguientes requisitos son malos requisitos:

2. El sistema procesara los clic de mouse suficientemente rápido como para asegurar que los usuarios no tendrán que esperar.
   - Problema: esto no es testeable $\rightarrow$ ¿cuán rápido es aceptable?
3. El usuario debe tener instalado Adobe Acrobat
   - Problema: esto no es algo que nuestro sistema debe hacer. Es un supuesto acerca de la configuración de PC del usuario, pero no es un requisito funcional del sistema.

Requisitos no funcionales



Requisitos no funcionales

Los requisitos no funcionales pueden afectar la arquitectura y al proceso

- tiempos de respuesta bajo de X segundos
- cierto nivel de disponibilidad
- ...

Requisitos no funcionales

Los requisitos no funcionales pueden afectar la arquitectura y al proceso

- tiempos de respuesta bajo de X segundos
- cierto nivel de disponibilidad
- ...

Ejemplo: El sistema debe estar disponible durante horas de oficina. Cualquier interrupción de servicio durante horas de oficina no puede más largo que 5 segundos por día.

Métricas para requisitos no funcionales

Propiedad	**Métrica**
Velocidad	transacciones por segundo
	tiempo de respuesta
Tamaño	MB
Facilidad de uso	tiempo de entrenamiento
	cantidad de paginas de ayuda
Reliability	Mean time to failure
	pbb de fallar
Robustez	tiempo necesario para reiniciar después de falla
	pbb de corrupción de datos tras falla
Portabilidad	% de lineas que dependen de arq.
	# de sistemas objetivo

Análisis de Requerimientos

La actividad de Análisis de Requerimientos (o Requisitos) de Software comprende:

1. levantamiento (o elicitación)
2. especificación
3. análisis
4. validación

Objetivo: generar un documento de requisitos, incluye req. de usuario y de sistema

¿Quienes leen este documento?

1. cliente: son los que pidieron el sistema, deben asegurarse de vamos a construir lo que ellos necesitan
2. jefe de proyecto: usan el doc para planificar el proyecto (presupuesto, recursos, etc.)
3. ingenieros: lo leen para entender que es lo que tienen que construir
4. QA: lo leen para desarrollar casos de prueba
5. . . . y más

¿Quienes leen este documento?

1. cliente: son los que pidieron el sistema, deben asegurarse de vamos a construir lo que ellos necesitan
2. jefe de proyecto: usan el doc para planificar el proyecto (presupuesto, recursos, etc.)
3. ingenieros: lo leen para entender que es lo que tienen que construir
4. QA: lo leen para desarrollar casos de prueba
5. . . . y más

Ahora volvamos al proceso de análisis de requisitos . . . elicitación, especificación, análisis y validación, en forma iterativa

1-Levantamiento (o Elicitación)

- Obtener los requerimientos
- Preguntar al cliente, usuarios u otros interesados cosas como:
  - ¿Qué se debe lograr?, ¿Cómo satisface el producto las necesidades?, ¿Cómo se utilizará el sistema o producto diariamente?, etc.
- Problemas asociados al levantamiento:

1-Levantamiento (o Elicitación)

- Obtener los requerimientos
- Preguntar al cliente, usuarios u otros interesados cosas como:
  - ¿Qué se debe lograr?, ¿Cómo satisface el producto las necesidades?, ¿Cómo se utilizará el sistema o producto diariamente?, etc.
- Problemas asociados al levantamiento:
  - Problemas de Ámbito: el límite del sistema está mal definido
  - Problemas de Comprensión: los clientes/usuarios no están seguros de qué es lo que necesitan, no comprenden del todo el dominio del problema, omiten información o especifican requerimientos ambiguos
  - Problemas de Volatilidad: problemas cambian conforme transcurre el tiempo

2-Especificación de Requerimientos

- Redacción, modelamiento
- Puede ser un documento escrito, un conjunto de modelos gráficos, plantillas, un modelo matemático formal, un prototipo o cualquier combinación de estos

3-Análisis de Requerimientos

- Revisarlos, resolver inconsistencias – contradicciones
- Mediante un enfoque iterativo, los requerimientos se eliminan, combinan o modifican de forma que cada parte alcance cierto grado de satisfacción

4-Validación de Requerimientos

- Visto bueno del cliente/usuario
- Lista de verificación para la Validación de Requerimientos:
  - ¿Son correctos?
  - ¿Son consistentes?
  - ¿Están completos?
  - ¿Son realistas?
  - ¿Son verificables?
  - ¿Describen una sola cosa (cada uno)?
  - ¿Son rastreables?

Gestión de Requerimientos

- Gestión de Requerimientos
  - Conjunto de actividades que ayudan al equipo de proyecto a identificar, controlar y rastrear los requerimientos y sus cambios en cualquier momento mientras se desarrolla el proyecto
- Tabla Genérica de Rastreabilidad

R\A	A_1	A_2	...	A_m
R_1		✓		✓
R_2	✓	✓		
R_3				✓
R_4	✓		✓	
...				
R_n			✓	

- **R**equisito
- **A**specto específico del sistema o su ambiente

Algunas observaciones

- El origen de los requerimientos en este caso es una entrevista, o reunión de trabajo
  - ¿Existirán más fuentes de información?

Algunas observaciones

- El origen de los requerimientos en este caso es una entrevista, o reunión de trabajo
  - ¿Existirán más fuentes de información?
- Existen declaraciones de distinto nivel de profundidad
  - El cliente hace afirmaciones tan puntuales como "el listado debe salir en 2 segundos"
  - Y otras tan genéricas como "Todo debe ser rápido pues es vital para el éxito del sistema"
- Ambos niveles de profundidad entregan información para nuestro Análisis de Requerimientos

Objetivos

Es clave distinguir:

- Objetivos del Negocio
- Objetivos del Proyecto
- Objetivos del Sistema

Objetivos del Negocio

- El software se desarrolla para satisfacer una necesidad, o resolver un problema
- Debemos llegar a comprender bien el fondo del problema desde el punto de vista de el Negocio, para reconocer requerimientos intrínsecos de él, que puede que el cliente no especifique o los de por sabidos
- En el ejemplo, la declaración del cliente "permite agilizar el proceso de despacho de tarjetas de navidad" nos da información acerca del objetivo de negocio
- Es un objetivo, ya que no es una afirmación tan detallada, es más de alto nivel

Objetivos del Proyecto

- Software se desarrolla en Proyectos de Software
- Hay características del proyecto que determinan ciertas propiedades del software
  - Incluso que sea usado o no
- En el ejemplo, la declaración del cliente "es importantísimo que la entrega final se cumpla de acuerdo a la planificación" nos da información acerca de un objetivo de proyecto
  - Finalizar el software en los plazos y costos estimados es siempre un objetivo del proyecto
- Otros objetivos del proyecto, planteados por el Cliente, pueden presentar restricciones importantes al desarrollar software
  - "Queremos que este proyecto sea completamente Open Source, para demostrar un caso de éxito de este enfoque"
  - . . . o "queremos que sea completamente Microsoft"

Objetivos del Sistema

- En el ejemplo, hay afirmaciones del cliente que son, derechamente, características esperadas del software
  - P.ej. "Tener un buen tiempo de respuesta"
- Según nuestra definición, "parece" ser un Requerimiento
- ¿Por qué son Objetivos y no Requerimientos?
  - Porque no podemos verificar inequívocamente que "el sistema tiene un buen tiempo de respuesta"
  - Pero sí podemos Verificar que un listado se despliegue en menos de 2 segundos
- En resumen, características del sistema que no pueden ser verificables siguen siendo objetivos
- Nosotros queremos Requerimientos ☺

Objetivos v/s Requerimientos

- Los objetivos son afirmaciones de alto nivel que nos guían hacia la identificación de requerimientos
  - Los requerimientos deben poder ser verificables (decir inequívocamente que están presentes en el sistema)
- No debemos dejar pasar un objetivo por requerimiento
- Como Analistas, debemos estar atentos en todo momento al Objetivo de Negocio: nuestra experiencia con sistemas puede permitirnos aconsejar al cliente a soluciones más simples que las que a él se le ocurren, pero que siguen cumpliendo el mismo objetivo de negocio

Desafíos en el proceso de elicitación

“Ok, PERO . . .”

- Reacción inicial de un usuario a un sistema:
  - “¡wow! ¡me encanta!”

“Ok, PERO . . .”

- Reacción inicial de un usuario a un sistema:
  - “¡wow! ¡me encanta!”
  - “esta ok, ¿pero que tal si . . . ?”, “¿no seria mejor si . . . ?”

“Ok, PERO . . .”

- Reacción inicial de un usuario a un sistema:
  - “¡wow! ¡me encanta!”
  - “esta ok, ¿pero que tal si . . . ?”, “¿no seria mejor si . . . ?”
- La segunda reacción es la más común ☹
  - es parte de nuestra naturaleza humana

“Ok, PERO . . .”

- Reacción inicial de un usuario a un sistema:
  - “¡wow! ¡me encanta!”
  - “esta ok, ¿pero que tal si . . . ?”, “¿no seria mejor si . . . ?”
- La segunda reacción es la más común ☹
  - es parte de nuestra naturaleza humana
- Solución: tratar de anticipar los “Ok, pero . . . ” durante la elicitación
  - . . . y tomarlos en cuenta en planificación de actividades (más tiempo, recursos)

El síndrome de "Las ruinas por descubrir"

- Es difícil saber cuando finalizar la etapa de elicitación
  - ¿termina cuando se han descubierto todos los requisitos?
  - ¿o cuando se tiene una cantidad no despreciable?
  - . . . es como preguntarle a un arqueólogo cuantas ruinas quedan por descubrir

El síndrome de "Las ruinas por descubrir"

- Es difícil saber cuando finalizar la etapa de elicitación
  - ¿termina cuando se han descubierto todos los requisitos?
  - ¿o cuando se tiene una cantidad no despreciable?
  - . . . es como preguntarle a un arqueólogo cuantas ruinas quedan por descubrir
- Solución: tratar de acotar el esfuerzo de elicitación
  - primero enfocarse en los problemas que el sistema pretende solucionar
  - ¿y lo deseable pero prescindible? solo si queda tiempo/presupuesto

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado
- Los usuarios creen saber lo que quieren, hasta que los desarrolladores hacen una entrega y no es lo que querían

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado
- Los usuarios creen saber lo que quieren, hasta que los desarrolladores hacen una entrega y no es lo que querían
  - Solución: proveer retroalimentación temprana, usar técnicas activas de elicitación como prototipos y escenarios

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado
- Los usuarios creen saber lo que quieren, hasta que los desarrolladores hacen una entrega y no es lo que querían
  - Solución: proveer retroalimentación temprana, usar técnicas activas de elicitación como prototipos y escenarios
- El analista cree entender los problemas del usuario mejor que el usuario

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado
- Los usuarios creen saber lo que quieren, hasta que los desarrolladores hacen una entrega y no es lo que querían
  - Solución: proveer retroalimentación temprana, usar técnicas activas de elicitación como prototipos y escenarios
- El analista cree entender los problemas del usuario mejor que el usuario
  - Solución: pon al analista en el lugar del usuario (juego de roles)

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado
- Los usuarios creen saber lo que quieren, hasta que los desarrolladores hacen una entrega y no es lo que querían
  - Solución: proveer retroalimentación temprana, usar técnicas activas de elicitación como prototipos y escenarios
- El analista cree entender los problemas del usuario mejor que el usuario
  - Solución: pon al analista en el lugar del usuario (juego de roles)
- Todos tienen alguna agenda oculta para este proyecto

El síndrome "Usuario vs. Desarrollador"

- Los usuarios no saben lo que quieren, o saben lo que quieren, pero no saben explicarse
  - Solución: los usuarios son expertos en el tema, probar con distintas técnicas de elicitación si las que están usando no dan resultado
- Los usuarios creen saber lo que quieren, hasta que los desarrolladores hacen una entrega y no es lo que querían
  - Solución: proveer retroalimentación temprana, usar técnicas activas de elicitación como prototipos y escenarios
- El analista cree entender los problemas del usuario mejor que el usuario
  - Solución: pon al analista en el lugar del usuario (juego de roles)
- Todos tienen alguna agenda oculta para este proyecto
  - Solución: todos tenemos alguna agenda, asumir esto y continuar

El síndrome "Vivir con los pecados de los antepasados"

- Los clientes y usuarios tienen memoria
  - éste no es ni el primer ni el ultimo proyecto en que han participado
  - tampoco es el único que ha prometido que todo va a ser mejor
  - otros proyectos han fallado, o entregado productos de baja calidad
  - ¿por qué el de ustedes va a ser diferente?

El síndrome "Vivir con los pecados de los antepasados"

- Los clientes y usuarios tienen memoria
  - éste no es ni el primer ni el ultimo proyecto en que han participado
  - tampoco es el único que ha prometido que todo va a ser mejor
  - otros proyectos han fallado, o entregado productos de baja calidad
  - ¿por qué el de ustedes va a ser diferente?
- Solución: establecer relación de confianza (toma tiempo)
  - retroalimentación temprana, expectativas claras
  - presupuesto y planificación realista
  - entregar features claves en forma temprana

Resumen

- Esta es una de las etapas más críticas de cualquier proyecto
- Debemos tener una clara idea de lo que pide el cliente y los usuarios
- Ing. de Requisitos es un proceso iterativo:
  - Mientras especificamos algunos requisitos, se nos ocurren otros
  - Podemos encontrar contradicciones entre los requisitos
  - Los requisitos pueden cambiar en forma brusca
  - Tener clara trazabilidad para cada requisito
- Siempre validar con cliente y usuarios, el sistema es para ellos

Próxima clase: técnicas para elicitar requisitos y especificarlos