



UNIVERSIDAD TÉCNICA
FEDERICO SANTA MARÍA



Departamento de Informática
Universidad Técnica Federico Santa María

Ingeniería de Requisitos

Ingeniería de Software

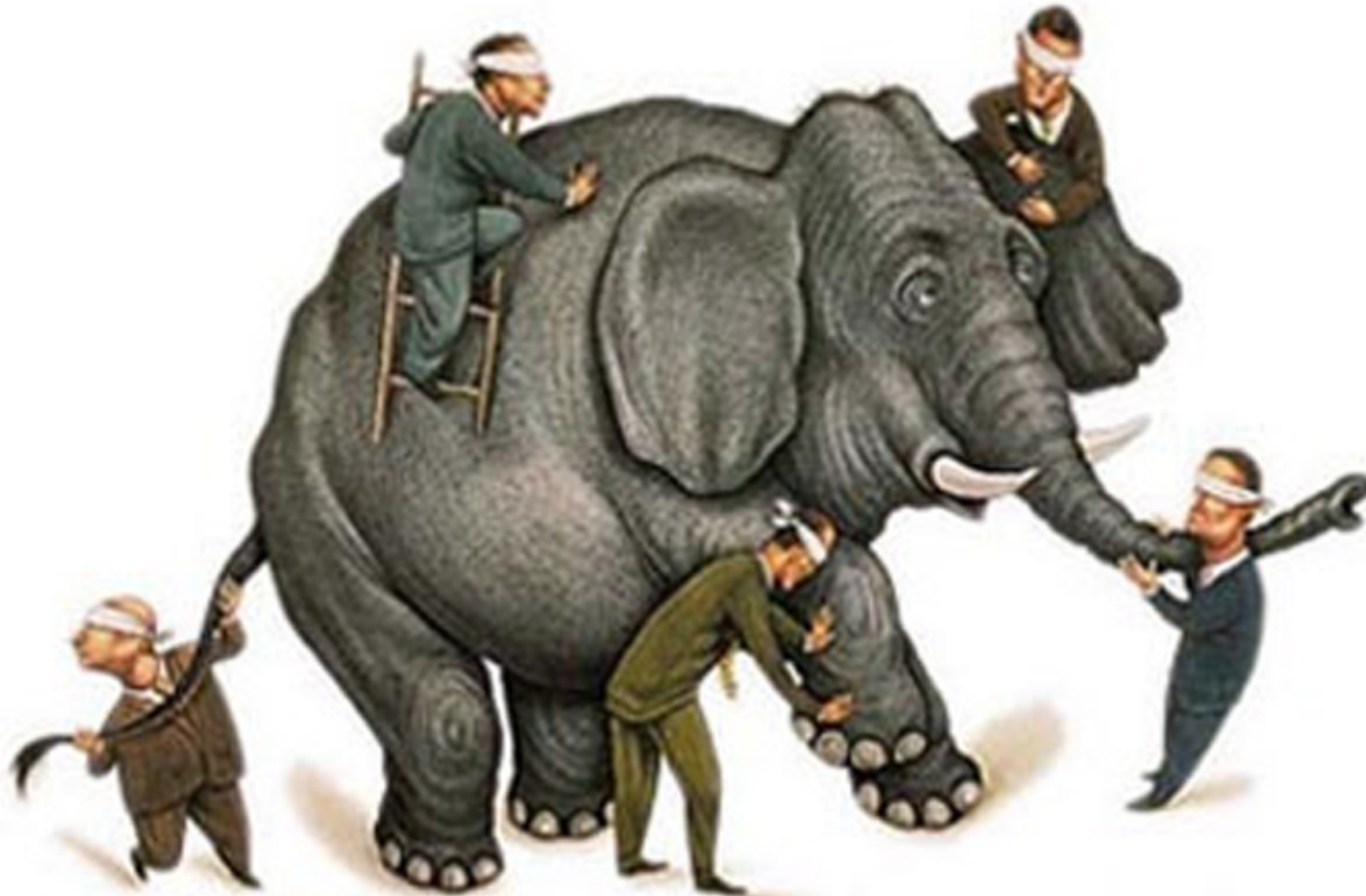
Hernán Astudillo & Gastón Márquez

Departamento de Informática

Universidad Técnica Federico Santa María

<hernan@inf.utfsm.cl, gaston.marquez@sansano.usm.cl>

Contexto



Los ciegos y el elefante

- Un rey le pidió a seis ciegos que tocaran un elefante y que lo describieran
 - el que tocó la *pata* del elefante dijo que era como un *pilar*
 - el que tocó su *cola* dijo que era como una *cuerda*
 - el que tocó la *oreja* dijo que era como un *abanico*
 - el que tocó su *panza* dijo que era como una *pared*
 - el que tocó el *colmillo* dijo que era como una *lanza*.
- Los ciegos se pusieron a discutir sobre lo que era un elefante
 - cada uno pensaba que estaba en lo cierto y que los demás estaban equivocados
- El rey les explicó:
 - “Todos están en lo cierto. Cada uno tocó una parte diferente del elefante. El elefante tiene *todas* las características que mencionaron, aunque parezcan incompatibles.”

A&DSW [1]

- Interacción con el cliente
- El cliente dijo sus necesidades (Sabía que tenía un problema, pero no sabía lo que quería, ¿no?)
- Luego, listado de *posibles* requisitos. ¿Eran todos factibles?
- Los Requisitos se obtuvieron por la visita del cliente (la mayoría)
- ¿Bastaba con solamente “anotar” lo que el cliente decía?
- En general, hubo muchas cosas que no se pudo hacer por tiempo, conocimiento, otros.
- El cliente creyó haber dicho todo, pero bastó un par de preguntas para saber que recién había dicho solo el 20% de sus necesidades

A&DSW [2]

- Entonces, ¿Qué podemos decir de nuestra experiencia?
 - Tuvimos problemas para **entender** los Requisitos que obtuvimos desde el cliente
 - No **organizamos** los Requisitos
 - Gastamos muy poco **tiempo** en verificar lo que íbamos a realizar realmente
 - Nunca consideramos el **control** de los cambios de Requisitos
 - Quizás, fallamos en **establecer** las bases del sistema antes de pensar

A&DSW [3]

- Experiencia de grupos de Santiago y Valparaíso
 - *“Nosotros queríamos contruir software lo más rápido posible y es por eso que empezamos en el momento a desarrollarlo”*
 - (sin haber entendido bien las necesidades del cliente)
 - *“Nosotros al momento de ir desarrollando el proyecto íbamos aprendiendo sobre algunos requisitos que no habíamos entendido bien”*
 - *“Recién después de presentar el proyecto a la cliente, pudimos entender lo que ella realmente quería”*
 - *“Cada vez que le enviábamos un correo a la cliente, algunos requisitos iban cambiando rápidamente, por lo que llegamos a pensar que documentarlos era una pérdida de tiempo”*

A&DSW [4]

- Los argumentos descritos por los equipos contienen mucha verdad...
 - ...para pequeños proyectos que tardan un par de meses en ser desarrollados.
- ¿Qué ocurre con proyectos de gran tamaño y complejidad?
 - Con estos argumentos, el proyecto fallaría.

Enfoque: IR

- IR : Ingeniería de Requisitos
 - Abarca durante la comunicación de las actividades del proyecto hasta el modelamiento
 - Construye un puente entre los Requisitos del sistema, diseño y construcción del software
- ¿Qué permite examinar IR?
 - El contexto del software que será realizado
 - Las necesidades específicas de diseño y construcción
 - Las prioridades que guían el orden en que cada trabajo debe ser realizado
 - La información, función y comportamiento que tendrá impacto en el resultado final

¿Revolucionario u obvio?

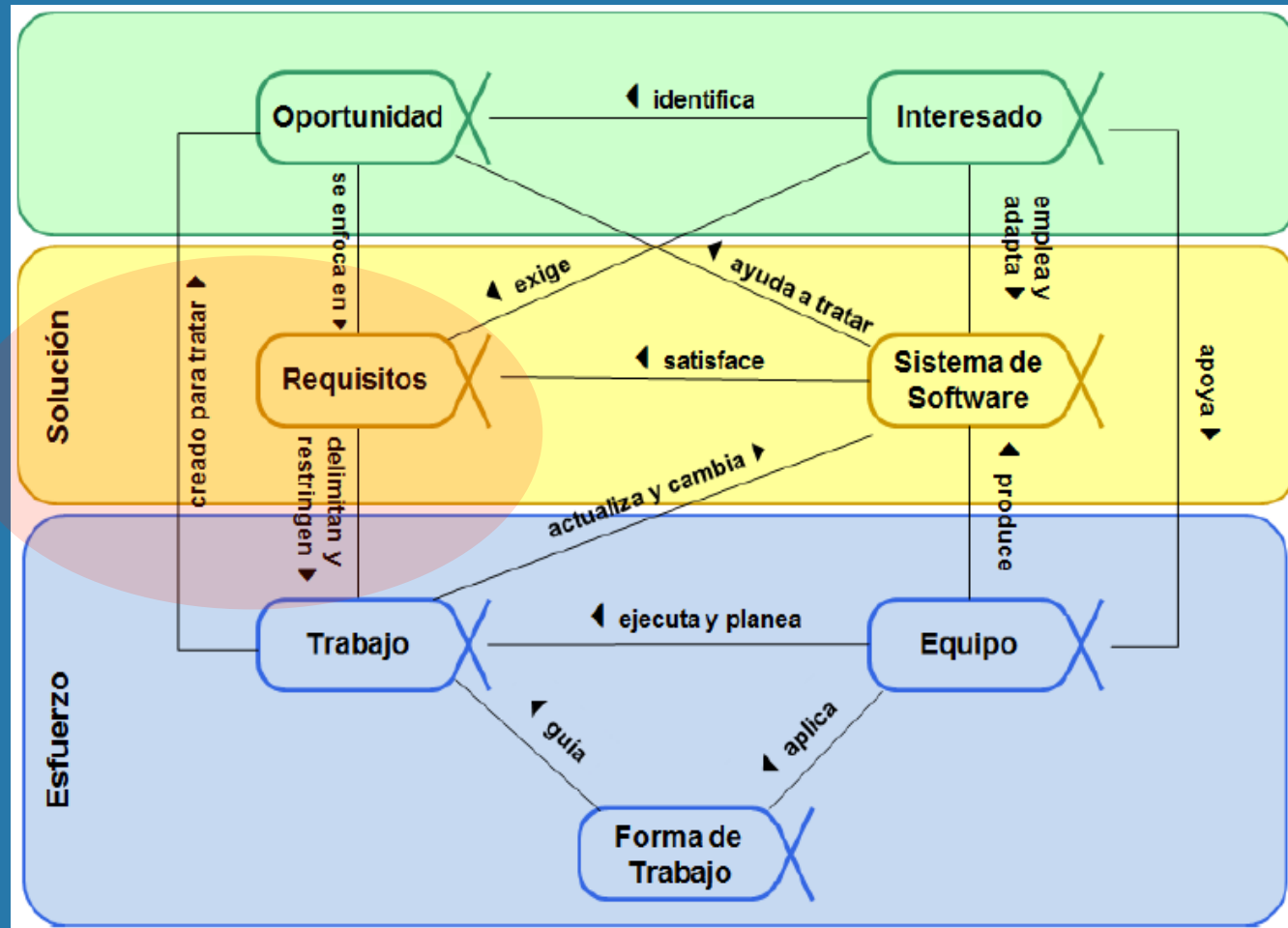
- 30-50% de funcionalidad es innecesaria (overhead)
 - [C. Ebert, R. Dumke - 2007]
- Estudio de 8000 proyectos en EEUU (2000)
 - En plazo y costo: 9%-16%
 - Retrasados/excedidos: 53%-60%
 - Cancelados (abortados): 31%
- ¿Porqué fueron cancelados?
 - Requisitos incompletos: 13%
 - Falta de participación del usuario: 12%
 - Falta de recursos: 11%
 - Expectativas descontroladas: 10%
 - Falta de apoyo gerencial: 9%
 - Requisitos inestables: 9%
 - Falta de planificación: 8%
 - Obsolescencia pre-parto: 7%

Más argumentos

- Cuantificación de Impacto
 - Costo de reparar errores en los requisitos...
 - Detectado durante análisis: \$X
 - Detectado durante diseño: \$5X
 - Detectado durante construcción: \$10X
 - Detectado durante prueba: \$20X
 - Detectado después de entregado: \$100X-\$200X

Requisitos

Contexto en SEMAT



Contexto en SEMAT

- Los *Interesados (stakeholders)* exigen requisitos
- La *Oportunidad* se enfoca en requisitos
- El *Sistema de Software* satisface los requisitos
- Los requisitos delimitan y restringen el *Trabajo* (proyecto)

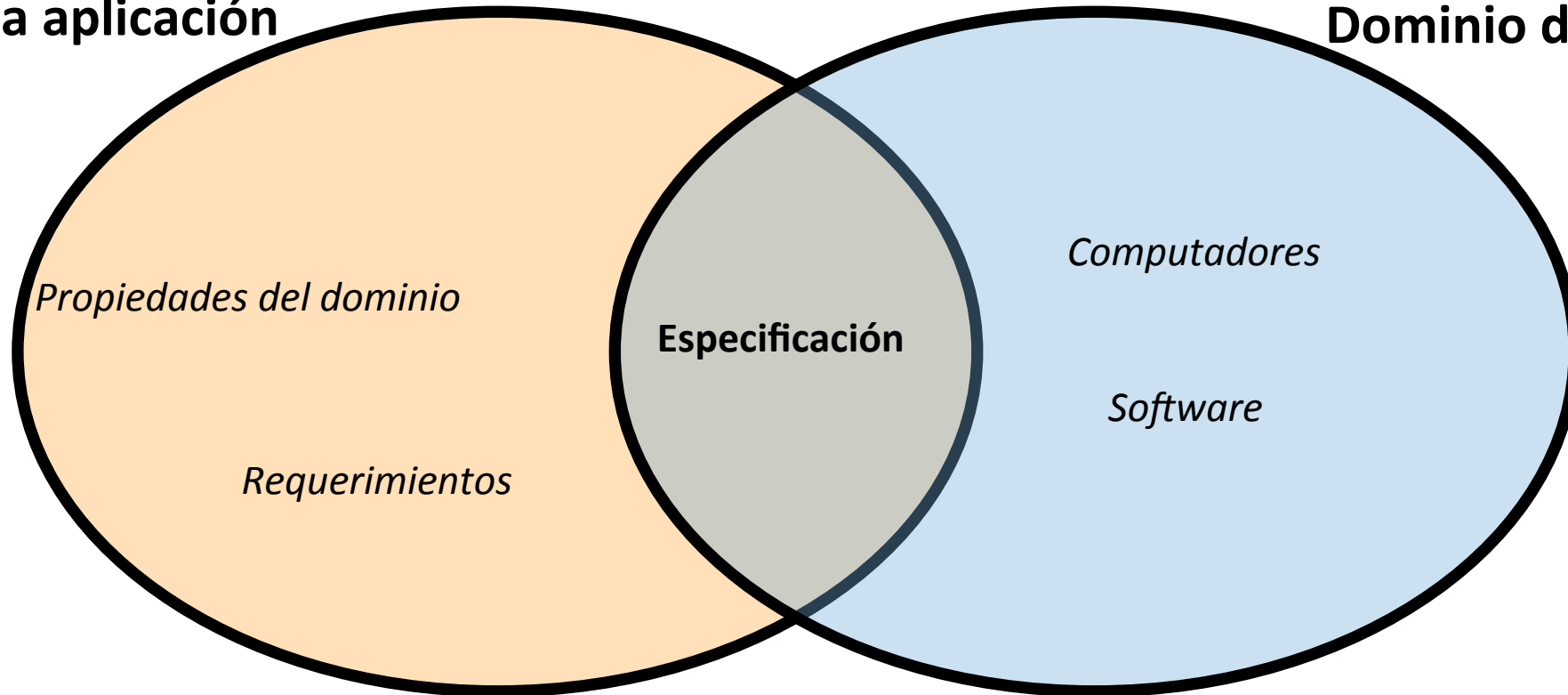
Requisitos [1]

- Requisitos: descripciones de lo que el sistema debe/debería hacer
 - Capacidades y condiciones a las que el sistema debe adherir
- Clasificación típica:
 1. *Requisitos funcionales*: servicios/funciones a proveer (ej: “sistema debe calcular promedio de ingreso mensual”)
 2. *Requisitos extra-funcionales* (NFRs, non-functional requirements): restricciones sobre servicios/funciones del sistema (ej: “uptime de 99%”)
- FURPS+: otro modelo de clasificación
 - Functional, Usability, Reliability, Performance, Supportability, (+: Implementation, Interface, Operations, Packaging, Legal)

Requisitos [2]

Dominio de la aplicación

Dominio de la máquina



Requisitos [3]

- Propiedades del dominio
 - *Cosas* en el dominio de la aplicación que son verdaderas o no cuando se construye el softwarepropuesto
- Requisitos
 - Cosas en el dominio de la aplicación que deseamos que se hagan realidad mediante la construcción del software
- Especificación
 - Es la descripción del comportamiento que el software debe tener para cumplir con los requisitos

Requisitos y riesgo

- ¿Para qué estudiamos requisitos?
 - Para reducir el riesgo de construcción
 - (R1) Efectividad: riesgo de hacer algo inútil
 - (R2) Eficiencia: riesgo de sub-/sobre-estimar recursos
- Clasifiquemos las causas de cancelación
 - Requisitos incompletos: 13% (R1)
 - Falta de participación del usuario: 12% (R1)
 - Falta de recursos: 11% (R2)
 - Expectativas descontroladas: 10%
 - Falta de apoyo gerencial: 9%
 - Requisitos inestables: 9%
 - Falta de planificación: 8% (R2)
 - Obsolescencia pre-parto: 7% (R1)

Requisitos funcionales [1]

- Funcionalidad, características, servicios que el sistema:
 - Debe proveer (si son obligatorios)
 - Debería proveer (si son deseables)
- Analogía:
 - Requisitos funcionales: los “verbos”

Requisitos funcionales [2]

- Ejemplo: Proyecto Programa Institucional de Inglés Comunicacional (<http://eq42.fdo-may.ubiobio.cl/sgic/>)
- Algunos Requisitos funcionales
 - *El sistema debe permitir la inscripción de una asignatura de inglés comunicacional, mostrando él o los posibles cursos a inscribir.*
 - *El sistema debe permitir la autenticación del usuario a través del RUT y la contraseña institucional.*
 - *El sistema debe reconocer la sede respectiva del estudiante, Chillán o Concepción. Para así mostrar las asignaturas y secciones que corresponden.*

Requisitos extra-funcionales (NFRs) [1]

- Condiciones y restricciones sobre el sistema
- Potenciales orígenes:
 - Producto (ej: *uptime de 99%; respuestas en menos de 5[s]*)
 - Organización (ej: *desarrollo sólo con .NET*)
 - Externos (ej: *cumplimiento de normas bancarias, cumplimiento DO-178C en software aeronáutico*)
- No minimizar impacto: NFR no satisfechos pueden dejar un sistema sin uso
 - *Ej: sistema en avión no cumple con condiciones de estabilidad no es certificado y queda fuera del mercado (pero hacía lo que debía hacer)*
- Analogía:
 - Requisitos extra-funcionales: los “adverbios”

Requisitos extra-funcionales (NFRs) [2]

- Ejemplo: Proyecto Programa Institucional de Inglés Comunicacional (<http://eq42.fdo-may.ubiobio.cl/sgic/>)
- Algunos Requisitos extra-funcionales
 - *La mayoría de los usuarios de la aplicación no son personas experimentadas. Por lo anterior, el sistema deberá ser fácil de entender y utilizar. (Usabilidad)*
 - *El representante de la empresa debe tener el 100% de confianza en el sistema, por lo que en cada entrega del sistema se probará todo. (Fiabilidad)*
 - *El representante de la Institución dio a entender que es de suma importancia mantener el control de acceso al sistema. (Seguridad)*
 - *El sistema debe funcionar en un servidor HTTP Apache 2.2.3, usando MySQL versión 5.0, Sublime Text 2.0 y Netbeans 7.4.*

Ejercicio

- Formar los grupos ya establecidos para el proyecto de ISW
- Identificar requisitos funcionales y extra-funcionales del proyecto semestral.
- (pausa)
- ¿Son los mismos en todos los grupos?

Proceso de Ingeniería de Requisitos

Ingeniería de Requisitos [1]

- Definición de IR [Zave, 1997]

*“La Ingeniería de Requisitos es la rama de la ingeniería de software ocupada de las **metas del mundo real**, funciones de, y restricciones de sistemas de software. También se ocupa de la relación de estos factores con **especificaciones precisas** de comportamiento de software, y de su **evolución en el tiempo y a través de familias de software**.”*

Ingeniería de Requisitos [2]

- Puntos clave
 - “Metas del mundo real”
 - Representa el ¿porqué? y ¿qué? de un sistema
 - “Especificaciones precisas”
 - Provee las bases para...
 - **Analizar** los Requisitos
 - **Validar** lo que los stakeholders quieren
 - **Definir** lo que los diseñadores deben proponer que se construya
 - **Verificar** que se está haciendo lo correcto
 - “Evolución en el tiempo y a través de familias de software”
 - Enfatiza la realidad del cambio tecnológico constante

Ingeniería de Requisitos [3]

- Pero, ¿por qué Ingeniería?
 - Una definición típica de Ingeniería es *creación de soluciones costo-efectivas a problemas prácticos aplicando conocimiento científico* [Shaw, 1990]
- Interesante, ¿no?
 - El término Ingeniería nos recuerda que IR es una parte importante en la ingeniería de procesos: toma problema del mundo real (*real world goals*), y se idea una solución óptima (*precise specifications*) a ser desarrollada en una plataforma (*evolution over time and across software families*)

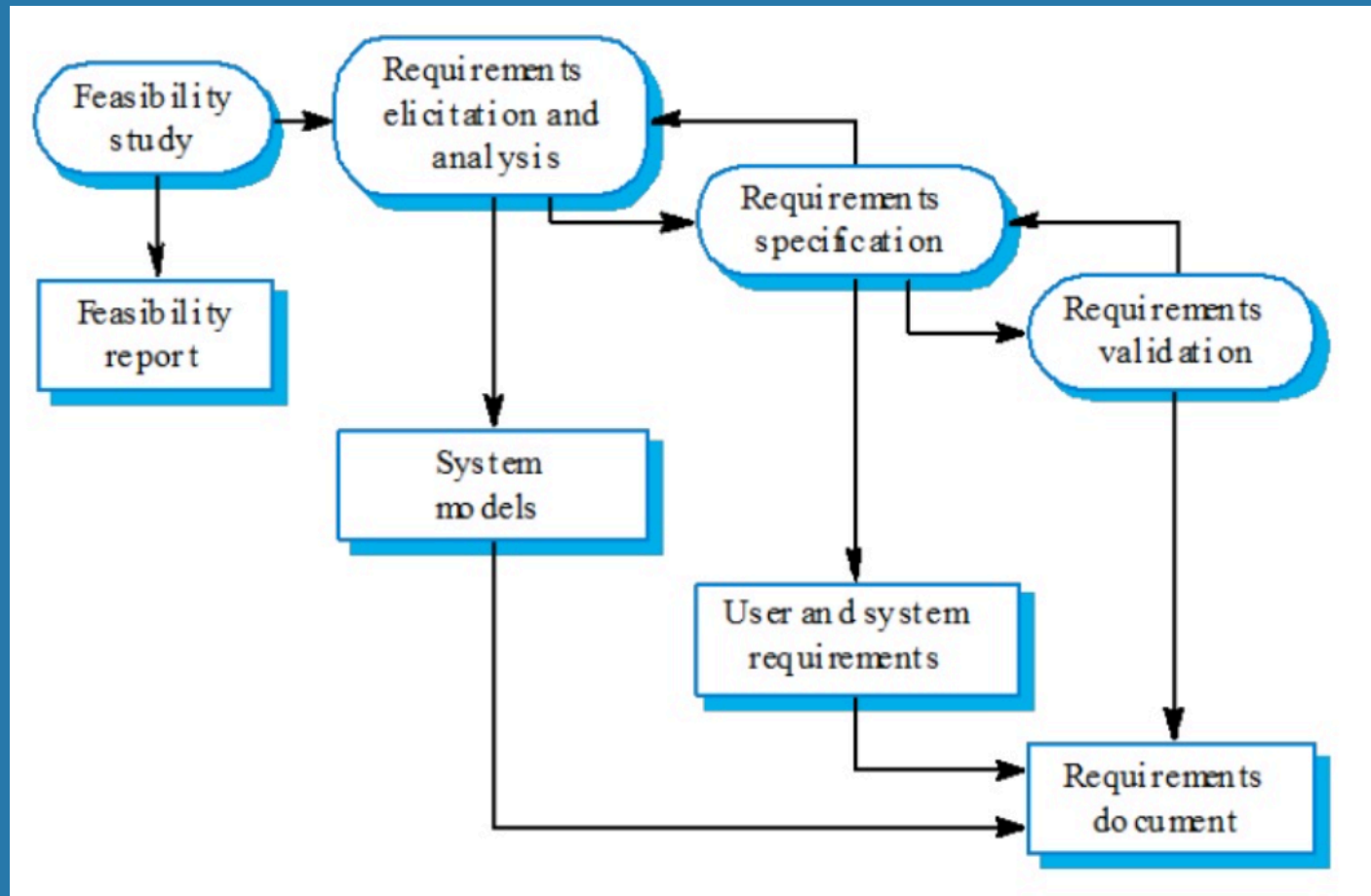
Ingeniería de Requisitos [6]

- Lo anterior nos hace pensar
 - Lo que hicimos en A&DSW ¿Fue suficiente?
 - ¿Basta solamente escuchar al cliente y tomar nota?
- Por lo que se ve, se puede hacer muchas cosas más que sólo anotar frases del cliente.
- Entonces, entendamos qué son los Requisitos Funcionales y Extra- Funcionales

Proceso de Ingeniería de Requisitos [1]

- El proceso usado para IR varía dependiendo del dominio de la aplicación, las personas involucradas y los requisitos obtenidos
- Actividades genéricas comunes a todos los procesos:
 1. Elicitación (levantamiento) de Requisitos
 2. Análisis de Requisitos
 3. Validación de Requisitos
 4. Administración de Requisitos

Proceso de Ingeniería de Requisitos [2]



Proceso de Ingeniería de Requisitos

1. Estudio de Factibilidad

- Decidir si el proyecto se realizará o no
 - ¿El sistema contribuye a los objetivos organizacionales?
 - ¿Se puede realizar con la tecnología y presupuesto actual?
 - ¿Se puede integrar con otros sistemas?

2. Elicitación (levantamiento) y Análisis

- Identificar y describir requisitos
 - ¿Saben los stakeholders lo que quieren?
 - ¿Los stakeholders expresaron correctamente sus necesidades?
 - ¿Existen conflictos de interés entre los stakeholders?

3. Validación

- Constatar si los requisitos representan bien lo deseado
 - ¿Permiten definir el sistema?
 - ¿Hay costos y prioridades asociados a los requisitos?
 - ¿Hay una forma de arreglar problemas?

Proceso de Ingeniería de Requisitos

- La IR no es necesariamente un proceso secuencial
 - Es útil actualizar la declaración del problema en cada etapa del desarrollo
 - Las actividades de la Ingeniería de Requisitos continúan a través del proceso de desarrollo

Perspectivas para Describir Requisitos

- Funcional (clásico)
 - Especificar en lenguaje natural lo que el sistema debe hacer
 - “El sistema debe permitir inscribir, modificar y desinscribir cursos”
- Operacional
 - Escenarios concretos de acción
 - Casos de uso
 - Otros tipos de escenarios (casos límite, casos de abuso, etc.)
- Objetivos
 - Especificar no qué hará el sistema, sino para qué
 - Permiten negociación entre stakeholders & priorizar escenarios

Lo perfecto es enemigo de lo bueno

- La descripción del problema siempre será imperfecta
 - Contendrá inconsistencias
 - Omitirá información
- La solución perfecta puede no ser efectiva en costo
 - El análisis de Requisitos es costoso
 - Para diferentes proyectos, el costo/beneficio es distinto
- La declaración del problema nunca debe considerarse como “final”
 - Los cambios son inevitables, y por lo tanto deben ser planificados
 - Debe existir una forma para incorporar cambios periódicamente

Elicitación de Requisitos

Elicitación de Requisitos

- La elicitación de Requisitos es normalmente la actividad inicial en IR
- Elicitación → Captura [Goguen et al., 1994]
- La información obtenida de la elicitación es interpretada, analizada, modelada y validada hasta que los ingenieros de Requisitos estén seguros de que está (razonablemente) completa

Elicitación de Requisitos

- Determinar *qué problemas deben ser resueltos* e identificar los *límites del sistema* [Nuseibeh et al., 2000]
 - Los límites definen, a alto nivel, cómo el sistema mejorará el entorno actual
- Identificar a los *stakeholders* [Sharp et al., 1999]
 - Clientes (Customers) → los que pagan el sistema
 - Desarrolladores → los que diseñan, construyen y mantienen el sistema
 - Usuarios → los que interactúan con el sistema para realizar su trabajo
- Determinar los *objetivos* del sistema [Dardenne et al., 1993]
 - Permite enfocarse en el dominio del problema y las necesidades de los stakeholders, encontrando posibles soluciones

Elicitación de Requisitos

- Los usuarios pueden encontrar difícil articular toda esta información
 - Si hubiésemos entregado los requisitos directamente a nuestro cliente en A&DSW...
 - ...probablemente hubiera leído sólo la primera página ☹

Requisitos a elicitar [3]

- Para no tener problemas con los usuarios, los ingenieros de requisitos pueden describir las **tareas** de los usuarios.
- Dichas tareas puede ser representadas por los *Casos de Uso* [Schneider et al., 1998] en un *escenario* determinado [Jarke et al., 1998]

Técnicas de elicitación [1]

- Técnica tradicional
 - Cuestionarios
 - Encuestas
 - Entrevistas
 - Análisis de la documentación existente
 - Análisis de modelos de procesos y estándares
- Técnicas grupales
 - *Brainstorming*
 - *Focus group*
 - RAD/JAD workshop [Maiden et al. 1996] (<https://bobsleanlearning.wordpress.com/reference/what-is-rad-jad/>)

Técnicas de elicitación [2]

- Prototipos
 - Ayudan al rápido feedback del cliente
 - Provoca discusión y mejoras
- Model-Driven
 - Métodos basados en Metas
 - KAOS [van Lamsweerde et al., 1998] (<http://www.objectiver.com/fileadmin/download/documents/KaosTutorial.pdf>)
 - i* (“i-star”) [Chung et al., 2000] (<http://www.cs.toronto.edu/km/istar/>)
 - Métodos basados en Escenarios
 - CREWS [Maiden et al., 1996] (<http://sunsite.informatik.rwth-aachen.de/CREWS/>)

Técnicas de elicitación [3]

- Cognitivas
 - Análisis de protocolo → los expertos piensan mientras desarrollan una tarea siendo observado por especialistas cognitivos
 - “Laddering” → uso de sondas para elicitar estructuras y contenidos del conocimientos del stakeholder
 - “Card sorting” → se le pide a los stakeholders ordenar cartas según dominio de entidades
 - “Repertory grids” → matriz de entidades a partir de preguntas a los stakeholders
- Contextuales
 - Alternativas a las cognitivas [Goguen, 1993]. Agregan técnicas etnográficas y análisis de conversaciones para obtener patrones de conversación e interacción

Modelado, Análisis y Comunicación

Modelado y Análisis de Requisitos [1]

- Ejemplo
 - *En A6DSW se tuvo que hacer modelos (Diagrama de Casos de Uso, Diagramas de Clases, otros); ¿qué se buscaba?*
- Modelamiento
 - Descripciones abstractas para ser interpretadas
 - Los modelos pueden ser utilizados para presentar muchas técnicas
 - ¿Para qué sirven los modelos?
- En las siguientes diapos, se verá algunas utilidades de los modelos bajo distintos contextos [Nuseibeh et al., 2000]

Modelado y Análisis de Requisitos [2]

- Modelado Empresarial
 - La mayoría de las tareas de IR se relacionan con la organización que necesita el software.
 - El modelado empresarial es útil para capturar el propósito del software en una organización [Loucopoulos et al., 1995].
 - Los modelos de objetivos son muy útiles en IR.
- Modelado de Datos
 - Los proyectos de software generan grandes volúmenes de información.
 - El modelamiento de datos ayuda a encontrar oportunidades de decisiones en las organizaciones
 - Entity-Relationship-Attribute (ERA) [<http://www.tac.mta.ca/tac/volumes/10/3/10-03.pdf>]

Modelado y Análisis de Requisitos [3]

- Modelado de Comportamiento
 - Modelar Requisitos involucra involucra modelar el comportamiento de los stakeholders.
 - ¿Qué se obtiene con esto?
 - Saber cómo se maneja el trabajo
 - Determinar funcionalidades clave
 - Finalmente, determinar un modelo operativo del sistema
- Modelado del dominio
 - Una parte importante de IR es sobre el desarrollo de las descripciones del dominio [Jackson et al., 1993]
 - El modelo del dominio provee una descripción abstracta del mundo en donde el software operará.

Modelado y Análisis de Requisitos [4]

- Modelado de requisitos extra-funcionales
 - Estos requisitos son generalmente los más difíciles de expresar como medición.
 - Los modelos que representan los NFRs se hacen para poder medir y testear el software.
 - Hay mucho conocimiento sobre algunos NFRs clave: seguridad, confiabilidad, escalabilidad, usabilidad...
- Ejemplo
 - *¿Cómo se podría haber modelado la confiabilidad del proyecto en ADSW?*
 - *¿Y en su proyecto de ISW?*

Modelado y Análisis de Requisitos [5]

- Análisis de Modelos de Requisitos
 - Un beneficio de modelar requisitos, es que se pueden analizar.
- Algunas técnicas de análisis de requisitos
 - Animación de requisitos [Gravell et al., 1996]
 - Razonamiento automático (case-based reasoning) [Maiden et al., 1992]
 - Knowledge-based critiquing [Fickas et al., 1988]
 - Consistency checking [Holzmann, 1997]
 - Validation & Verification (V&V)

Comunicación de los Requisitos [1]

- IR permite no sólo *descubrir y especificar*, sino también *comunicar* requisitos
 - Los *stakeholders* deben poder entender & validar fácilmente la descripción de sus necesidades
 - Es clave saber *cómo* se documentará los requisitos
- El foco de la documentación radica en la especificación de lenguajes y anotaciones, con variadas propuestas:
 - Lenguajes formales y semi-formales [Wieringa, 1996]
 - Lenguaje lógico [Antoniou, 1998]
 - Lenguaje natural [Ambriola et al., 1997]

Comunicación de los Requisitos [2]

- Volere [<http://www.volere.co.uk/index.htm>] es una de las formas de especificar y documentar Requisitos más usadas en la industria
- Ranking de documentación de requisitos [Makar, LiquidPlanner, 2015] más utilizados (industria, academica, investigación, otros)
 - Context Diagram
 - Functional Decomposition
 - Use Case Diagram
 - Sequence Diagram
 - AS-IS and TO-BE process model
 - User Stories
 - Mind maps

¿Qué hacen los analistas de requisitos?

- El analista de requisitos se formula 7 preguntas: ¿Quién?, ¿Qué?, ¿Dónde?, ¿Cuándo?, ¿Porqué?, ¿Cuál? y ¿Cómo? [Esterbrook, 2004]
 - ¿*Cuál* es el problema que necesita ser resuelto?
 - Límites del problema
 - ¿*Dónde* está el problema?
 - Contexto organizacional & Dominio del problema
 - ¿*Cuándo* necesita ser resuelto?
 - Restricciones de desarrollo
 - ¿*Qué* problema prevenimos si lo resolvemos?
 - Identificar Factibilidad & Riesgo
 - ¿De *quiénes* es el problema?
 - Identificar a los *stakeholders*
 - ¿*Porqué* necesita ser resuelto?
 - Identificar objetivos
 - ¿*Cómo* ayudará un software?
 - Identificar escenarios

Ejercicio

- Trabajar en los grupos del proyecto de ISW y responder las 7 preguntas para el proyecto.
- (pausa)
- Debatir:
 - ¿Tenemos todas respuestas parecidas? ¿O no? ¿Por qué?

Requisitos- Conclusiones

Evaluación de Especificaciones

- Nuestro propósito: determinar si ciertos requisitos propuestos son...
 - correctos (describen algo bien)
 - útiles (sirven para guiar construcción)
 - necesarios y suficientes
- Preguntas
 - ¿Son correctos?
 - ¿Son consistentes?
 - ¿Están completos?
 - ¿Son realistas?
 - ¿Son verificables?
 - ¿Describen una sola cosa (cada uno)?
 - ¿Son rastreables?

Requisitos y SQA

- Idea importante: trazabilidad de Requisitos
 - Reqs. alto nivel → Reqs. bajo nivel (detalles) → Modelos → Código → Ejecución
- Permite tener certeza de que lo que deseaba el usuario es lo que está en ejecución
- Herramientas actuales apoyan la trazabilidad
 - Requisitos se plasman en casos de pruebas que son automatizados y ejecutados contra el código de manera automática (o semiautomática)
- Métricas importantes:
 - Complejidad/Tamaño
 - Volatilidad (cambios/tiempo)
 - Tasa Reqs. Implementados (implementados/total)



UNIVERSIDAD TÉCNICA
FEDERICO SANTA MARÍA



Departamento de Informática
Universidad Técnica Federico Santa María

Ingeniería de Requisitos

Ingeniería de Software

Hernán Astudillo & Gastón Márquez
Departamento de Informática
Universidad Técnica Federico Santa María