



Ingeniería de Software I

Carlos Monsalve
monsalve@espol.edu.ec

Sección 4: Requerimientos

Ciclo de Vida

Etapas clásicas



Adaptado de "Introduction to Software Life Cycles", Element K, 2012

Requerimientos | ¿Qué son?

- Necesidades del cliente
 - Cumplimiento de objetivos
 - Problemas a solucionar
 - Mejoras a introducir
- Cada usuario tiene su propia visión de un mismo requerimiento



Requerimientos | Software vs. sistema

- Requerimientos del sistema
 - Incluye los requerimientos de software
 - Incluye otros requerimientos:
 - Hardware
 - Organizacionales
 - Telecomunicaciones, etc.
- Requerimiento de software: aquellos que son atacados directamente por una aplicación

Requerimientos | Características

- Únicos
- Completos
- No ambiguos
- Consistentes
- Factible
- Trazable
- Verificables



Cuento Sufi de Blanca Martin © 2011 Equilibre Gaia

Requerimiento

Sintaxis recomendada

[Condición] [Sujeto] [Acción] [Objeto] [Limitación]

Ejemplo: Cuando la señal x es recibida **[Condición]**, el sistema **[Sujeto]** debe fijar **[Acción]** el bit de señal x recibida **[Objeto]** dentro de 2 segundos **[Limitación]**.

[Condición] [Acción o Limitación] [Valor]

Ejemplo: En el estado de mar 1 **[Condición]**, el Sistema de Radar apunta a rangos más allá de **[Acción o Limitación]** 100 millas nauticas **[Valor]**.

[Sujeto] [Acción] [Valor]

Ejemplo: El Sistema de Facturación **[Sujeto]**, debe mostrar las facturas pendientes **[Acción]** en orden ascendente **[Valor]** en las que las facturas se pagarán.

Adaptado de: IEEE, Systems and software engineering — Life cycle processes — Requirements engineering, ISO/IEC/ IEEE 29148,2011

Requerimientos

Posibles fuentes

- Objetivos
- Conocimiento del dominio
- Actores clave: considerarlos a todos
- Ecosistema de operación: limitaciones
- Ecosistema de la organización: procesos de negocio

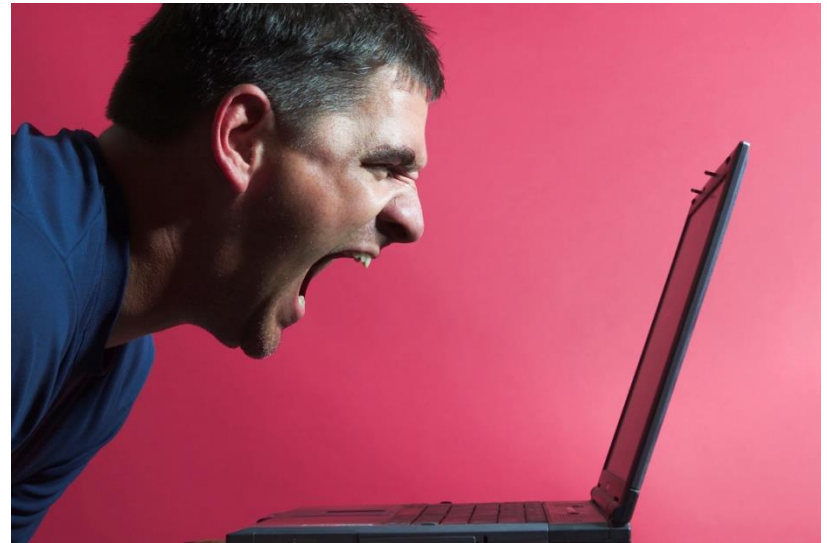
Requerimientos | Tipos

- Funcionales
 - **Capacidades** de comportamiento del software
- No Funcionales
 - Limitaciones
 - Requerimientos de **calidad**

Requerimientos Funcionales

Características

- Describen funciones a incorporarse en el software
- Representan expectativas funcionales de los actores clave
- Se describen como acciones de usuarios



Requerimientos No Funcionales

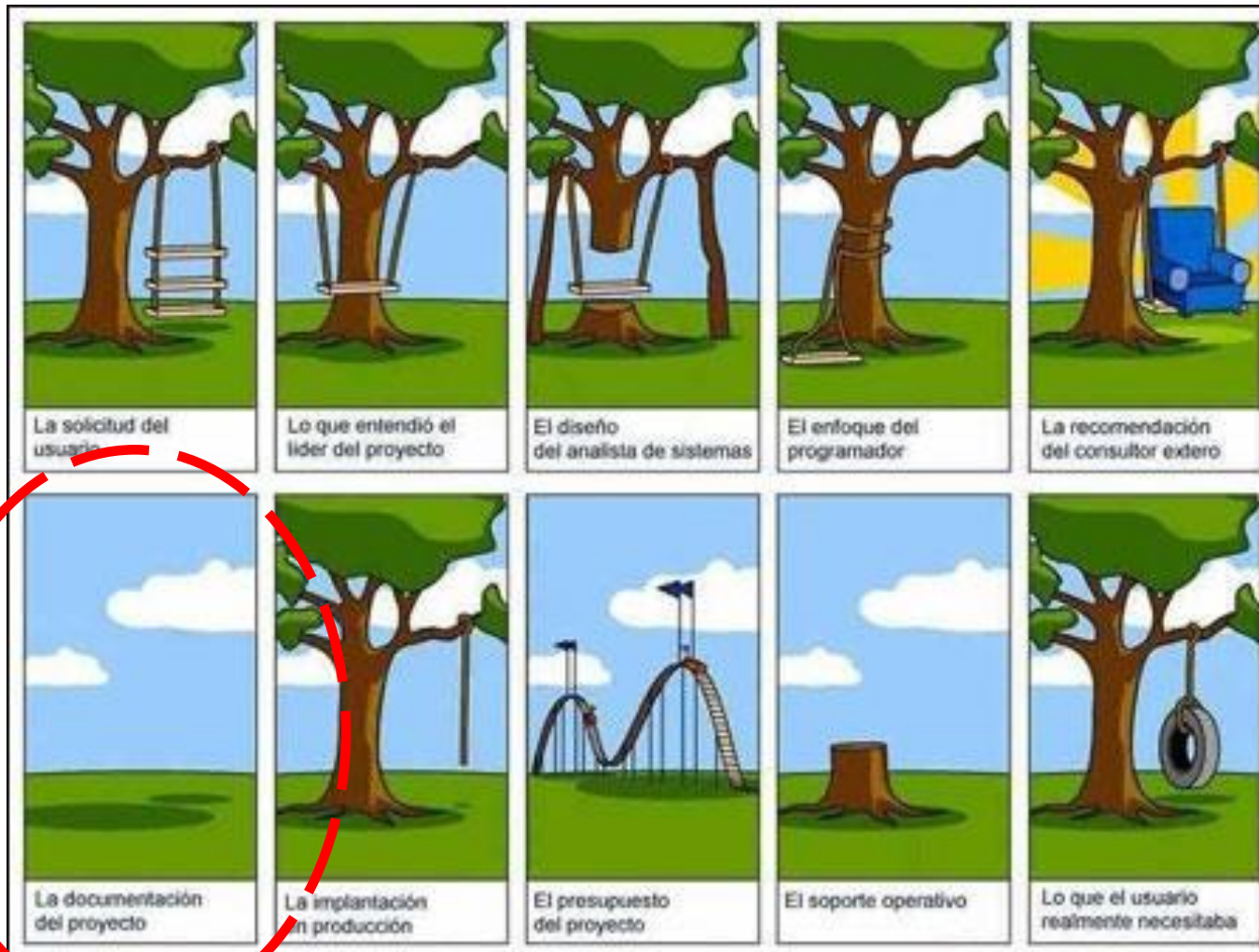
Características

- Atributos de calidad
- Definen características de:
 - Evolución del sistema (ej.: mantenibilidad)
 - Ejecución del sistema (ej.: seguridad)
- Ejemplos:
 - Confiabilidad
 - Escalabilidad
 - Rendimiento
 - Usabilidad

Requerimientos | ¿Porqué?

- Definen alcance del proyecto
- Mejor planificación
- Asignación de recursos
- Estimación
- Correcto diseño
- Satisfacción del cliente

Requerimientos Impacto



Impacto | No funcionales

- Tan importantes como los funcionales
- Ejemplos:
 - No cumplir criterios de seguridad
 - No cumplir expectativas de usabilidad
 - No ofrecer la escalabilidad deseada
 - No haber analizado la carga de transacciones

Requerimientos | Actividades críticas

- Identificar actores claves
- Levantar los requerimientos
- Analizar los requerimientos
 - Categorizarlos
 - Priorizarlos
- Documentar los requerimientos
- Validar y verificar los requerimientos

Actores Claves Tipos

- Dueños del negocio
- Usuarios finales
- Personal técnico
- Reguladores
- Gobierno
- Etc.



Actores Claves

Pirámide Organizacional



Levantar Requerimientos

Realidades

- Quizá la parte mas dura
- Actores
 - Difíciles de determinar
 - Difíciles de comprometer
 - Desconocen los problemas
 - Tienen dificultades en describir los problemas
- Requiere:
 - Participación muy activa
 - Considerar diferentes puntos de vista

Levantar Requerimientos

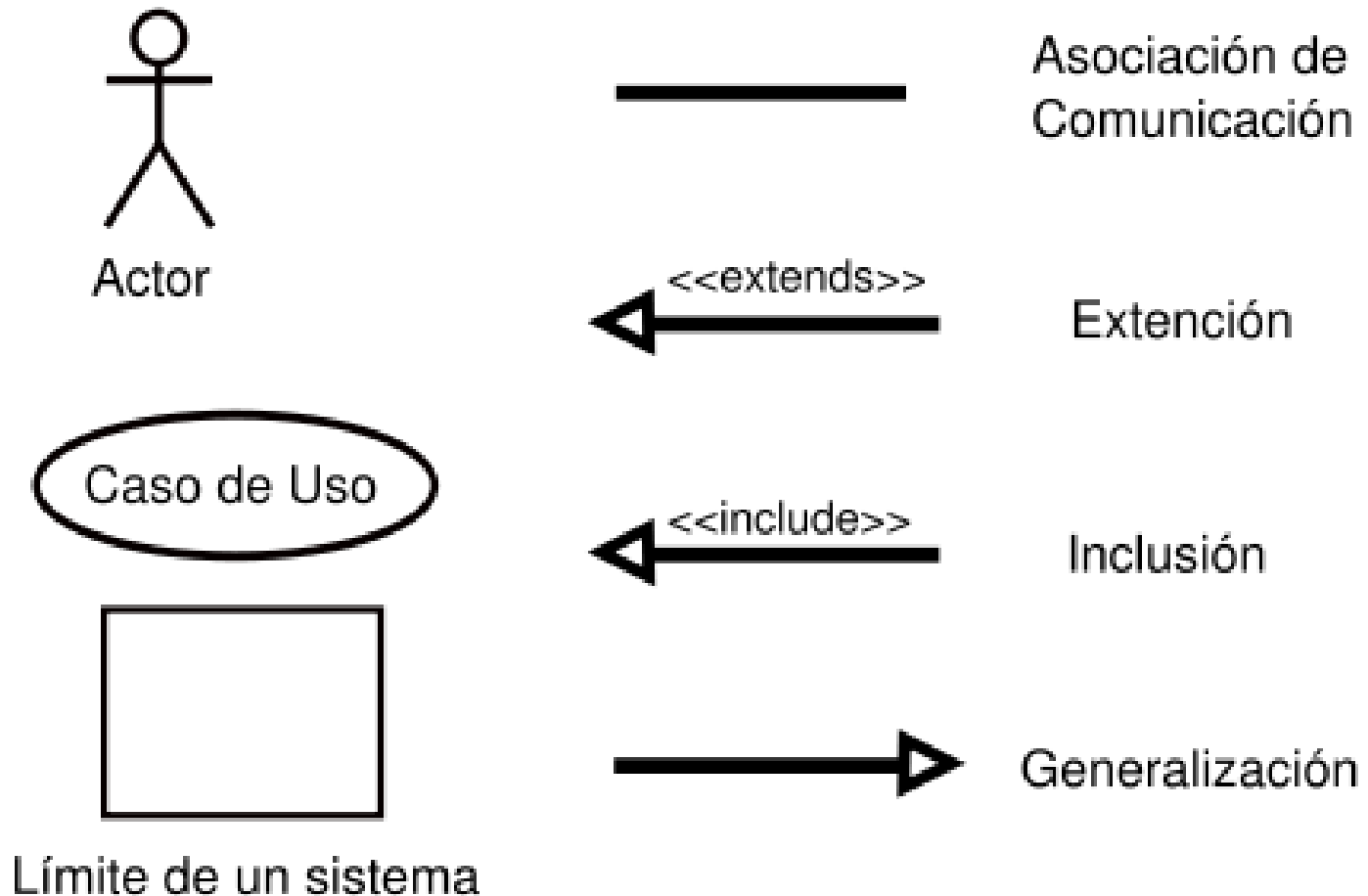
Técnicas

- Reuniones de lluvia de ideas (blue sky)
- Observación
- Entrevistas
- Escenarios: modelamiento conceptual
 - Casos de uso
- Encuestas
- Prototipos (ej.: de interface)

Casos de Uso | ¿Qué son?

- Describe:
 - Requerimientos funcionales
 - Interacciones entre software y actores
- Actores:
 - Usuarios
 - Hardware
 - Otras aplicaciones de software

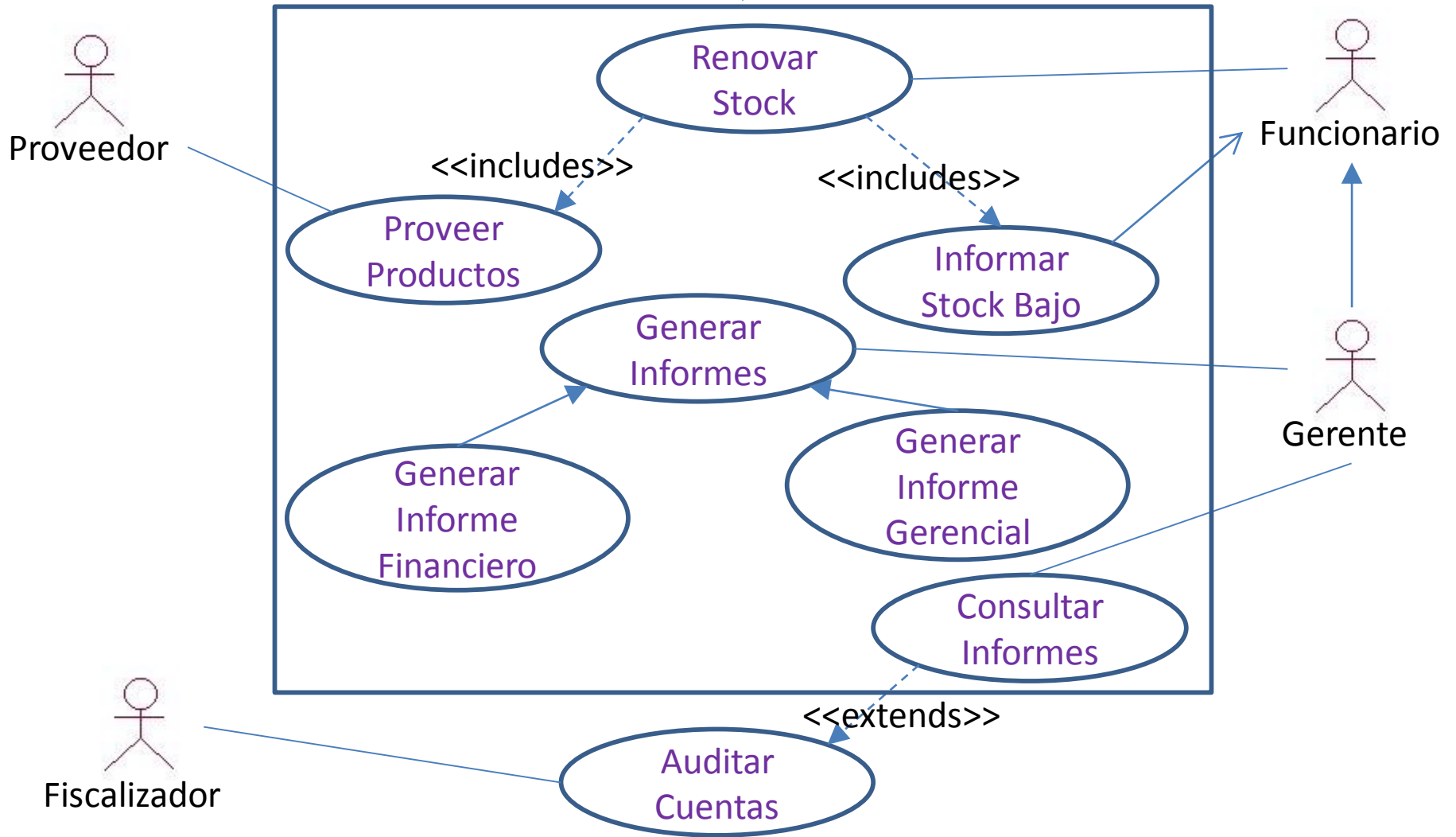
Casos de Uso | Elementos



Casos de Uso | Interacciones

- Inclusión:
 - Cuando un caso de uso incluye a otro
 - Incluido siempre ayuda a incluyente
- Extensión
 - Cuando un caso de uso puede requerir ayuda de otro
- Generalización:
 - Entre casos de uso o entre actores
 - Un elemento hereda propiedades del otro

Casos de Uso | Ejemplo



Casos de Uso | Detalles

- Identificador único y Título
- Actor(es): principal y secundarios
- Objetivos
- Resumen
- Pre-condiciones
- Post-condiciones
- Flujo ideal (caso principal, de éxito, ideal)
- Extensiones: flujo(s) alternativo(s)
- Disparadores (triggers) de cada flujo

Prototipo de Interface

Lo-Fi Mockup

- Para levantar requerimientos de interfaces de usuario
- Muy usado en desarrollo ágil
- Materiales: papel, lápiz, borrador, tijeras
- Fácilmente corregibles
 - ¿descartables?

Lo-Fi Mockup

Ejemplo

FANS DEL IDOLO!

SOLICITAR ADMISION

NOMBRE

APELLIDO

F. NAC.

email

- Consistentes con las historias de usuario
- Luego graficar interacciones
 - Crear guion a base de los mockups
- Herramientas:
 - Mockup builder
 - Framebox

Prototipo de Software

¿Porqué no?

- No para levantamiento de interfaces
- Puede intimidar a los usuarios
- La interacción con papel o medio virtual es más simple y sencillo
- Ayuda a concentrarse en lo funcional

Ágil | Historia de usuario

Perspectiva	Título	Prioridad
Como <i>tipo de usuario</i>		
	Requerimiento	
Quiero <i>actividad que quiere realizar</i>		
Con la finalidad de <i>lograr algún objetivo</i>	Razón	
Autor	Fecha	Estimado

Analizar Requerimientos

¿Para qué?

- Detectar posibles conflictos
- Solución de conflictos
- Determinar frontera del software
- Determinar como software interactúa con su medio ambiente

Requerimientos | Características

- Únicos
- Completos
- No ambiguos
- Consistentes
- Factible
- Trazable
- Verificables



Cuento Sufi de Blanca Martin © 2011 Equilibre Gaia

Analizar Requerimientos

Claves

- Comprender los procesos de negocio
- Conocer otras aplicaciones de la empresa
- Estimar impacto de la solución en el ecosistema
- Analizar el desarrollo futuro

Analizar Requerimientos

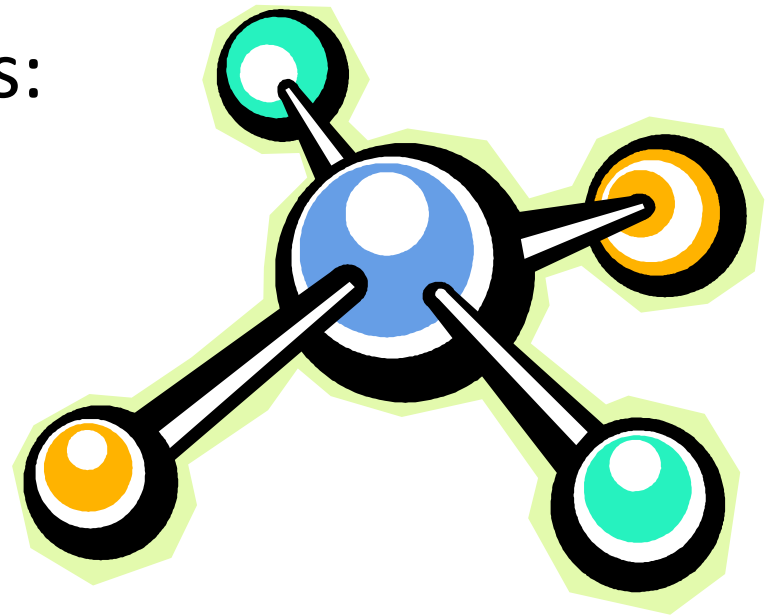
Técnicas

- Modelamiento conceptual
- Categorizar requerimientos
- Priorizar requerimientos
- Negociar requerimientos

Modelamiento Conceptual

Conceptos

- Crear modelos que representan el mundo real
- Objetivo: ayudar a comprender el mundo real
- Varios tipos de modelos:
 - Flujo de datos
 - Procesos de negocio
 - Objetos
 - Etc.



Tipos de Modelos | ¿Cómo escoger?

- Depende de la naturaleza del problema:
 - Nivel de rigurosidad exigido
 - Tipo de aplicación: negocios, tiempo real
- Experiencia del equipo de desarrollo
- Limitaciones o exigencias del cliente o de la organización
- Disponibilidad de recursos (herramientas)

Modelamiento Interfaces

- Modelamiento del contexto del software
 - Tipos de interfaces:
 - Entre componentes
 - Entre sistemas
 - Con los usuarios
 - Diagramas UML
- Evitar conflictos
 - Garantizar transparencia



Carga Transaccional

Factores

- Número de usuarios
- Número de usuarios concurrentes
- Volumen de datos
- Tipo de datos

Requerimientos Funcionales

Datos

- ¿Qué datos debe manejar nuestra aplicación?
 - Tipo
 - Cantidad
 - Frecuencia de uso
 - ¿Estructurado?
- Importante para:
 - Definir almacenamiento
 - Procesamiento
 - Recuperación



Requerimientos No Funcionales

¿Cuándo analizarlas?

- ERROR: Preocuparse en etapas finales
- CORRECTO: Lo más temprano posible
 - Un correcto diseño depende de ello

Requerimientos | Clasificación

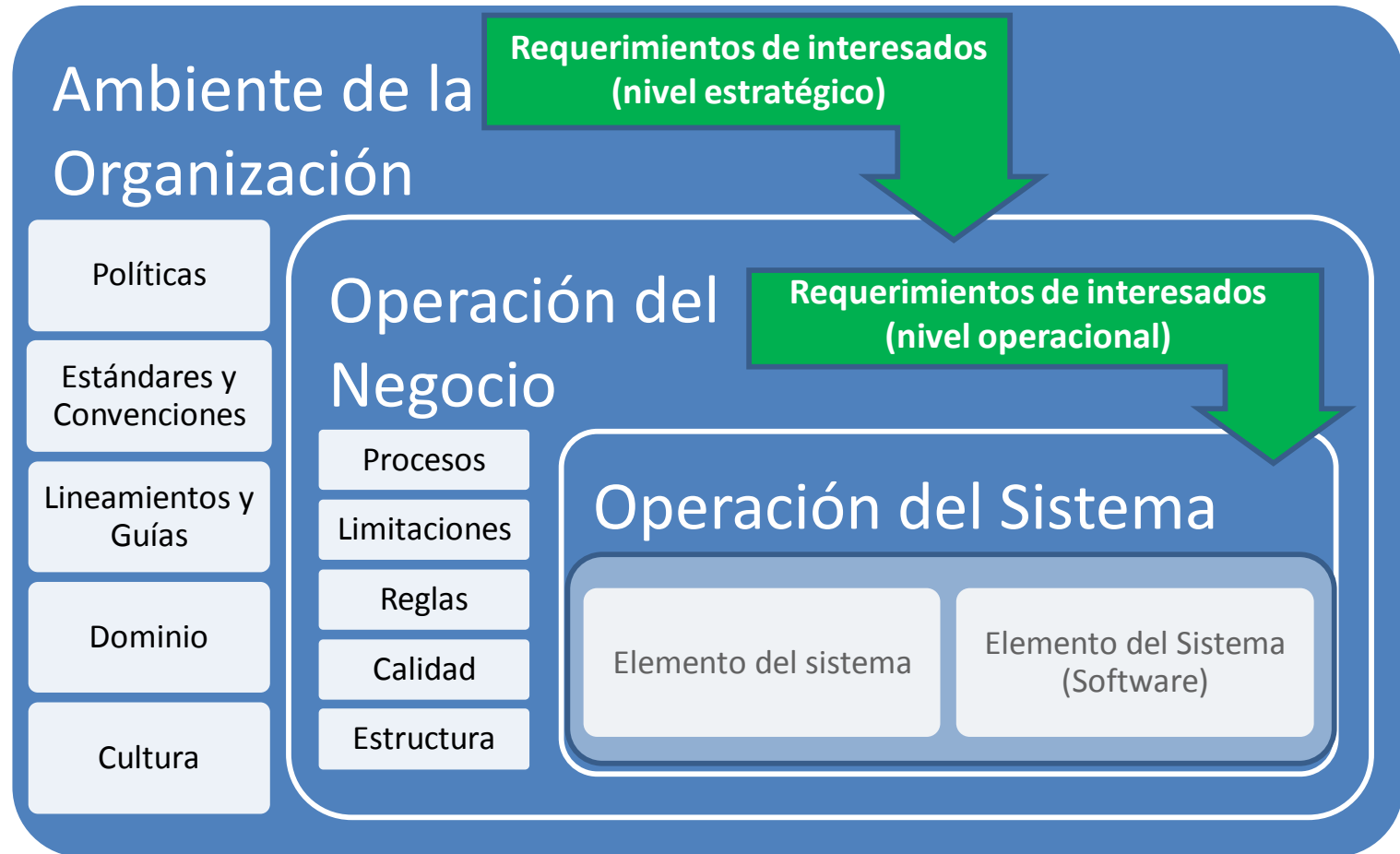
- Funcionales o No Funcionales (NFR)
- Origen: exigencias, sub requerimientos, hecho común a todos los requerimientos
- Alcance del requerimiento: global, específico
- Volatilidad, estabilidad
- Nivel de abstracción:
 - Alto nivel: nivel de usuario
 - Bajo nivel: nivel de sistema

Requerimientos

Basado en IEEE 29148

Ambiente Externo

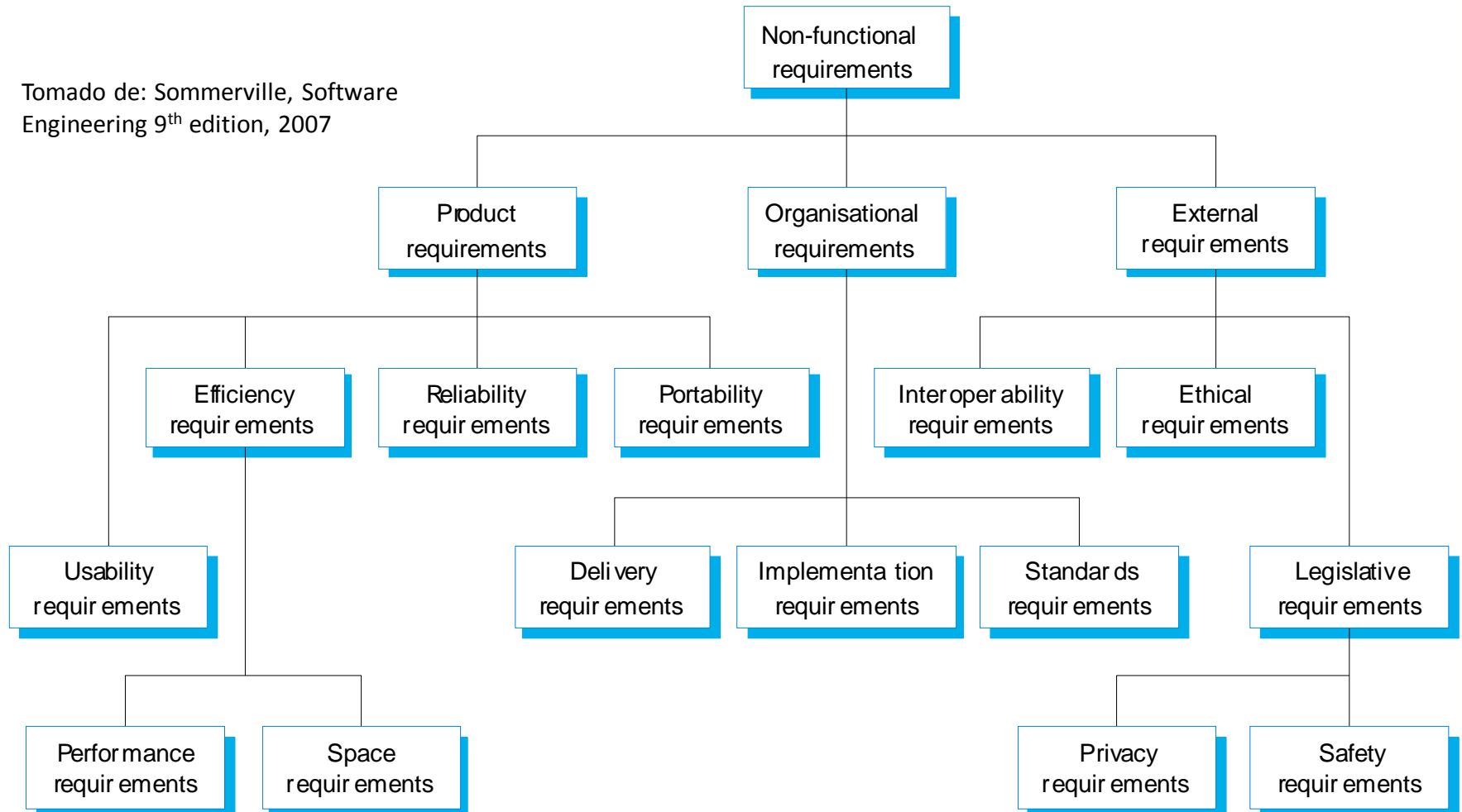
Mercado
Regulaciones
Sociedad
Tecnología
Sector laboral
Competencia
Ambiente natural
Estándares
Cultura



Requerimientos No Funcionales

Clasificación

Tomado de: Sommerville, Software Engineering 9th edition, 2007




Requerimientos | Priorizarlos

- Poder de decisión de actores clave
- Intereses del negocio
- Impacto
- Factibilidad (recursos)
- Tiempos



Prioridad | Ágil

- Recomendable:
 - 10 – Más importante
 - 20
 - 30
 - 40
 - 50 – Menos importante
- El cliente define las prioridades
 - Proveerle información: dependencias
- Pueden reajustarse a lo largo del proyecto

Requerimientos | Negociación

- Objetivo: resolver conflictos
- Actores con necesidades incompatibles
- Conflictos entre requerimientos y recursos
- Entonces, negociar con los actores, procurando consenso
- Solucionarlo con el cliente

Documentar Requerimientos

Especificaciones

- SRS: Software Requirements Specifications
- Debe incluir:
 - Los requerimientos de usuario (alto nivel)
 - Los requerimientos del sistema (bajo nivel)
- ¿A quién va dirigido?
 - A todos: desde alta gerencia, hasta desarrollador
- La comunicación efectiva es crítica:
 - Cuidar el lenguaje
 - Decidir correctamente el nivel de detalle

SRS | Estándar IEEE

- Estándar: ISO/IEC/IEEE 29148:2011
 - Systems and software engineering — Life cycle processes — **Requirements engineering**
- Exige trazabilidad (origen)
 - Consistencia con los requerimientos de sistema
- Describir:
 - Impacto y prioridad de cada requerimiento
 - Limitaciones y contexto
 - Cómo verificar los requerimientos

- El estándar IEEE 29148 recomienda:
 - Introducción (propósito, alcance, definiciones)
 - Referencias
 - Requerimientos específicos
 - Verificación (enfoques y métodos para calificar requerimientos específicos)
 - Apéndices (dependencias, abreviaturas)

SRS | Introducción

- Propósito
- Alcance (productos, que hacen, objetivos)
- Resumen del Producto
 - Perspectiva del producto (relación con otros productos)
 - Funciones del producto (describir las principales)
 - Características del usuario
 - Limitaciones (HW, regulatorias, seguridad, etc.)
- Asunciones y Definiciones

SRS | Referencias

- Lista de todos los documentos mencionados o utilizados en las SRS
- Cada documento debe estar identificado:
 - Nombre
 - Autores
 - Número de reporte
 - Fecha de publicación
 - Organización que publica

SRS

Requerimientos específicos

- Interfaces externas (entradas y salidas)
- Funciones (acciones para atender entradas y producir salidas)
- Usabilidad (criterios de satisfacción)
- Rendimiento (performance: estáticos, dinámicos)
- Base de datos (análisis lógico)
- Limitaciones de diseño (regulatorias, estándares, proyecto)
- Atributos del sistema de software (los ilities)
- Información de soporte (formatos, resultados de estudios y encuestas, antecedentes, etc.)

Verificar Requerimientos

¿En qué consiste?

- Comprobar que el SRS:
 - Está de acuerdo a los estándares de la organización
 - Es comprensible, consistente, y completo.
- Varios tipos de actores tanto de parte del cliente como del equipo desarrollador deberían participar.

Validar los Requerimientos

Conceptos básicos

- Validar: asegurar que el equipo desarrollador ha comprendido bien los requerimientos
 - Se ha documentado el software correcto
- Se lo debe realizar varias veces durante las actividades de requerimientos
- Objetivo: garantizar validez antes de asignar recursos a un requerimiento

Validar Requerimientos

Técnicas

- Revisiones: inspección una vez terminado el SRS
 - Errores, asunciones erradas, falta de claridad
 - Ágil: luego de definir un nuevo sprint backlog
- Prototipos: por software, en papel, etc.
 - Para validar y levantar nuevos requerimientos
- Validación de modelos: si fueron elaborados durante la etapa de análisis
- Diseñar pruebas de aceptación
 - Si no es posible hacerlo, requerimiento no es validable

Pruebas de Aceptación

Ágil

- Descripción del comportamiento de un software
- Entonces la base es la historia de usuario
- Pero: deseable automatizarlo
- Necesario: criterios de aceptación
 - Sin perder la claridad para el cliente
 - Resultado binario: pasa o no pasa



Criterios de Aceptación

Escenarios de uso

- Solución: añadir escenarios de uso
- Formato:
Dado algún contexto inicial (precondiciones),
Cuando un evento ocurre (acción que se da),
Entonces asegurar algunas salidas (consecuencias) .
- Se pueden añadir dos palabras claves **Y**, **Pero**
- Herramientas: JBehave, Cucumber, **Calabash**

Escenario de Uso | Ejemplo

- **As a** estudiante
- **I want to** registrarme en una materia X en línea
- **So that** no ir a la universidad para registrarme
- **Given** periodo de registros está abierto
- **And** existe cupo en materia X
- **And** prerequisites de materia X están aprobados
- **And** no se ha superado número máximo de créditos por semestre
- **When** estudiante solicita registro en materia X
- **Then** añadir materia X al registro de estudiante
- **And** reducir en 1 el cupo de materia X
- **And** actualizar el número de créditos registrados del estudiante