“Requerimientos”

Sebastian Uchitel

Ingeniería de Software 1, Agosto 2016

Agenda

- Una introducción ingenua a “Requerimientos”
- La dura realidad
- Ingeniería de Requerimientos

“Requerimientos”

La versión simplificada e ingenua

Que es un requerimiento?

"Una enunciación de una función que un software debe proveer."

De dónde vienen?

"De los usuarios."

Cuando hay que conseguirlos?

"Antes de programar. Incrementalmente"

Cómo los documentamos?

"En algún lenguaje. Puede ser lenguaje natural informal, gráfico o matemático."

Cuando paramos?

"Cuando el usuario no tiene más requerimientos."


REALITY
CHECK
AHEAD

Standish Group Report

MODERN RESOLUTION FOR ALL PROJECTS

	2011	2012	2013	2014	2015
SUCCESSFUL	29%	27%	31%	28%	29%
CHALLENGED	49%	56%	50%	55%	52%
FAILED	22%	17%	19%	17%	19%

The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011–2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.

Project Success Factors	% of Responses
1. User Involvement	15.9%
2. Executive Management Support	13.9%
3. Clear Statement of Requirements	13.0%
4. Proper Planning	9.6%
5. Realistic Expectations	8.2%
6. Smaller Project Milestones	7.7%
7. Competent Staff	7.2%
8. Ownership	5.3%
9. Clear Vision & Objectives	2.9%
10. Hard-Working, Focused Staff	2.4%
Other	13.9%

Project Challenged Factors	% of Responses
1. Lack of User Input	12.8%
2. Incomplete Requirements & Specifications	12.3%
3. Changing Requirements & Specifications	11.8%
4. Lack of Executive Support	7.5%
5. Technology Incompetence	7.0%
6. Lack of Resources	6.4%
7. Unrealistic Expectations	5.9%
8. Unclear Objectives	5.3%
9. Unrealistic Time Frames	4.3%
10. New Technology	3.7%
Other	23.0%

Project Impaired Factors	% of Responses
1. Incomplete Requirements	13.1%
2. Lack of User Involvement	12.4%
3. Lack of Resources	10.6%
4. Unrealistic Expectations	9.9%
5. Lack of Executive Support	9.3%
6. Changing Requirements & Specifications	8.7%
7. Lack of Planning	8.1%
8. Didn't Need It Any Longer	7.5%
9. Lack of IT Management	6.2%
10. Technology Illiteracy	4.3%
Other	9.9%

No estamos aprendiendo….

Standish Group Report

	1994	1998
Exitosos	16%	26%
Con problemas	53%	46%
Cancelados	31%	28%

- Causa percibida de éxito y fracaso (Top 3)

	Exitoso	Con problemas	Cancelado
1.	Usuarios involucrados	Falta de input de los usuarios	Requerimientos incompletos
2.	Apoyo de gerencia ejecutiva	Requerimientos incompletos	Falta de input de los usuarios
3.	Clara descripción de requerimientos	Requerimientos cambiantes	Falta de recursos

Resultados similares en otros estudios ...

- Relevamiento de 3800 organizaciones europeas, 17 países

  Los problemas mayores en software son...

  - Especificación de requerimientos
    > 50% respuestas
  - Gestión de requerimientos
    > 50% respuestas

  (European Software Institute, Technical Report 1996-TR95104)

Dificultades Esenciales de hacer "Requerimientos"

Completitud

Pertinencia

Estamos construyendo el software equivocado…

… que no es lo mismo que estar construyendo software con errores.

No Me Importa

Soy Programador	Soy Diseñador	Soy Tester	Soy Líder de Proyecto	Soy Dueño
… me pagan para…				
hacer un programa que satisface los requerimientos	hacer un diseño que garantiza los requerimientos	chequear que el software satisface los requerimientos	dirigir un equipo que desarrollará el software que satisface los requerimientos	para hacer lo que pidió el cliente

Si te importa…

Soy Programador

Soy Diseñador

Soy Tester

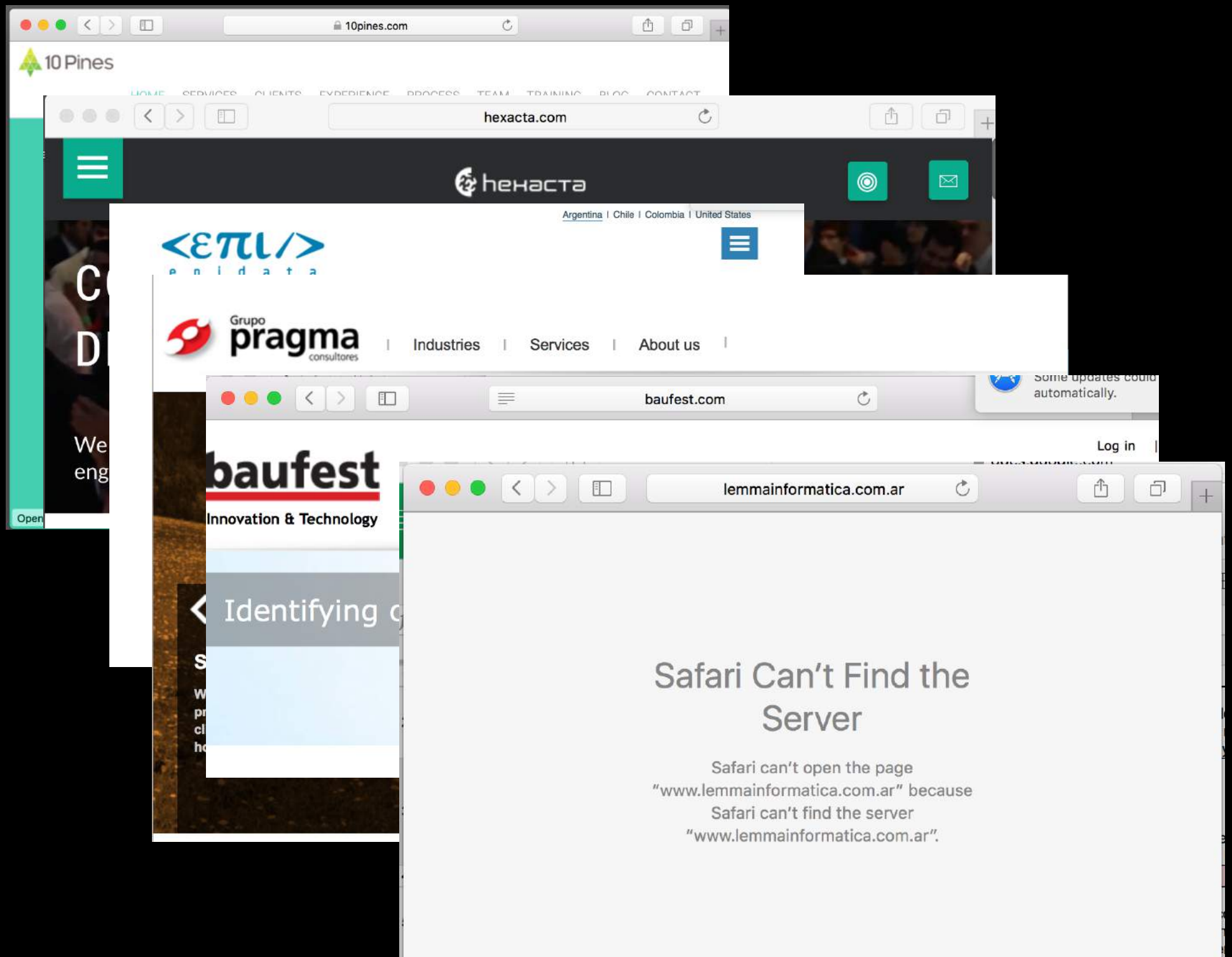
Soy Líder de Proyecto

Soy Dueño

… me pagan para…

Resolver Problemas

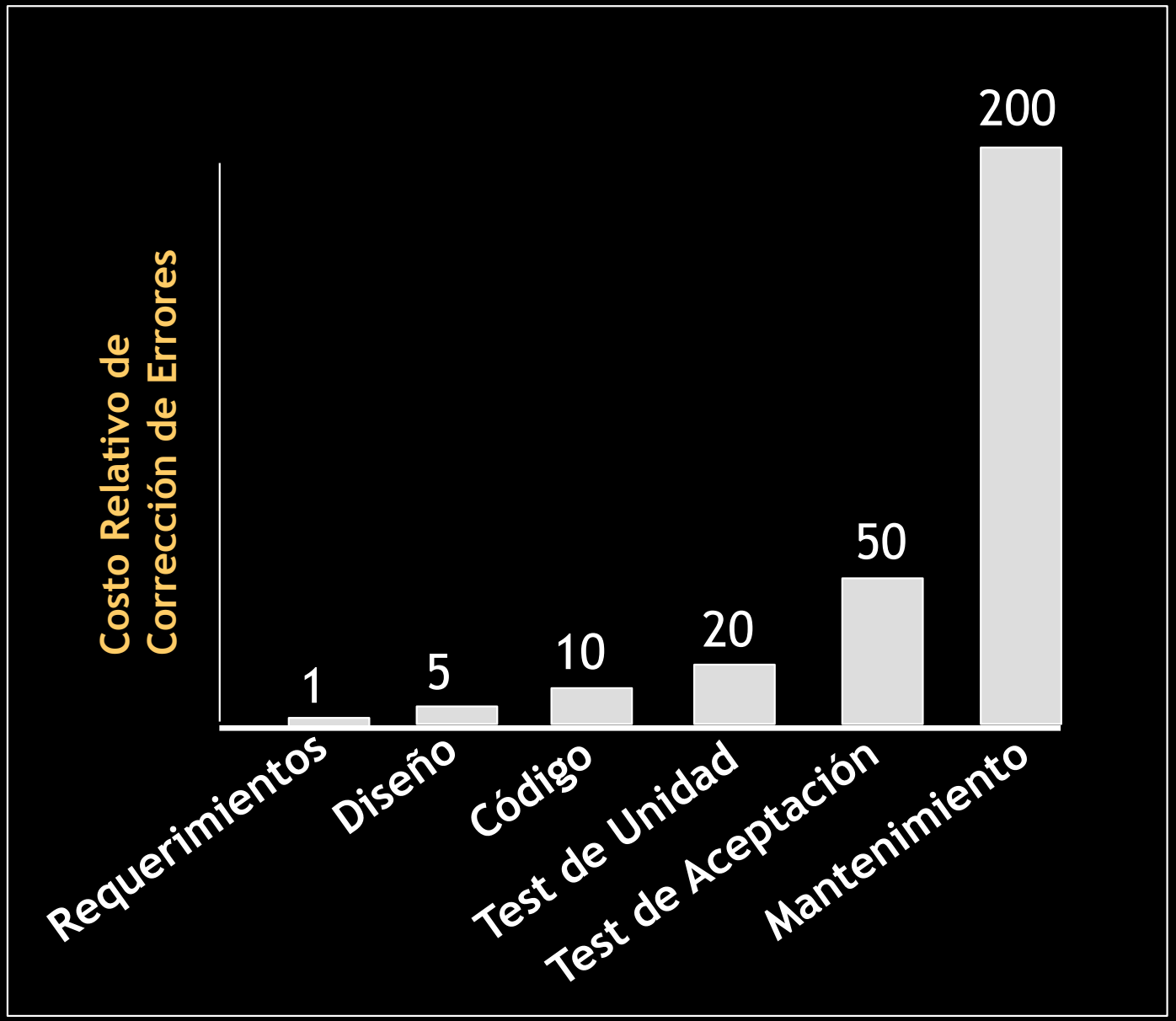
Hacer lo que me dicen


10pines.com
10 Pines
hexacta.com
hexacta
Argentina | Chile | Colombia | United States
Grupo pragma consultores
Industries
Services
About us
baufest.com
Some updates could automatically.
Log in
baufest
Innovation & Technology
Identifying
lemmainformatica.com.ar
Safari Can't Find the Server
Safari can't open the page "www.lemmainformatica.com.ar" because Safari can't find the server "www.lemmainformatica.com.ar".

Aceptar malos requerimientos es un problema ético también.

- Vidas humanas
  - Armas y sistemas de defensa
  - Sistemas de transporte
  - Sistemas Médicos
- Temas de Sociedad
  - E-Government
  - Sistema de Votación
  - Sustentabilidad

El Costo de Corrección Tardía



B.W. Boehm, "Software Engineering Economics", Prentice Hall, 1981.
B.W. Boehm, "Verifying and Validating Software Requirements and Design Specifications," IEEE Software, 1984

La voz de la experiencia...

- "…hardest, most important function of Software Engineering is the iterative extraction & refinement of requirements"
  (F. Brooks, No Silver Bullet: Essence and Accidents of Software Engineering, 1987)
- "Requirements is one of the six key Software Engineering activities…"
  D. L. Parnas , "Inaugural Lecture" U. Limerick, 2003)
- "the requirements for a system do not rise naturally; instead, they need to be engineered…"
  (T. E. Bell, T. A. Thayer. Software Requirements: Are They Really a Problem? ICSE 1976)

Resumen

- Hacer “Requerimientos” parece simple
- Pero muchos lo hacen mal
- Es causante principal de fracasos, sobre-costos y retrasos
- Lo difícil es completitud y pertinencia
- Nos debería importar

“Ingeniería de Requerimientos”

Propósito

Software se desarrolla propósito que es relativo a actividades humanas

$$Q = f(S, P)$$

Ingeniería de Requerimientos trata de la identificación del propósito.

Ingeniería de Requerimientos

- Trata de la identificación del propósito.
- Si no conocemos el propósito no podemos construir un sistema de calidad
- Un entendimiento pobre del propósito lleva a sistemas de baja calidad

Una birome que funcione en gravedad 0

El Mundo y la Maquina

(M. Jackson & P. Zave, 1995)

(M. Jackson, Software Requirements and Specifications, ACM Press Books, 1995)

- Abstracción que permite formular rigurosamente algunas nociones fundamentales de la Ingeniería de Requerimientos
- Conceptos:
  - Fenómenos
  - El mundo
  - La máquina
  - Aserciones

Fenómenos

- Un hecho, situación o evento cuya existencia puede observarse.
- Ejemplos para un sistema de despacho de ambulancias
  - la ocurrencia de incidentes
  - el reporte de incidentes por parte del público
  - la codificación de los detalles del llamado en el sistema
  - el almacenamiento de los datos en la base de datos
  - el cómputo de la ambulancia más cercana al incidente
  - la asignación de una ambulancia a un incidente
  - la propagación de la asignación al depósito de ambulancias
  - el arribo de la ambulancia al lugar del incidente

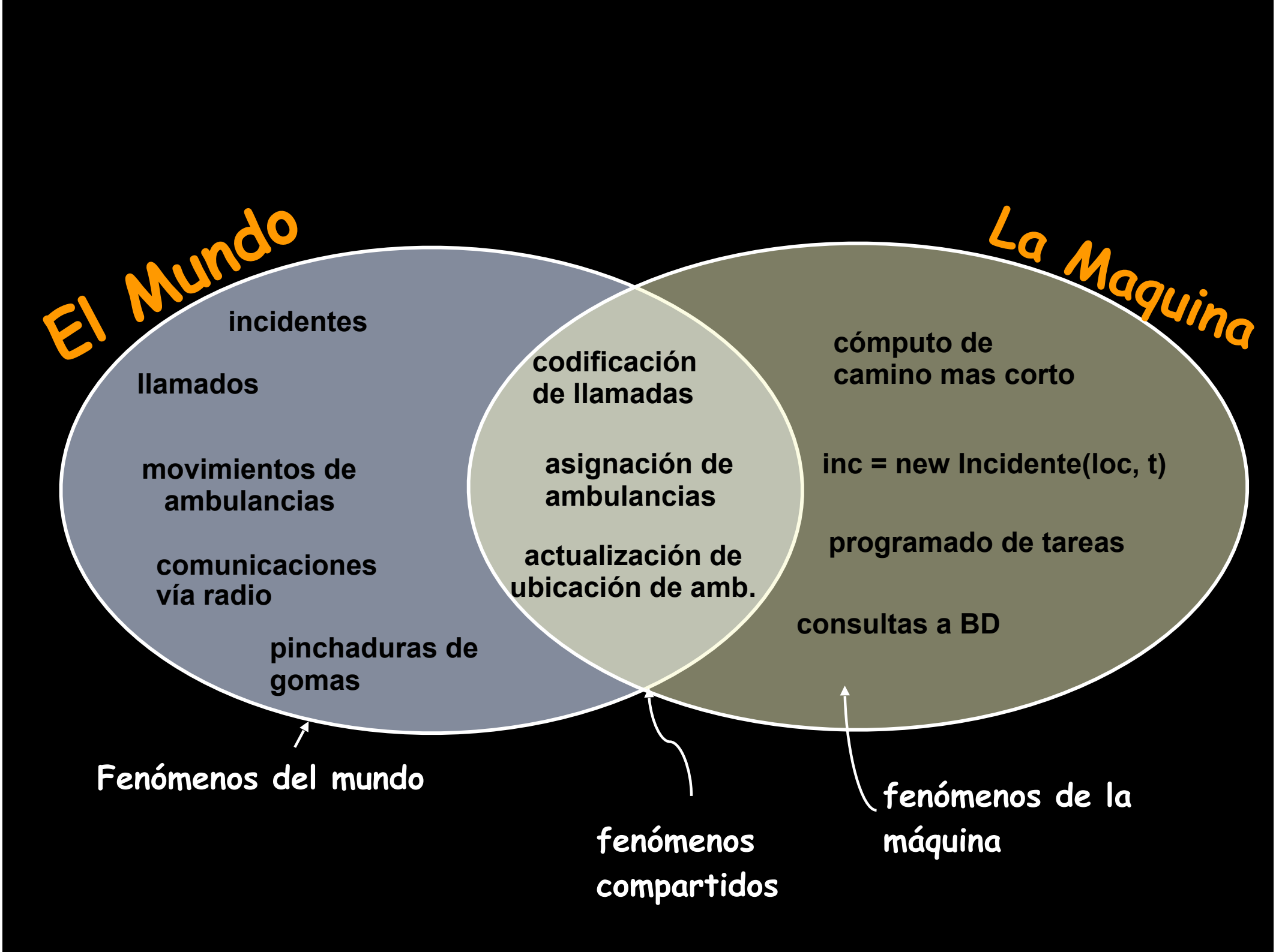
El Mundo vs la Máquina

- Máquina = porción del sistema a desarrollar o modificar
  - típica, pero no necesariamente, el software y hardware
- Mundo = porción del mundo afectado por la máquina
  - Conocido también como el ambiente o el entorno
- El mundo y la máquina se tocan en la interfaz de la máquina.
- El propósito de la máquina está en el mundo

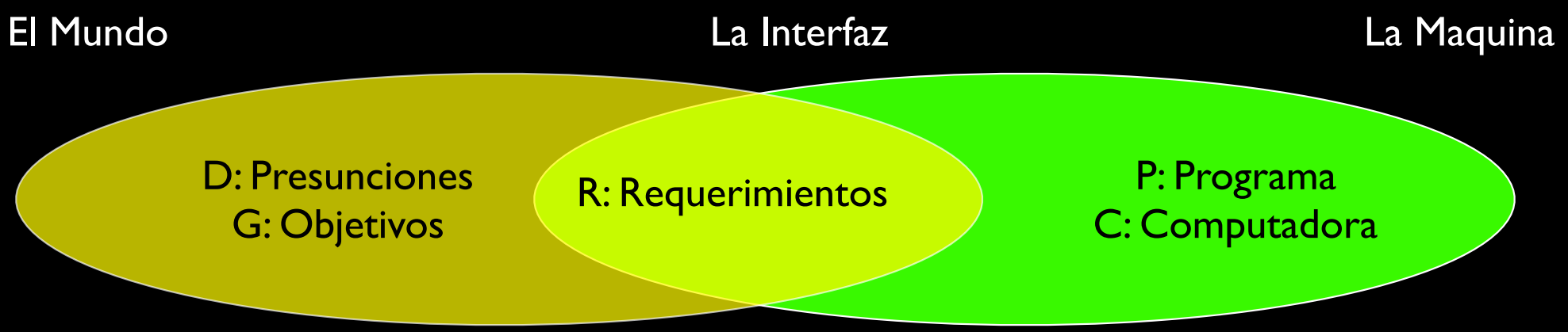
IR, el Mundo y la Máquina

- Ingeniería de Requerimientos trata de los fenómenos que ocurren en el mundo
  - la ocurrencia de incidentes
  - el reporte de incidentes por parte del público
  - la codificación de los detalles del llamado en el sistema
  - la asignación de una ambulancia a un incidente
  - el arribo de la ambulancia al lugar del incidente

  y no trata de los fenómenos que ocurren dentro de la máquina
  - el almacenamiento de los datos en la base de datos
  - el cómputo de la ambulancia más cercana al incidente
  - la propagación de la asignación al depósito de ambulancias


El Mundo
incidentes
llamados
movimientos de ambulancias
comunicaciones vía radio
pinchaduras de gomas
codificación de llamadas
asignación de ambulancias
actualización de ubicación de amb.
La Maquina
cómputo de camino mas corto
inc = new Incidente(loc, t)
programado de tareas
consultas a BD
Fenómenos del mundo
fenómenos compartidos
fenómenos de la máquina

- El Mundo y la Máquina -



Distintos tipos de aserciones

- Objetivo: Prescripción sobre fenómenos en el mundo
- Presunción: Descripción sobre fenómenos en el mundo (el dominio del problema)
- Requerimiento: Prescripción sobre fenómenos en la interfaz

(M. Jackson, Software Requirements and Specifications, ACM Press Books, 1995)

Ej. Sistema de Despacho de Ambulancias

- Objetivo
  - Para cada llamado urgente que reporta un incidente, una ambulancia deberá arribar a la ubicación del incidente dentro de los 14 minutos.
- Presunciones del dominio
  - Por cada llamado, los detalles del incidente son codificados correctamente.
  - Cuando una ambulancia es movilizada, alcanzará la ubicación correspondiente en el menor tiempo posible.
  - La ubicación de cada ambulancia es conocida vía GPS
  - El personal de ambulancia informa correctamente disponibilidad vía las terminales móviles de datos que están colocadas en las ambulancias
- Requerimiento
  - Cuando un llamado reportando un nuevo incidente es codificado, el sistema de despacho de ambulancias deberá movilizar la ambulancia disponible mas cercana de acuerdo a la información disponible vía los GPS y Terminales Móviles de Datos de las ambulancias.

V&V a la Michael Jackson

El modelo de Jackson permite formular...

- dos criterios de verificación
  - ¿Los requerimientos (R) de la máquina satisfacen los objetivos (G) dadas las suposiciones acerca del dominio (D)?

    R && D => G
  - El programa (P) ejecutando sobre el hardware (C) satisface los requerimientos (R)?

    P && C => R
- varios criterios de validación
  - Tenemos todos los objetivos? Son todos válidos?
  - Todos las presunciones del dominio son verdaderas?

Completitud de Requerimientos

- Los requerimientos R son completos si
  1. R garantiza G presumiendo D

     R && D => G
  2. G captura adecuadamente el problema a resolver
  3. D representa presunciones válidas acerca del mundo.

Observación: Los objetivos suelen ser menos, más fáciles de definir y más estables que los requerimientos.

Observación: Presunciones invalidas suelen ser motivo de fallas (c.f. Subte Nueva York)

Pertinencia de Requerimientos

- Los requerimientos R son pertinentes si eliminando un requerimiento, aún son completos.

P2P e-Commerce

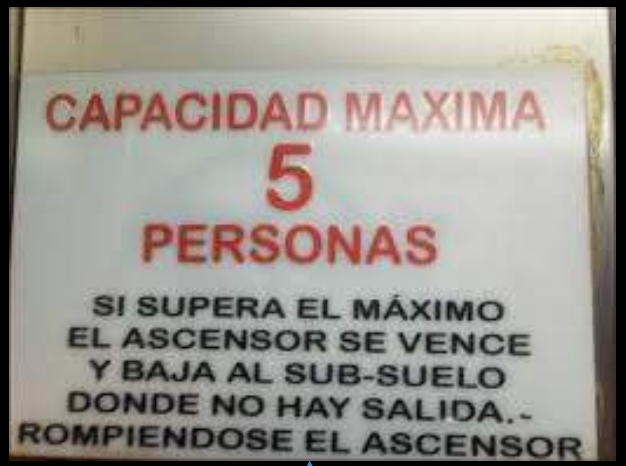
Interfaz Mundo Máquina

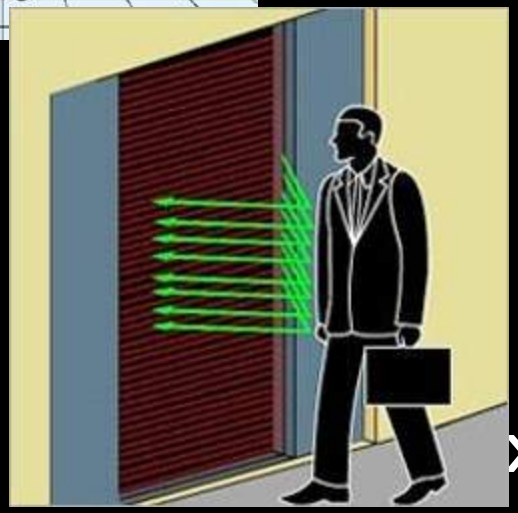
- Juega un rol clave
- Ingeniería de Requerimientos incluye
  - Exploración de ternas (interfaz, R y D) tal que R && D implica G
  - Evaluar las alternativas, para seleccionar las mas apropiada

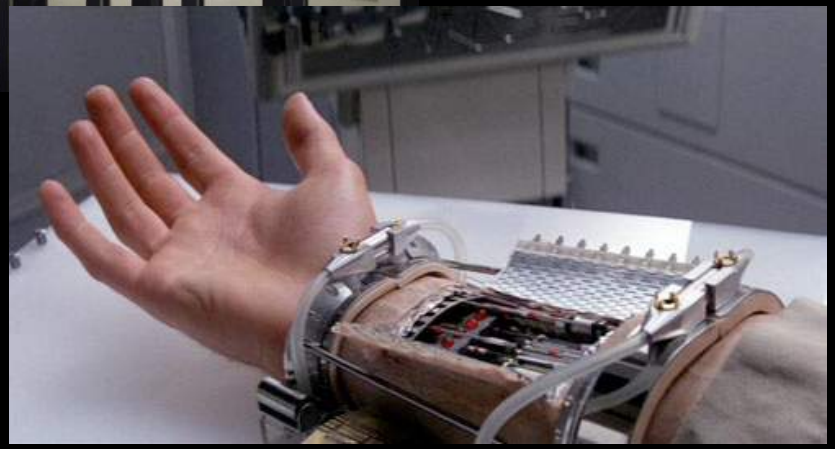
Análisis de Costo

Análisis de Riesgo

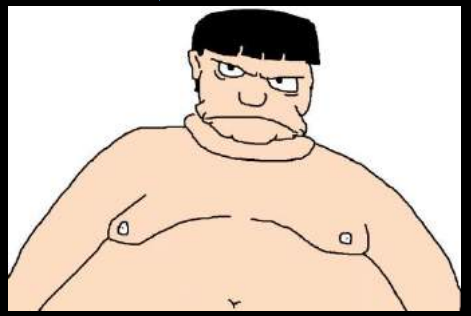
Interfaces para un Controlador de Ascensores



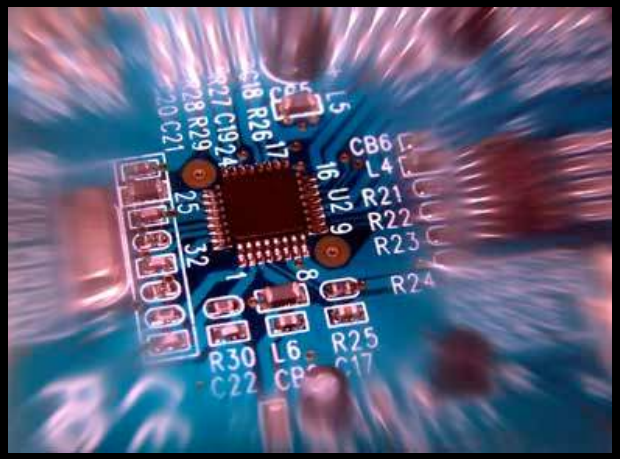




x



Software Embebido



- En hardware
- En actividades humanas
- En un mundo físico

Sistemas intensivos en Software = Software + Hardware + Entorno

El problema (y solución) es el resultado del comportamiento emergente de la interacción entre los componentes del sistema

Ingeniería de Requerimientos ~ Diseño de Sistemas Intensivos en Software

Resumen

- Objetivos vs. Requerimientos vs Presunciones
- Entender el problema (los objetivos) es clave para razonar sobre completitud y pertinencia de requerimientos
- Correr la interfaz, cambia las presunciones y por lo tanto el riesgo y el costo.
- Muchos proyectos fallan por errarle al problema o subestimar el riesgo de las presunciones.
- Hacer IR es diseñar sistemas

Taxonomía de Requerimientos

Requerimientos Funcionales

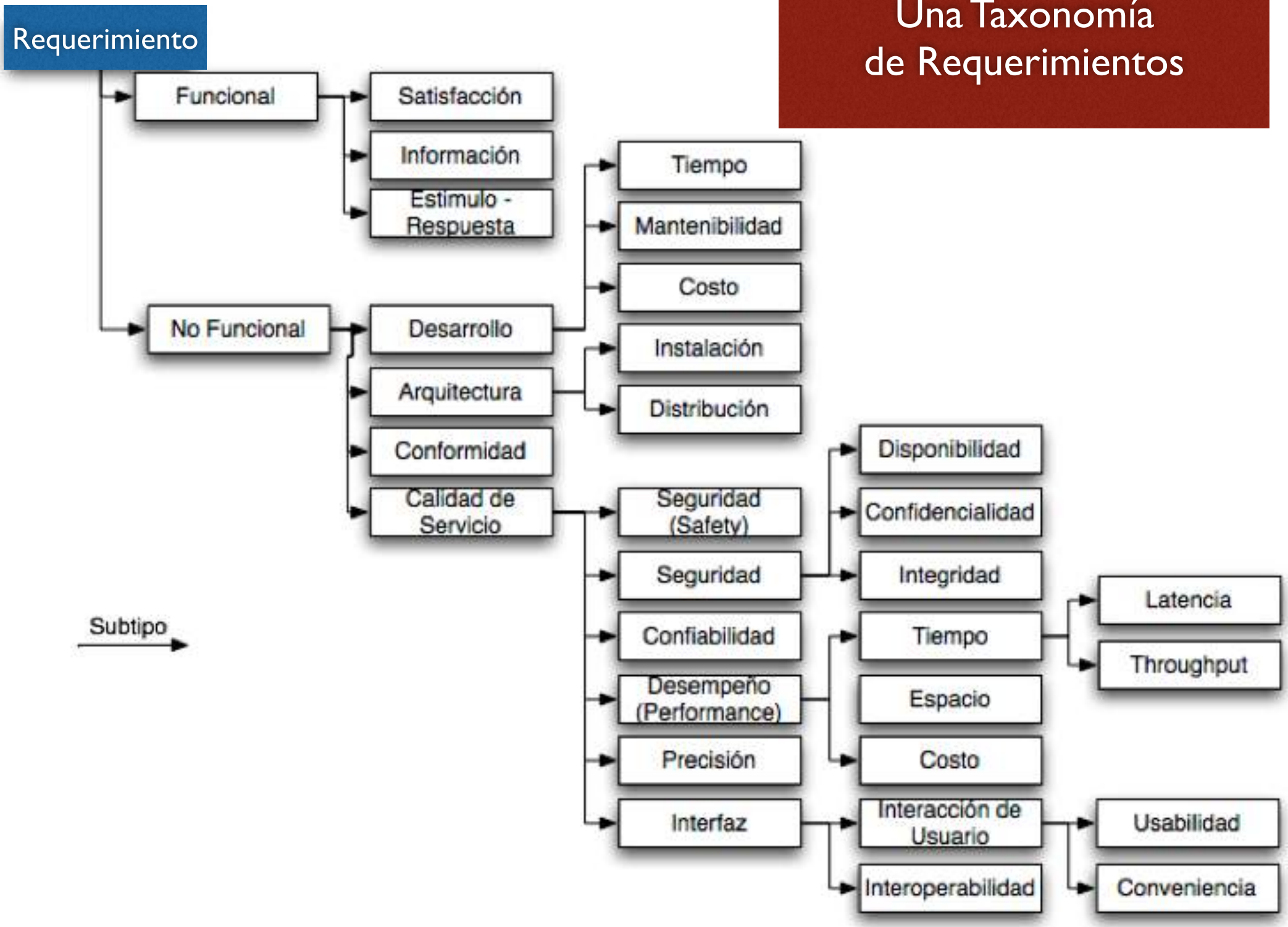
- Funciones o servicios a ser provistos por el sistema
- Derivan en operaciones concretas en la interfaz
- Derivan en módulos particulares del código que los implementan

Requerimientos No Funcionales o de Calidad

- Restricciones adicionales.
- Cómo/Cuán bien deben proveerse los servicios
- Tienden a afectar grandes porciones de funcionalidad (cross-cutting)

Como muchas taxonomías, no suelen definir clases disjuntas ni suelen tener definiciones precisas

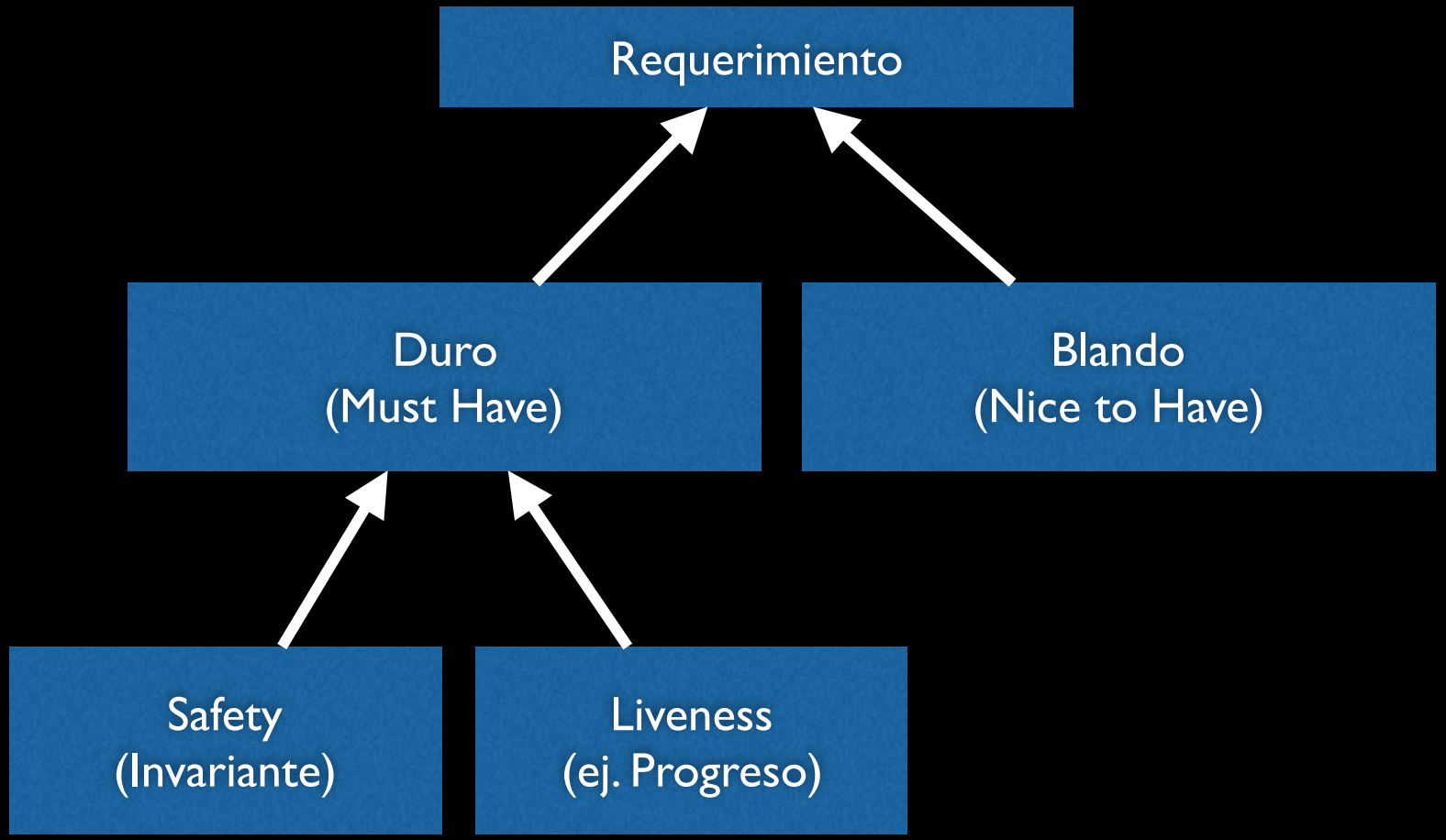
Una Taxonomía de Requerimientos



¿Para qué sirven las Categorías?

- Proveen heurísticas para
  - Elicitar requerimientos. Como checklist
  - Analizar y reutilizar modelos
    - Interacciones típicas entre categorías
    - Técnicas de análisis especializadas
- No deben usarse para
  - Particionar estrictamente los requerimientos
  - Invertir tiempo y esfuerzo excesivo en decidir la categoría de un objetivo

Otra Taxonomía



Requerimientos Blandos vs No Funcionales

- Clasificaciones ortogonales
  - El controlador de velocidad ajustará la velocidad en el caso de exceso de velocidad en menos de 50 milisegundos.
  - El controlador de velocidad ajustará la velocidad en el caso de exceso de velocidad en el menor tiempo posible.

Aserciones Medibles

- Para los de comportamiento debe haber un 'criterio de aceptación'
  - Objetivo: "El software será intuitivo e instrucciones de uso auto-contenidas"
  - Criterio: "95% de los clientes existentes del banco podrán extraer plata y depositar cheques en menos de dos minutos en su primer encuentro con el sistema"
- Para los blandos debe haber un 'criterio de comparación'
  - Objetivo: "El software será lo más intuitivo posible y ..."
  - Criterio: ... basado en tiempo que tardan clientes en extraer plata y depositar cheques en su primer encuentro con el sistema"

- Elección no trivial
  - Stakeholders no suelen ser tan específicos.
  - Fácil de medir no implica correspondencia con objetivo
  - Métricas estándar no necesariamente se corresponden con objetivos

Ejemplos de Criterios de Aceptación

Calidad	Métrica
Velocidad / Performance	Tiempo de respuesta Transacciones por segundo
Facilidad de Uso	Tiempo para completar tareas Tiempo de entrenamiento
Confiabilidad	Tiempo promedio entre fallas, Probabilidad de falla bajo demanda Probabilidad de no disponibilidad
Seguridad Física	Tiempo promedio entre amenaza física, ...
Robustez	Tiempo para restablecer funcionalidad después de falla porcentaje de eventos que producen falla
Portabilidad	Tiempo requerido para portar sistema a otra plataforma

De dónde vienen?

"De los usuarios."

FALSO! (x2)

Los usuarios son sólo una fuente de información

No vienen, hay que elicitarlos

Dónde buscamos información?

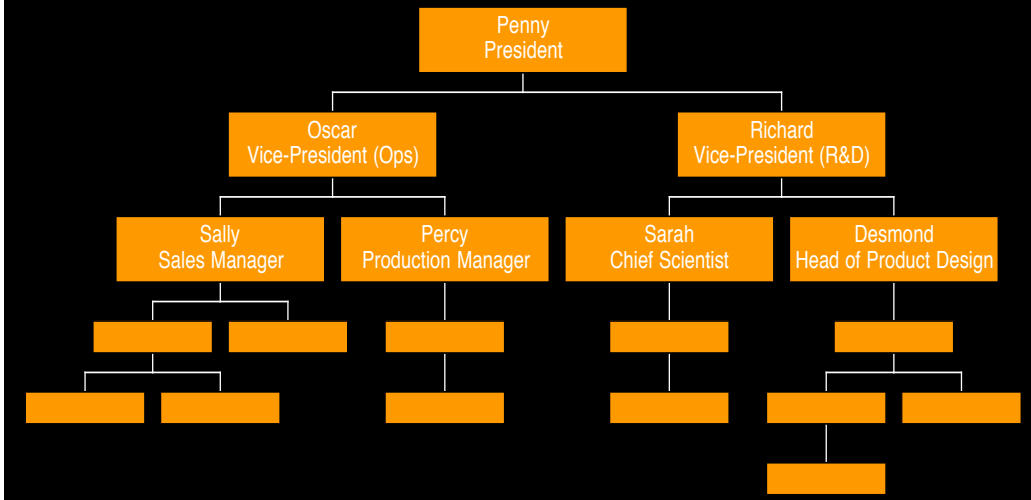
- Stakeholders
- Dominio del Problema
- Documentación

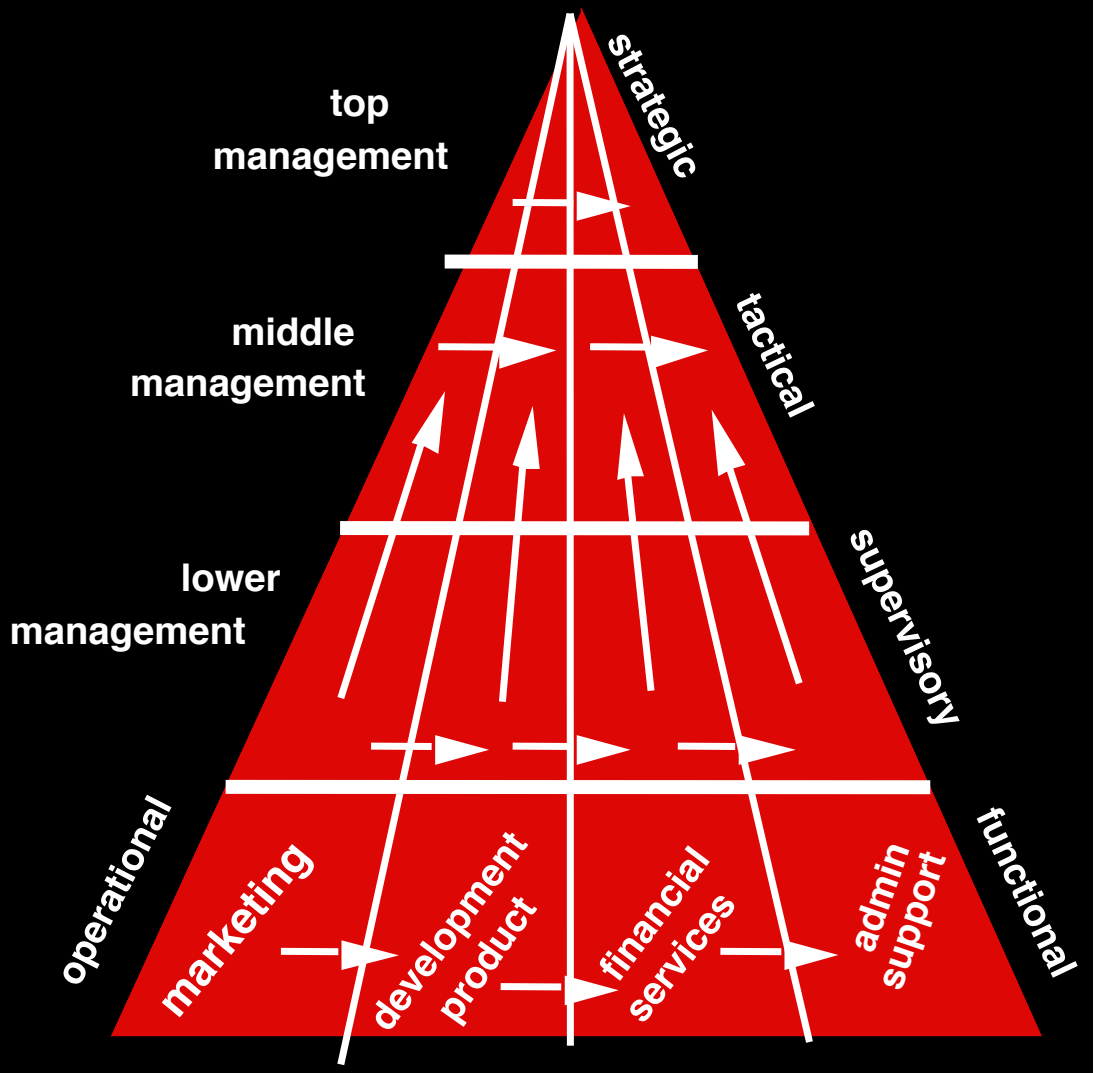
Stakeholders

Aquellos que pueden impactar o ser impactados por el sistema (actual o futuro)

- Usuarios / Roles (frecuente vs ocasional, experto vs novato, ...)
- Stakeholders Financieros (¿quién paga? ¿quién administra el contrato, la plata?)
- Entidades externas (Gobierno, entidades reguladoras, gremios, asociaciones profesionales, competidores)
- Equipo de desarrollo (programadores, testers, gerentes)
- Equipo de soporte (Personal de capacitación, centro de cómputo, servicios de red, …)
- Equipo de Mantenimiento (administradores de redes, BD, sistema operativo, programadores...)
- Stakeholders hostiles (potenciales atacantes)
- Otros (¿Quiénes se benefician? Quienes verán una perdida o deterioro de calidad de trabajo)

Modelos de Stakeholders





Organigramas muestran

- Áreas de responsabilidad
- Líneas de autoridad

Técnicas de Elicitación con Stakeholders

- Técnicas tradicionales
  - Entrevistas, Cuestionarios y Encuestas
  - Focus groups y workshops. Joint Application Design.
- Técnicas de observación
  - Análisis de protocolos
  - Etnografía
- Técnicas basadas en representaciones
  - Prototipación
  - Validación de diagramas
- Técnicas de elicitación de conocimiento

Entrevistas

- Requiere
  - alto grado de especialización
  - mucho trabajo de post-procesamiento
- Múltiples estrategias
  - Ej. Agenda abierta vs. Cerrada
- Excelente para entender contexto del proyecto
- Ojo con
  - Preguntas tendenciosas
  - Conocimiento tácito
  - Sesgos (ej. de motivación incidental)

Cuestionarios y Encuestas

- Recolección rápida de gran número de personas
- Realizable remotamente
- Información recolectada tiene una interpretación simplista
- Cuidado con
  - —Sesgo en muestreo
  - —Cuestionarios mal diseñados

Elicitación Grupal

- "Focus groups"
  - Objetivo: generar consenso
  - Facilitador entrenado y sin agenda propia es fundamental
  - Importante tener "todos" los stakeholders
- Brainstorming
  - Objetivo: generar ideas
  - Altamente des-estructuradas
- Cuidado con:
  - Presión de pares
  - Relaciones de dominación
  - Comportamiento de manada

Técnicas de Observación

- Análisis de Protocolos
  - —Observación de prácticas de trabajo
  - —Observación Directa vs Indirecta
  - —Facilita verbalización de actividades cognitivas
- Etnografía
  - —"Viviendo con los gorilas en la selva"
  - —Activo vs Pasivo
  - —Muy buena para revelar problemas de interacción
  - —Hawthorne effect -> requiere tiempo grande de adaptación

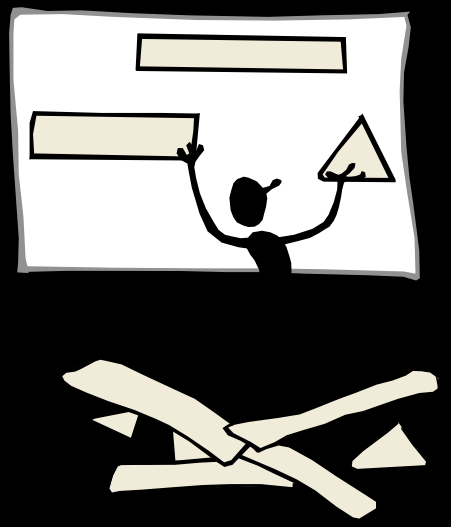
Técnicas de Elicitación de Conocimiento

- Objetivo: descubrir conocimiento de expertos que frecuentemente es tácita, idiosincrática y difícil de articular y justificar
- Algunas técnicas derivadas de la psicología
  - Clasificación
  - Estructuración
  - Matrices atributo/objeto
  - Análisis de proximidad multi-dimensional

G. Rugg and P. McGeorge, “Laddering”, *Expert Systems*, 12(4): 339-346, 1995.
M. Shaw and B. Gaines, “Requirements Acquisition”, *IEE/BCS Software Engineering Journal*, 11(3): 149-165, May 1996
E. Hudlicka, “Requirements Elicitation with Indirect Knowledge Elicitation Techniques: comparison of three methods”, ICRE, 1996

Validación de Diagramas

- Uso de diagramas para facilitar elicitación en entrevistas y sesiones grupales
- Notaciones informales o interpretaciones informales de modelos
  - —Casos de uso
  - —Escenarios
  - —Modelo de objetivos
  - —Maquinas de Estado
  - —Diagramas de flujo de datos/control
  - —Diagramas de Contexto
  - —etc...



Prototipación

- Un prototipo es un borrador de (una parte del) sistema
  - No necesariamente software
- Tipos de prototipos
  - funcional, interfaz de usuario, factibilidad técnica
  - horizontal vs vertical
  - Descartable vs. evolucionable
- Buenos para validación y elicitar información tácita
- Desventajas
  - Número limitado de aspectos que son prototipables.
  - Genera falsas expectativas.
  - Foco temprana en una solución puede dificultar exploración de soluciones alternativas.
  - Costo

Dificultades

- Capacidad de observación limitada
  - — Costo, Visión Operacional
  - — Probe-Hawthorne effect
- Problemas de comunicación
  - — gente con antecedentes y lenguajes distintos
- Múltiples fuentes
  - — muchos stakeholders,
  - — fuentes pueden estar en conflicto
- Fuentes de no disponibles
  - — “todo el mundo lo sabe...”
  - — Stakeholders ocupados y/o no interesados

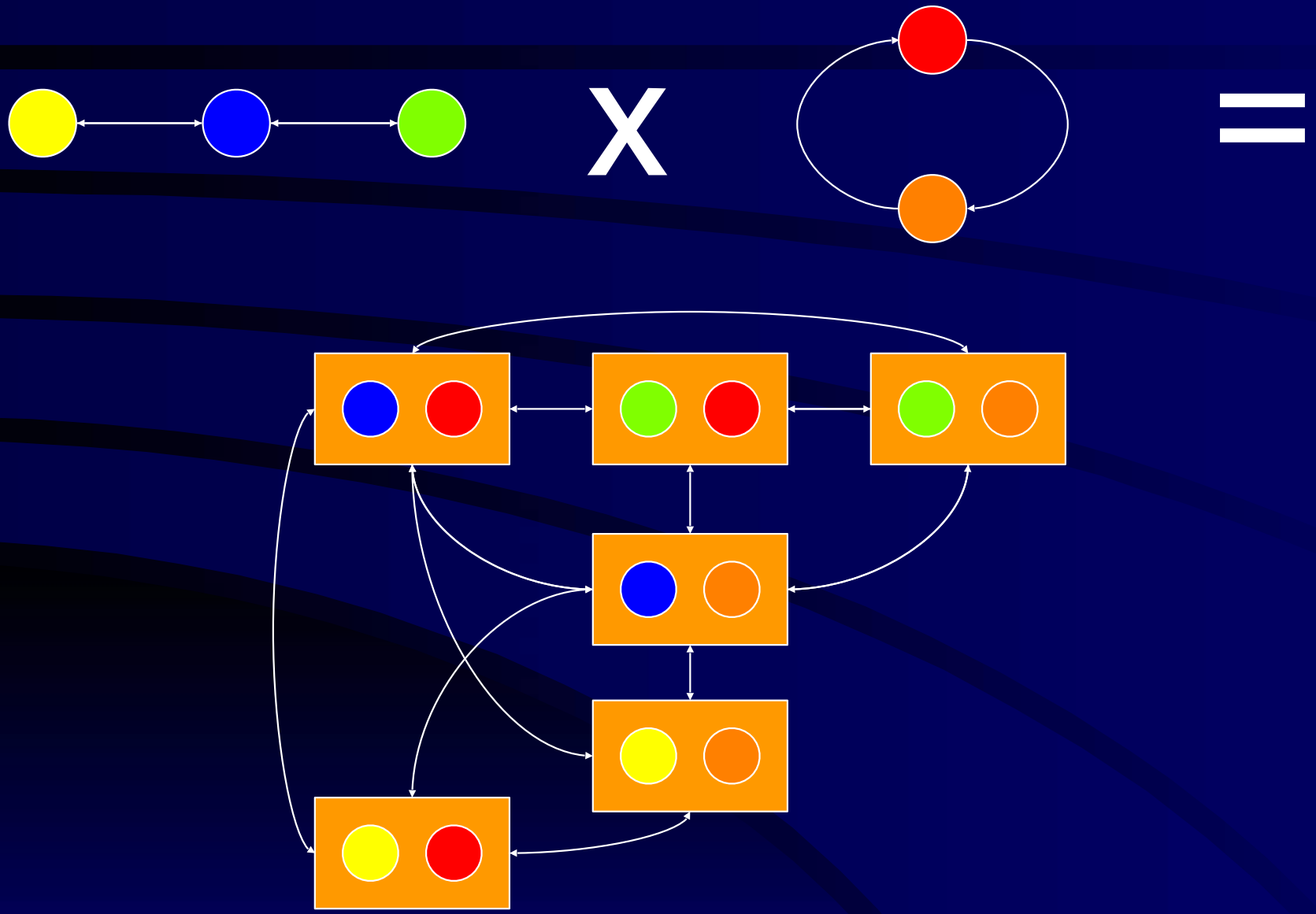
Dificultades

- Aspectos políticos
  - — Libertad y conveniencia de hablar...
- Conocimiento tácito
  - — Dificultad de racionalización de comportamiento propio
  - — Haz lo que digo y no lo que hago...
- Cambios
  - — estructura organizacional, gente, necesidades y prioridades, ...
- Sesgo
  - — Motivación incidental, Observacional, Notacional,

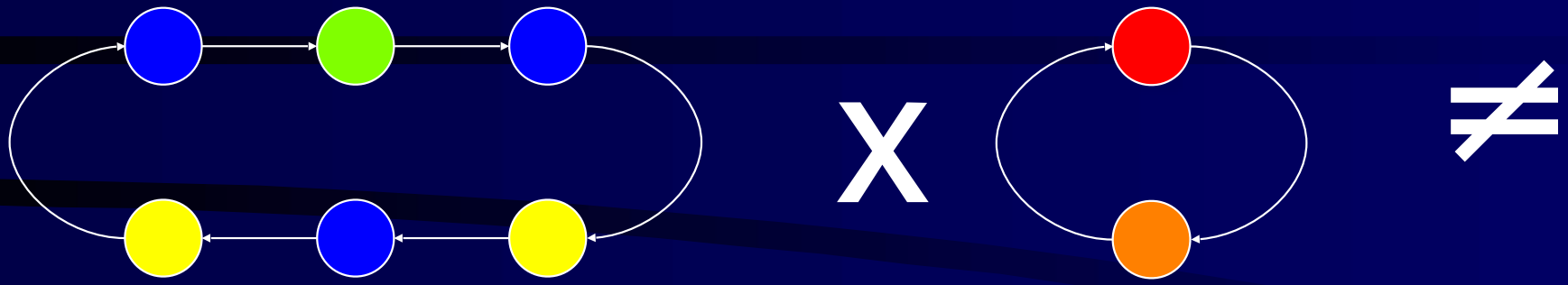
"Elicitación es una actividad
que presenta problemas fundamentalmente difíciles"

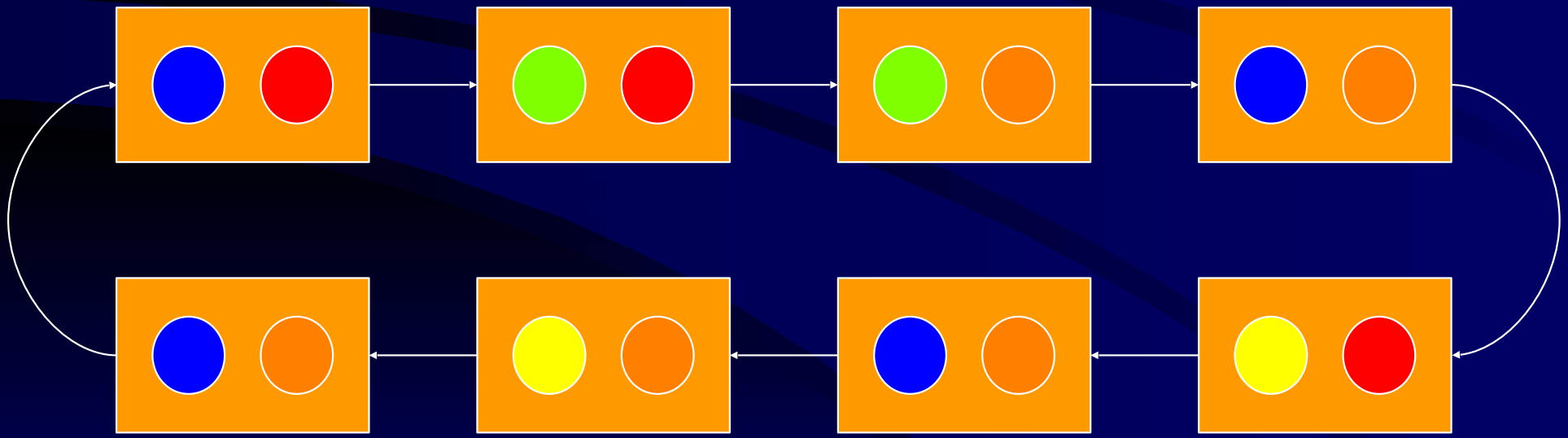
Hagamos una experiencia de primera mano....

¿Una Descripción Posible?

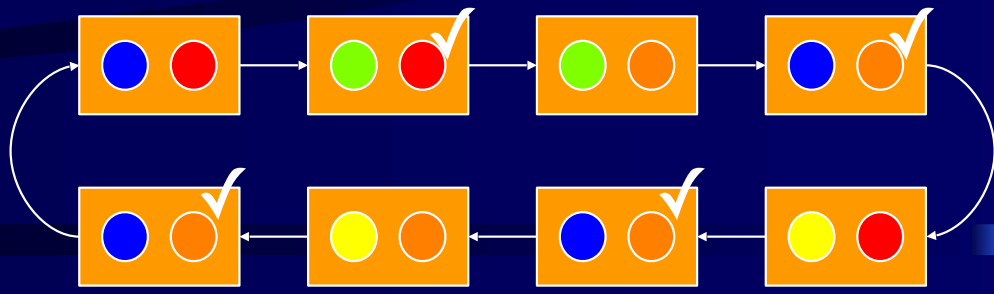


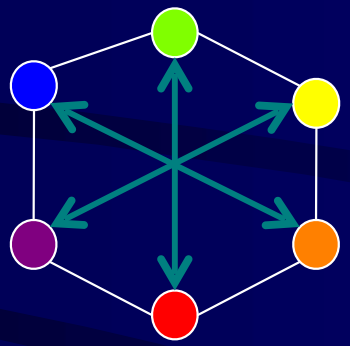
¿Otra?





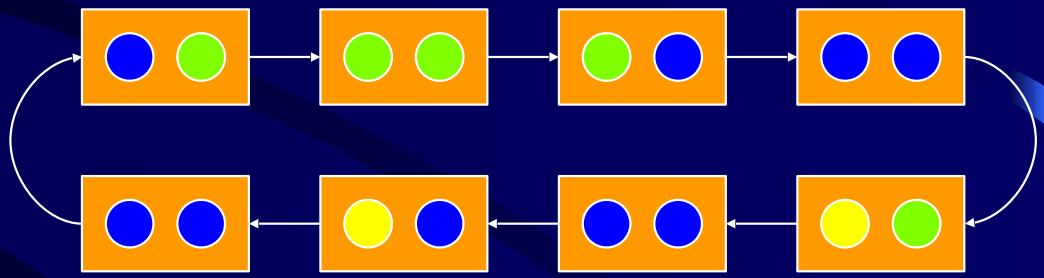
Hermoso ⟷ **Horrible**





Combinaciones Hermosas ⟷

Igual **Diferente**



Introspección y Lectura

- Introspección
  - "Pensar" o "Inventar" requerimientos
- Lectura
  - Usar documentación existente como fuente de requerimientos
  - Reportes, Requerimientos del sistema anterior, manuales de usuario de sistemas externos, manuales operativos del personal
- Ambas técnicas son
  - Muy comunes (particularmente al inicio de IR)
  - Muy malas en revelar necesidades reales
  - Inherentemente sesgadas

Category A response times

Response times, broken down by Primary Care Trust (PCT) area, will be updated here on a monthly basis.

The figures relate to the first vehicle - whether it be a rapid response car, motorcycle or ambulance - to arrive at an incident. The national performance standard is to reach 75% of Category A (immediately life threatening) calls within eight minutes.

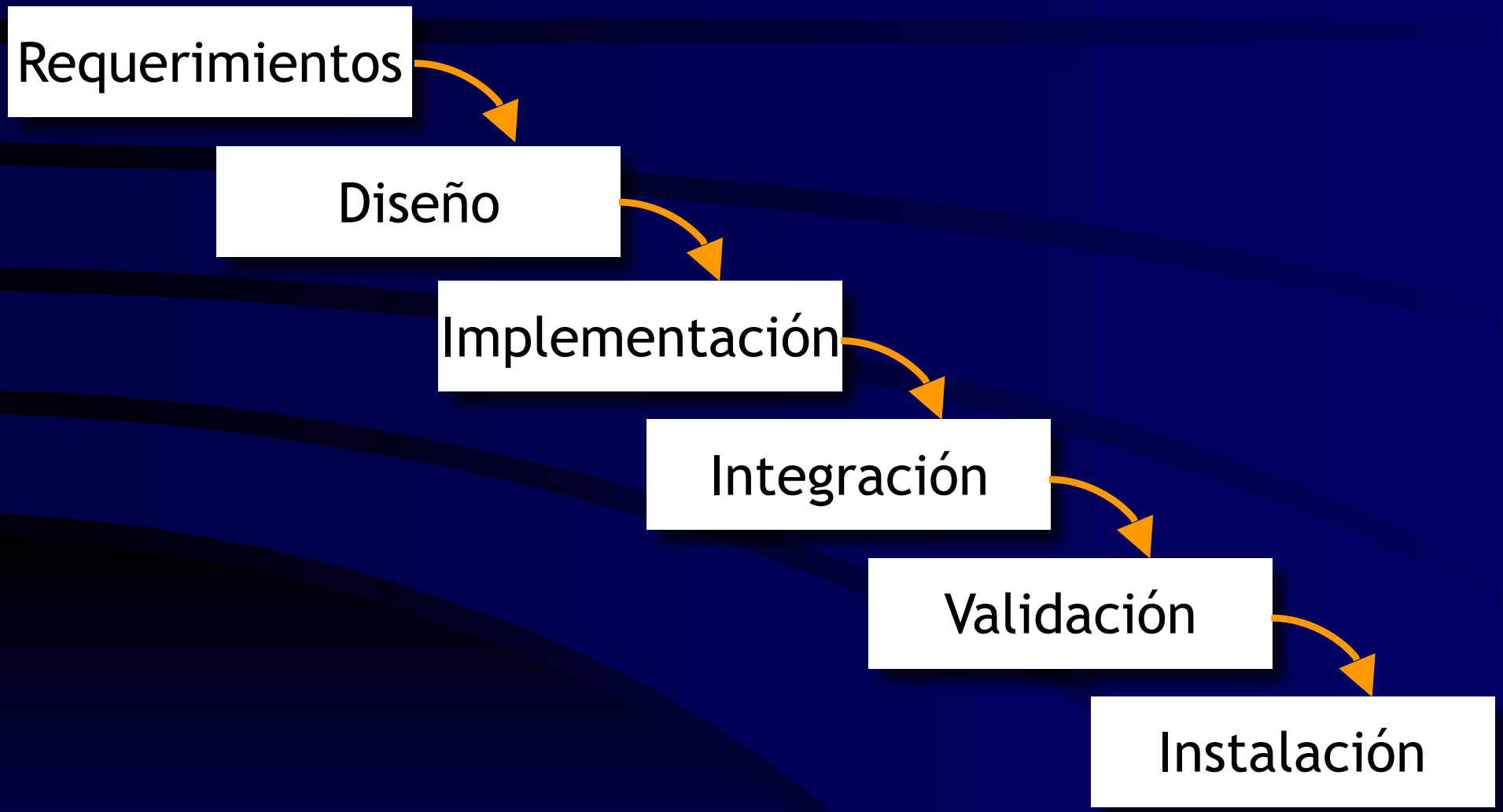
Proportion of Category A calls reached in 8 minutes: National standard = 75%

	Feb-06	Mar-06	Apr-06	May-06	Jun-06	Jul-06
Brent	78%	80%	82%	79%	72%	72%
Ealing	71%	76%	80%	74%	71%	71%
Hammersmith & Fulham	82%	84%	85%	83%	81%	73%
Harrow	68%	76%	75%	73%	72%	72%
Hillingdon	73%	78%	74%	72%	73%	69%
Hounslow	66%	75%	73%	68%	69%	74%
Kensington & Chelsea	69%	69%	68%	71%	68%	69%
Westminster	71%	75%	77%	72%	68%	74%
NW London SHA	**72%**	**77%**	**77%**	**74%**	**72%**	**71%**
Barnet	69%	70%	75%	68%	67%	64%
Camden	75%	81%	78%	74%	68%	74%
Enfield	81%	84%	82%	82%	73%	76%

Cuando hay que conseguirlos?

"Antes de programar. Incrementalmente"

Modelo Cascada (Royce, 1970)



Actividades y Entidades

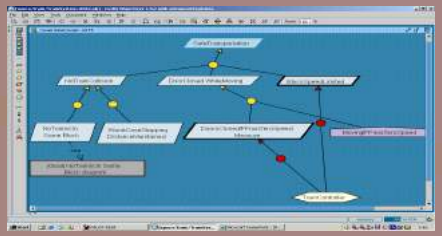
stakeholders

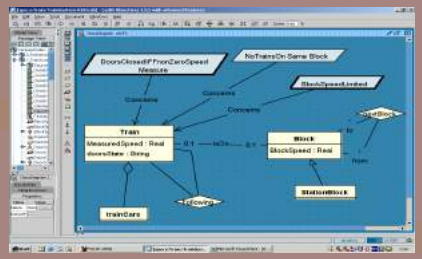
sistemas existentes

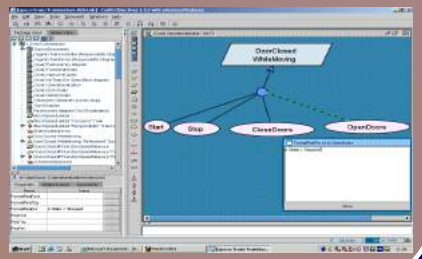
documentos

elicitación y modelado

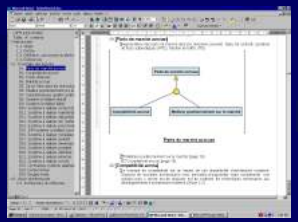
Modelos de Requerimientos







especificación



Especificación de Requerimientos

análisis y validación

negociación y priorización

¿Para qué sirve un SRS?

- Comunicar de manera precisa los requerimientos, objetivos y presunciones del dominio
- Contrato
  - legal, documento interno o a modo de memorando
- Base para estimación (tamaño, costo, tiempo) y planificación de proyecto
- Base para evaluación de producto final
  - verificación y validación
  - Debería tener suficiente información para decidir si el producto final es aceptable (satisface los requerimientos)
- Base para el control de cambios
  - Requerimientos cambian, software evoluciona, el entorno evoluciona

Standard de IEEE para un SRS

Adaptado de IEEE-STD-830

1 Introduction
- Purpose
- Scope
- Definitions, acronyms, abbreviations
- Reference documents
- Overview

2 Overall Description
- Product perspective
- Product functions
- User characteristics
- Constraints
- Assumptions and Dependencies

3 Specific Requirements

Appendices

Index

Identifica el producto y el dominio de la aplicación

Define el contenido y estructura del resto del documento

Describe todas las interfaces externas: sistema, usuario, hardware, software

Resumen de funciones principales

Cualquier cosa que limitará las opciones del desarrollador (ej. regulaciones, limitaciones de hardware, etc)

La parte principal del documento. IEEE STD provee 8 esqueletos diferentes para esta sección

IEEE STD Sección 3 (ejemplo)

3.1 External Interface Requirements

- 3.1.1 User Interfaces
- 3.1.2 Hardware Interfaces
- 3.1.3 Software Interfaces
- 3.1.4 Communication Interfaces

3.2 Functional Requirements

this section organized by mode, user class, feature, etc. For example:

- 3.2.1 User Class 1
  - 3.2.1.1 Functional Requirement 1.1
  - …
- 3.2.2 User Class 2
  - 3.2.1.1 Functional Requirement 1.1
  - …
- …

3.3 Performance Requirements

3.4 Design Constraints

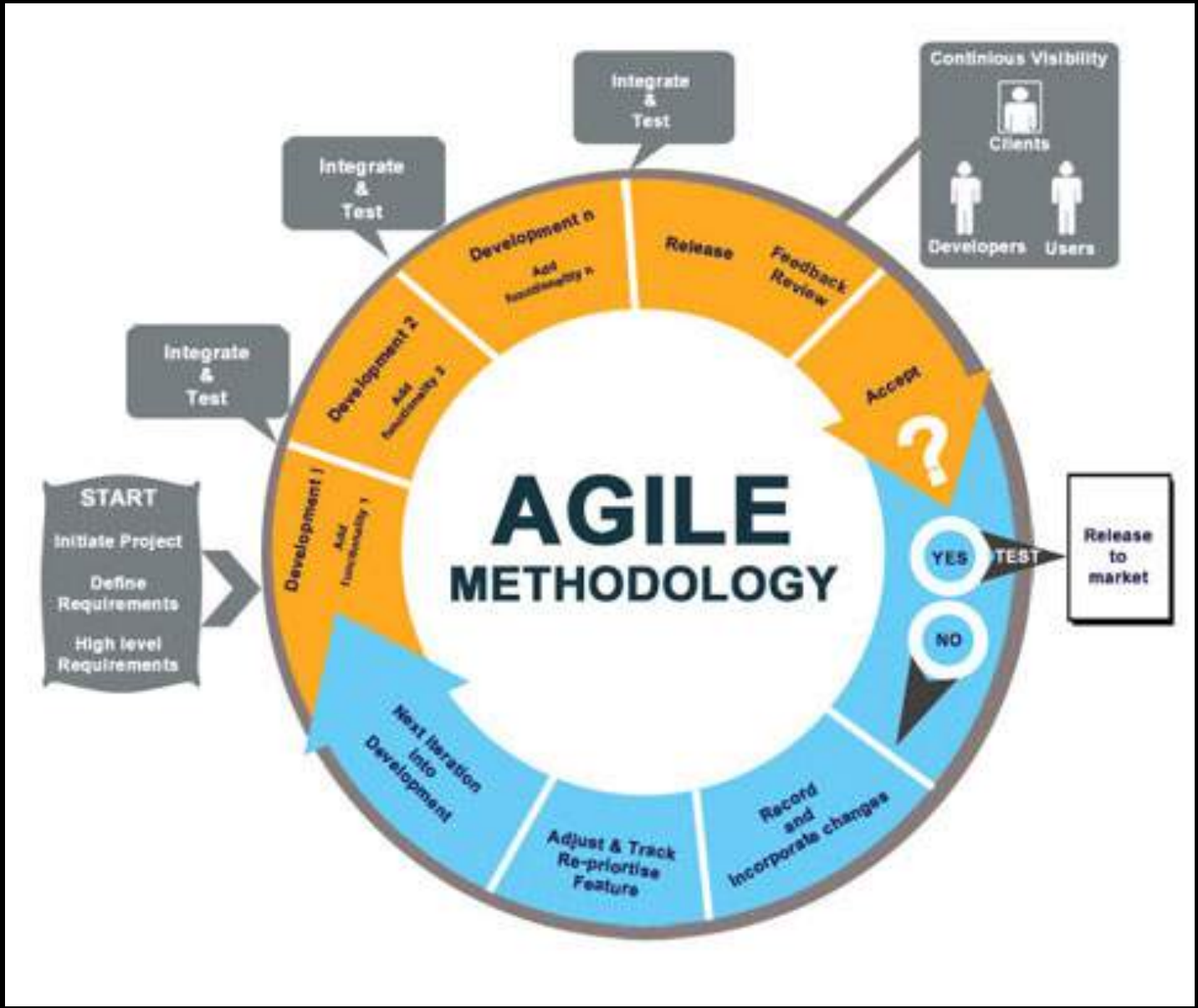
- 3.4.1 Standards compliance
- 3.4.2 Hardware limitations
- etc.

3.5 Software System Attributes

- 3.5.1 Reliability
- 3.5.2 Availability
- 3.5.3 Security
- 3.5.4 Maintainability
- 3.5.5 Portability

3.6 Other Requirements

La “Realidad” sobre Requerimientos a la Cascada


AGILE
METHODOLOGY
START
Initiate Project
Define Requirements
High level Requirements
Development 1
Add functionality 1
Development 2
Add functionality 2
Development n
Add functionality n
Integrate & Test
Integrate & Test
Integrate & Test
Release
Feedback Review
Accept
YES
NO
TEST
Release to market
Record and incorporate changes
Adjust & Track Re-prioritise Feature
Next iteration into Development
Continious Visibility
Clients
Developers
Users

Requerimientos y Agilidad

- Métodos ágiles suelen minimizar la necesidad de entender el problema de fondo porque asumen alguna de la siguientes:
  - La responsabilidad de alinear objetivos del negocio con requerimientos es del otro (ej. el cliente) y fue hecho con anterioridad.
  - El problema y/o la solución son variantes de cosas ya hechas (diseño normal vs diseño radical)
  - Decisiones locales llevan a un optimo global

Actividades y Entidades

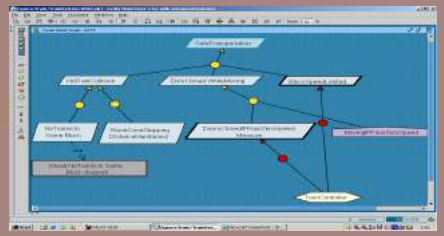
stakeholders

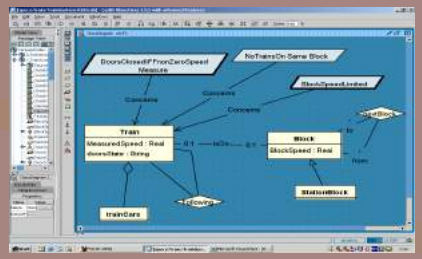
sistemas existentes

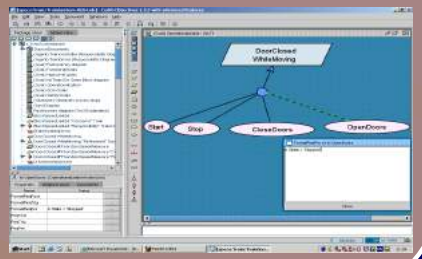
documentos

elicitación y modelado

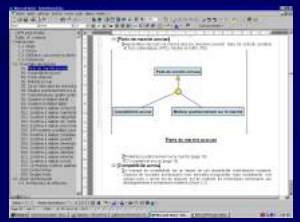
Modelos de Requerimientos







especificación



Especificación de Requerimientos

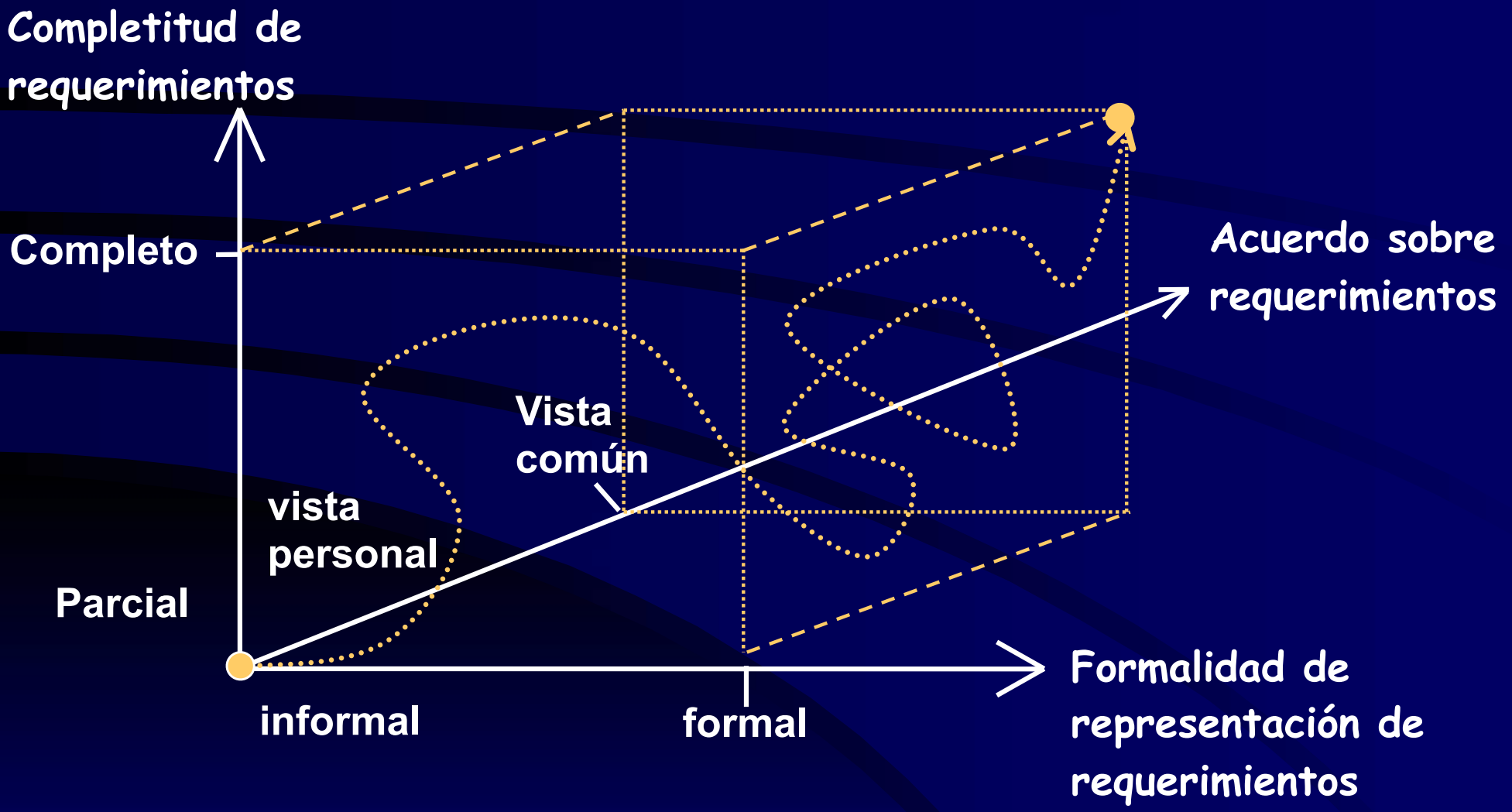
análisis y validación

negociación y priorización

Ciclo de Vida de la IR

(Según Boehm, 1988 - Kontoya/Somerville, 1997)

Caracterización de Progreso



[Pohl 1996]

Puntos de Inicio

¿Cómo empieza un proyecto?
¿Qué hay que saber al principio?
¿Cómo influencia el proceso de IR?

El Puntapié Inicial

- Típicamente todo empieza con una declaración informal y ambigua:
  - E.g. Biblioteca universitaria: El director quiere informatizar el sistema de compras de libros.
  - E.g. Compañía de Seguros: El gerente quiere bajar el tiempo de procesamiento de un reclamo al seguro de 2 meses a 2 semanas
  - E.g. Compañía de Teléfonos: El CIO quiere integrar sistemas de facturación y de administración de clientes de las diferentes filiales...
  - E.g. Agencia espacial: El presidente quiere mandar una misión tripulada a Marte antes del 2020

Un error clásico: Lanzarse a un proceso de exploración de la funcionalidad del sistema a construir...

Explorando el Alcance
- El síntoma y el problema -

- Problema inicial:
  - —"Libros no están disponibles para los alumnos cuando empiezan las clases"
- Por qué?
  - —Porque las ordenes de compra llegan demasiado tarde.
- Por qué?
  - —Porque los profesores son asignados a materias pocos meses antes del comienzo del cuatrimestre
- Por qué?
  - —Porque no está claro quienes están disponibles para dar clase
- Por qué?
  - —Porque hay un recambio constante de personal.
- Por qué?
  - —Mercado competitivo, salarios bajos y sistema de contratación lento.

El Sistema depende del alcance

- Software de Compras para Bibliotecas
  - — Para minimizar el tiempo entre el pedido y la compra.
- Software de Asignación de Profesores a Materias
  - — Para minimizar el tiempo entre que se conocen a los profesores y se los designa a materias.
- Aumento de Salario
  - — Para prevenir recambio
- Software de administración de personal
  - — Para acelerar tramite de incorporación de personal.
- El director de estudios fija el material de lectura obligatorio.
  - — Elimina el síntoma....

Inicio de un Proyecto

- La misión es entender el contexto y alcance para luego
  - fijar una estrategia para la Ingeniería de Requerimientos
  - darle peso relativo a las distintas actividades de IR
  - entender el rol que jugará el SRS
- ¿Qué se necesita saber?
  - tipo de proyecto y contexto organizacional
  - Cuales son los objetivos principales
  - Cuales son los mayores riesgos
  - Cual es el alcance del proyecto
  - Quienes son los stakeholders
  - Vale la pena hacer el proyecto

Tipos de Proyectos

No todos los proyectos son iguales...

- Greenfield vs. Brownfield
  - Software a construir de cero, aprovechamiento de oportunidades tecnológicas o de mercado.
  - Evolución (mejorar, integrar, adaptar, extender) de un sistema sistema de software existente
- Diseño (de sistema) radical vs. normal
  - Combinaciones probadas de tecnologías y procesos probados
  - Inclusión de tecnologías novedosas, modos de interacción entre componentes innovadores, organización de workflows no estándar.

Tipos de Proyectos

- Motivado por un Cliente vs. Mercado
  - Necesidades concretas de un cliente particular en un contexto organizacional especifico al cual se accede en forma directa
  - Necesidades potenciales de un segmento de acuerdo a la percepción de tercero.
- Motivado por problema vs. tecnología
  - Insatisfacción con sistema actual o falta de solución a un problema nuevo
  - Oportunidad creada por tecnología nueva para resolver un problema mejor. E.g. GPS, aplicaciones móviles...
- Producto vs. Familia de productos
  - Ejemplo de familia: Industria automotriz, móviles, …
  - A considerar variantes de sistemas: usuarios, entorno, interfases, requerimientos

El Entorno Organizacional

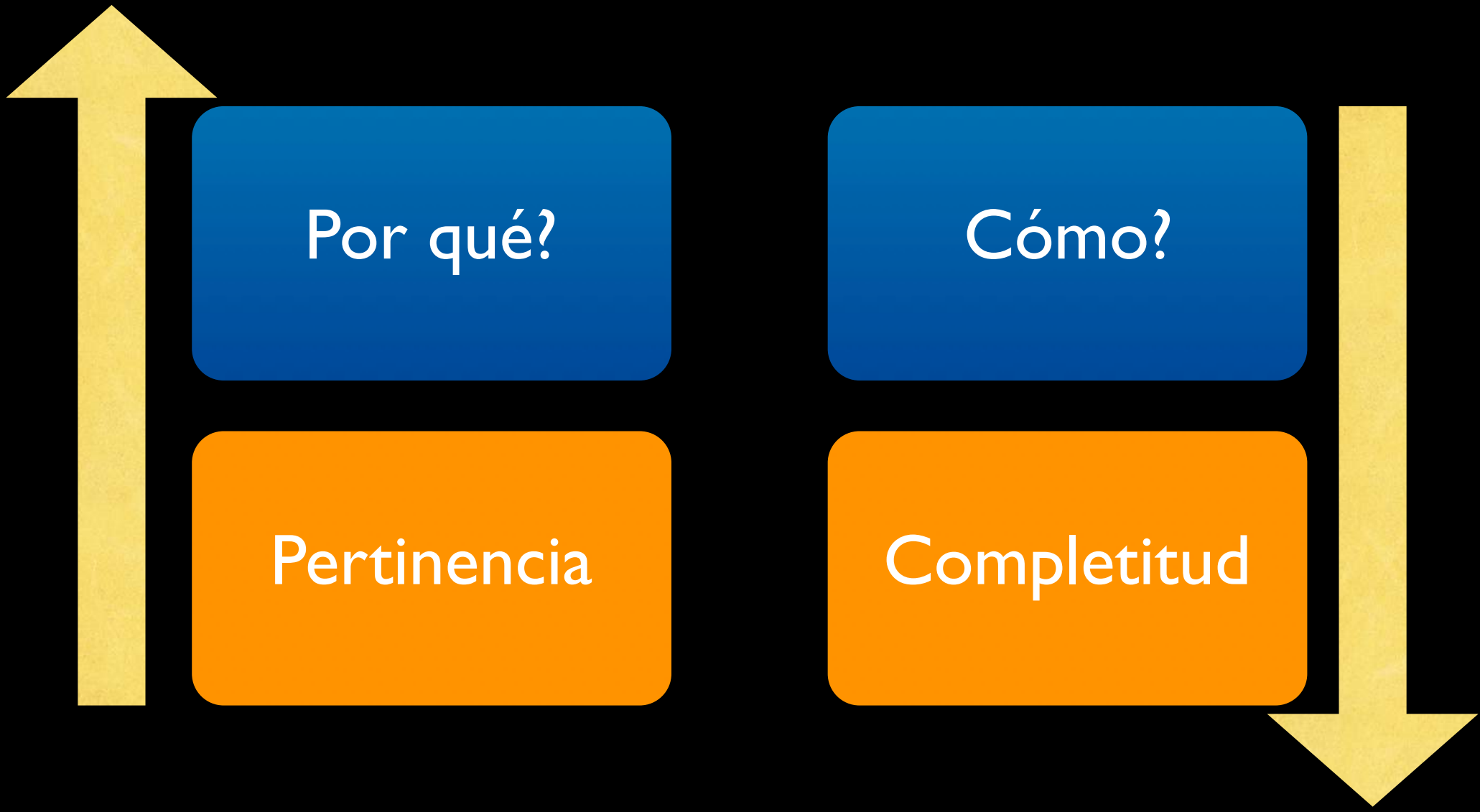
- Desarrollo in-house
  - Cliente y desarrollador pertenecen a la misma organización
- Desarrollo por encargo
  - Una organización le pide a otra un sistema para sus requerimientos específicos
- Customización
  - Un producto genérico o un entorno es configurado para las necesidades de un cliente
- Desarrollo distribuido
  - El conocimiento de un dominio de aplicación y el uso del sistema es utilizado entre varias organizaciones

Impacto del Tipo de Proyecto

El tipo de proyecto da pautas
de cómo encarar un proceso de IR

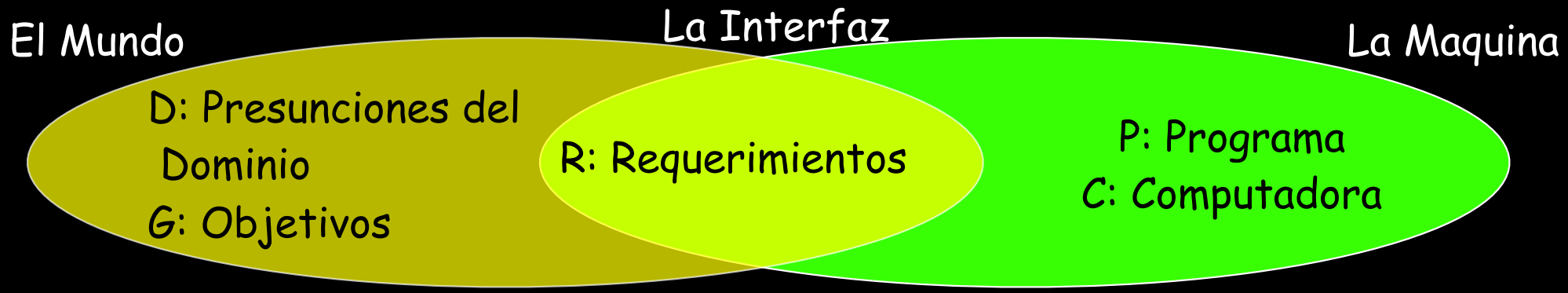
- El peso relativo de las actividades de IR
- El uso de técnicas para el soporte de actividades de IR
- El grado de entrelazamiento requerimientos/diseño
- Los tipos de stakeholders a considerar
- Los tipos de desarrolladores involucrados
- Los riesgos del proyecto
- Necesidad y uso del SRS

Dos Herramientas de Análisis Fundamentales



Definir el alcance

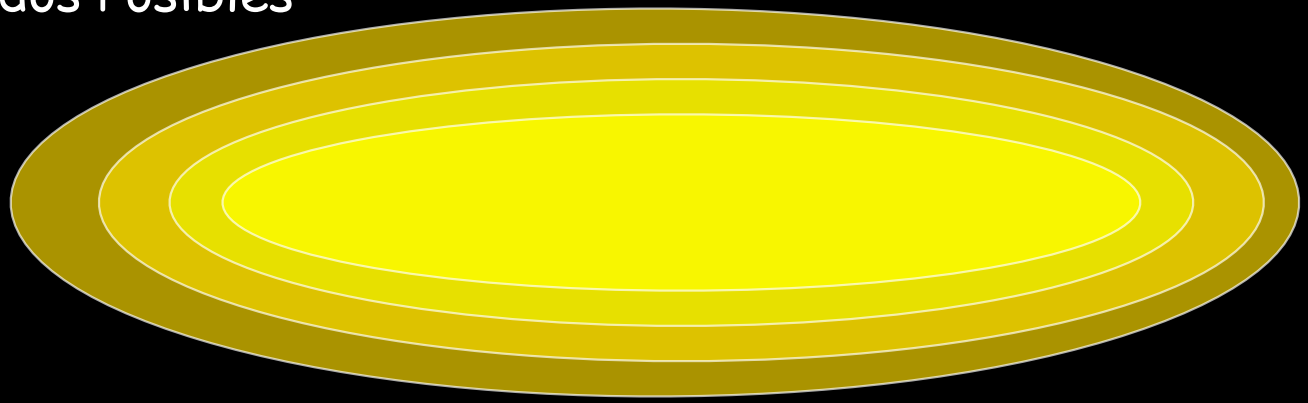
- ¿Qué partes del mundo nos interesan?
- ¿Qué partes del mundo NO nos interesan?
- Que partes del mundo tenemos la autoridad y/o posibilidad de cambiar?



Riesgos en la Definición de Alcance

- Un alcance demasiado limitado:
  - riesgo de perder oportunidades, o de construir el sistema equivocado
- Un alcance demasiado amplio:
  - riesgo de sobrepasar nuestra autoridad y competencia, esfuerzo perdido

Los Mundos Posibles



Heurísticas para definir el alcance

- Puede el problema específico ser resuelto independientemente del problema general?
- Hasta que punto el problema general resuelve el síntoma?
- El alcance elegido es aceptable para el cliente?
- Alguien estaría dispuesto a pagar por una solución a este problema?
- Existe una solución razonable en costo a este problema?

El alcance debe ser explorado.

Análisis de Factibilidad

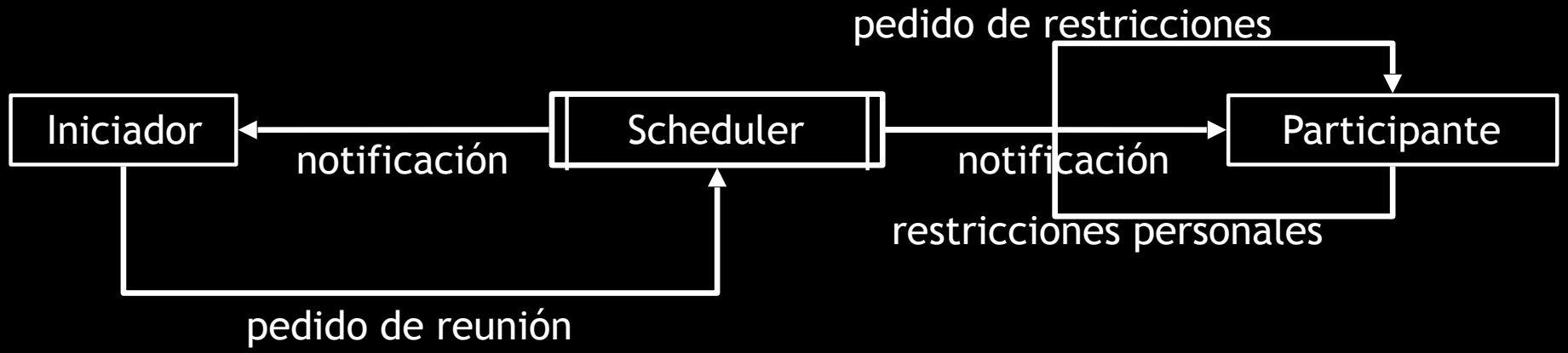
Varios dimensiones a analizar...

- Costo
  - ¿Cuánto está dispuesto a pagar el cliente?
    (a diferencia del costo del proyecto)
- Tiempo
  - ¿Para cuándo se necesita el sistema?
    (a diferencia del tiempo de desarrollo esperado)
- ¿Es técnicamente posible?
- ¿Es legal? ¿Ético? ¿Políticamente viable?

¿Cómo se responde esta pregunta si no se conocen aún los requerimientos?

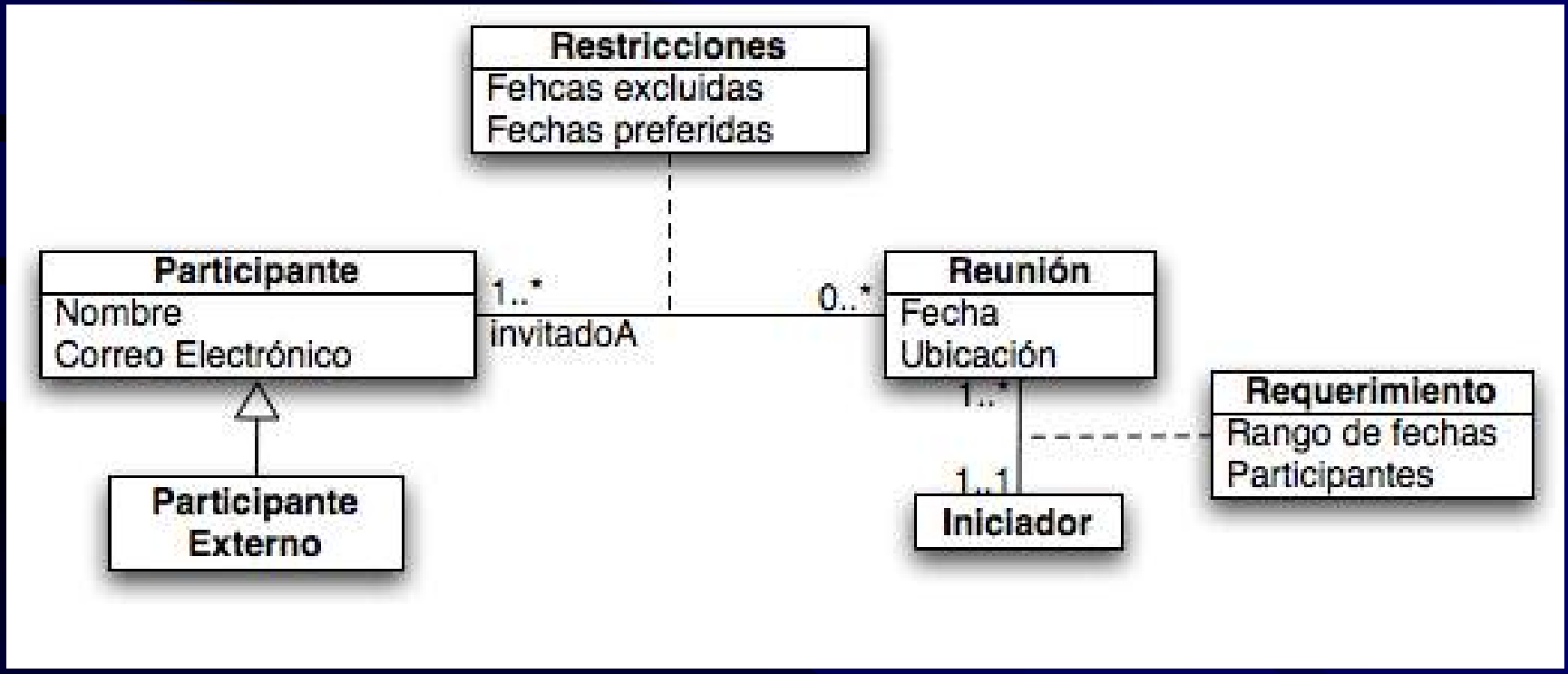
Documentación

Diagrama de Contexto



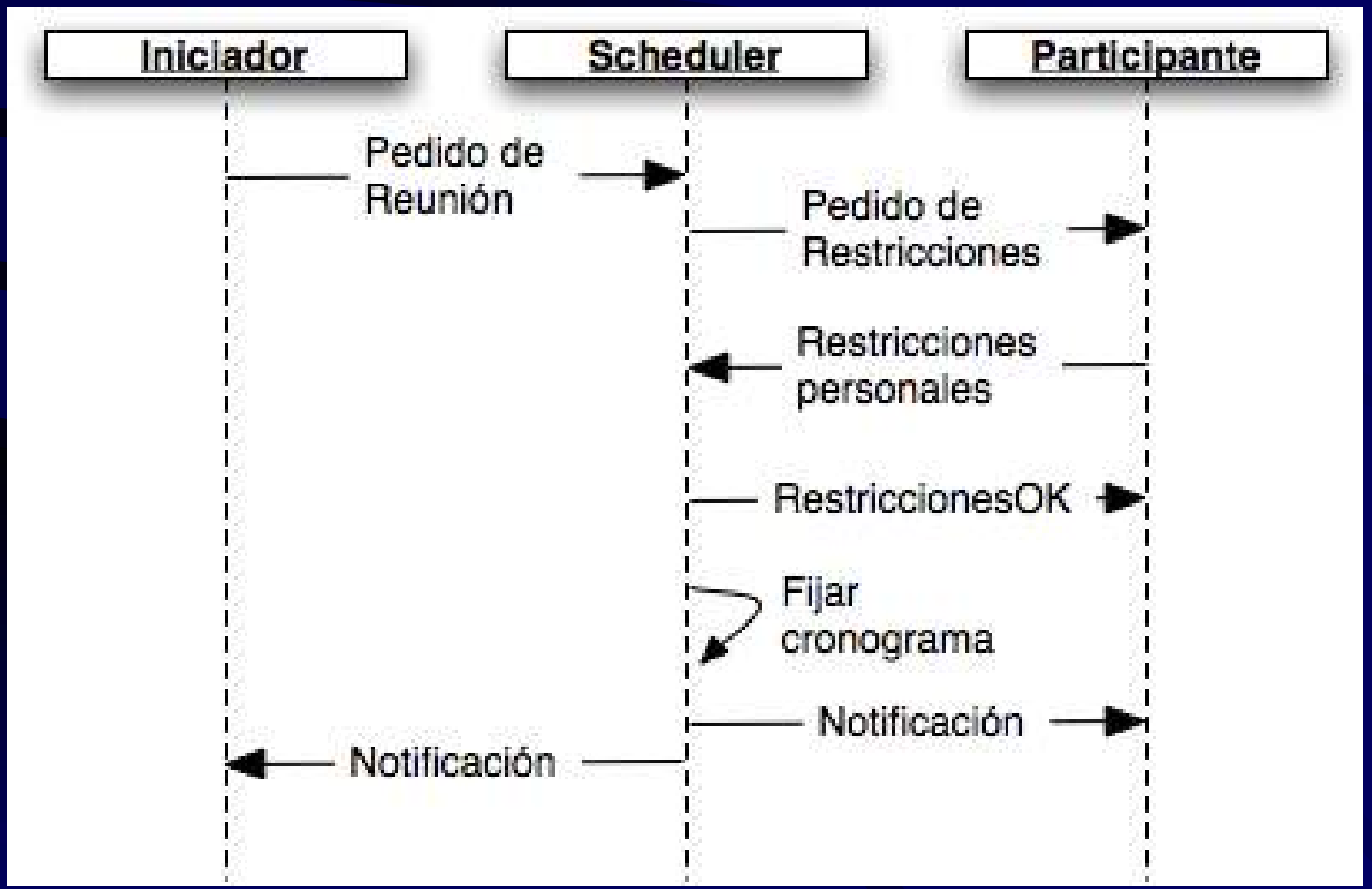
Diagramas Estructurales del Dominio

- Ej. Diagramas de clase, modelos entidad relación



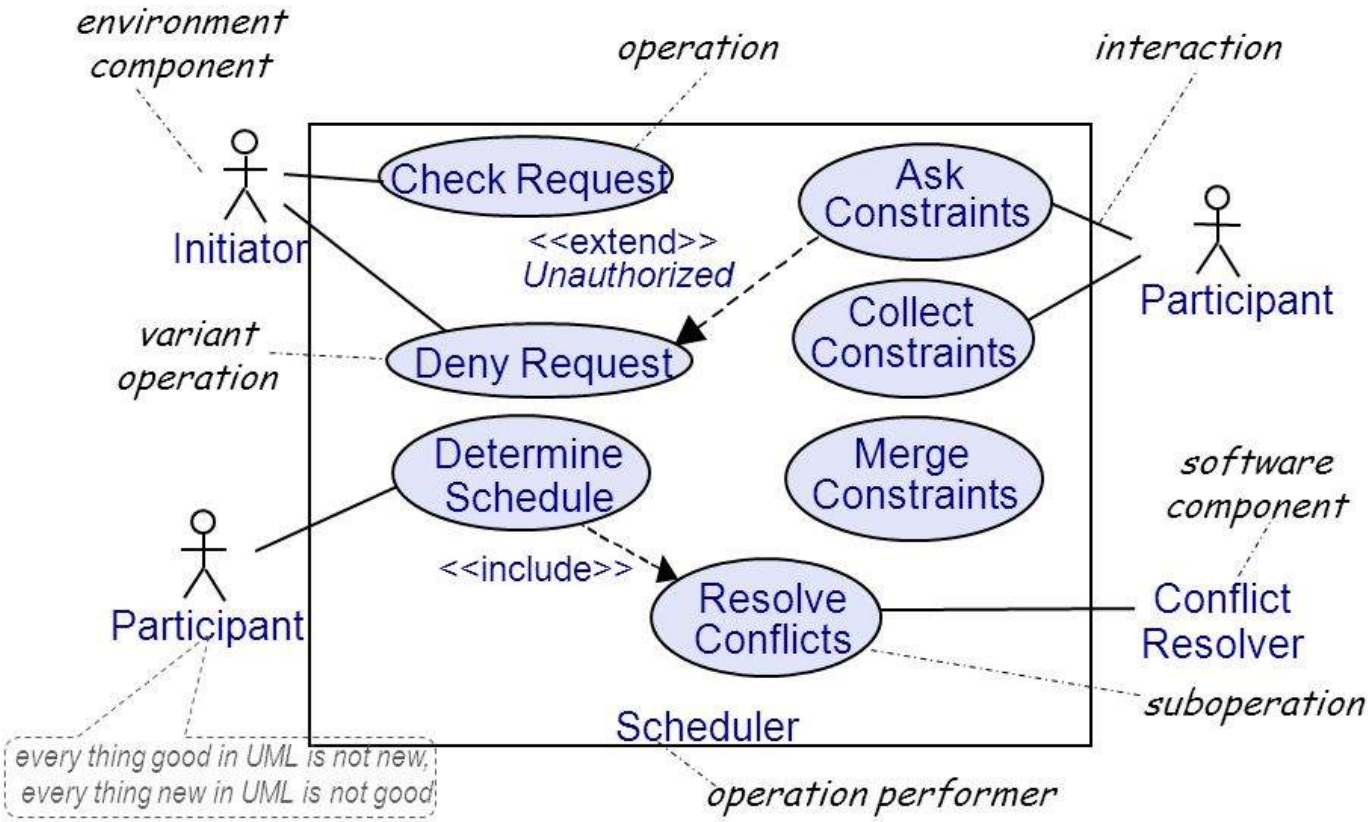
Diagramas de Secuencia

- Conceptos:
  - Tiempo, comunicación o interacción entre agentes
  - Descripción basada en ejemplos.



Maquinas de Estados

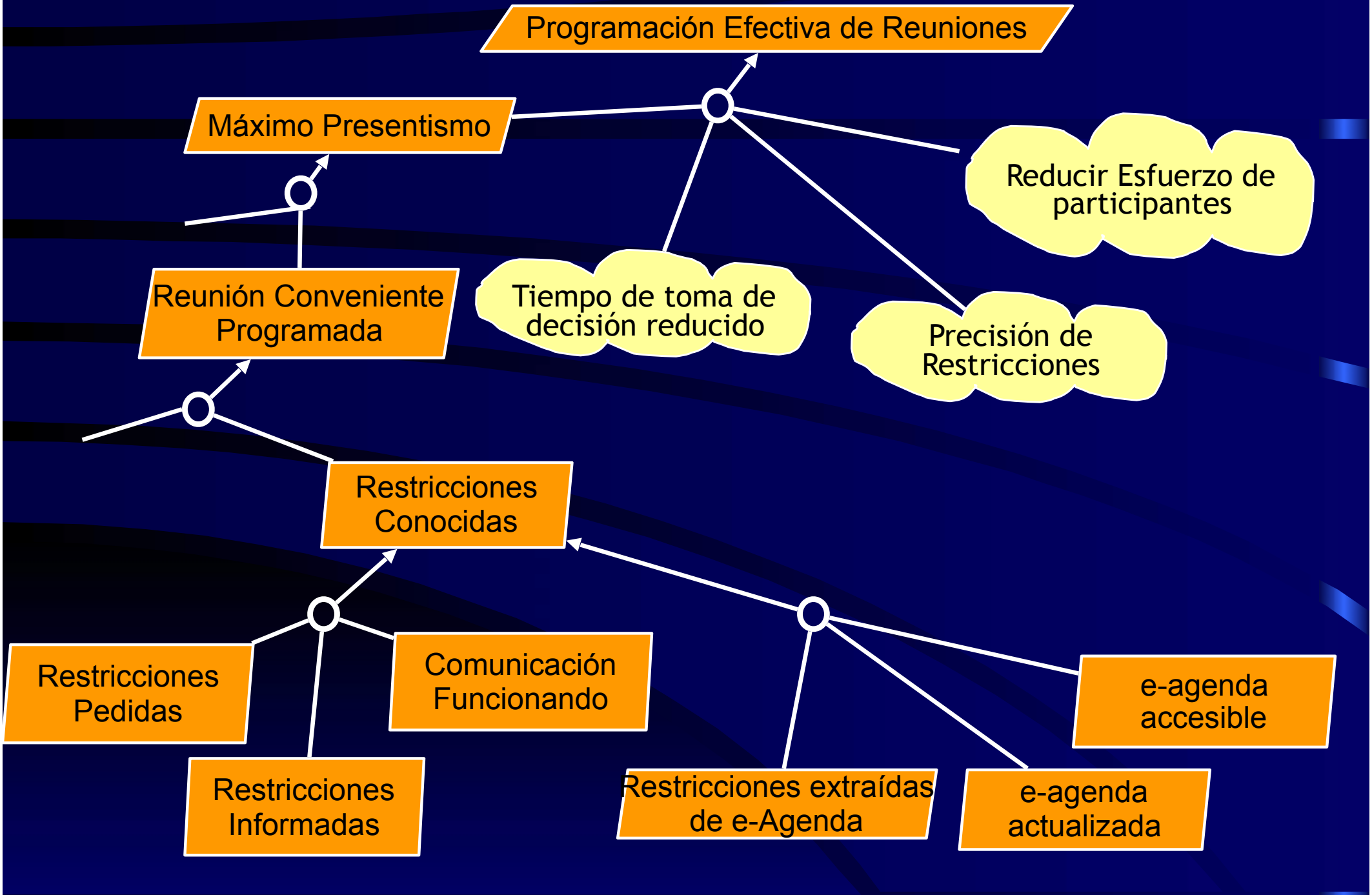
Use case diagram: example



Especificación de Operaciones

Operación: Programar Reunion
Responsable: Software
Def: Define fecha y hora de reunion
Entrada: set(Restricciones)
Salida: fecha f
Pre: size(Restricciones) > 0
Post: Consistente (f, Restricciones)
Trig: Por pedido del inciador

Modelo de Objetivos



Cerrando…

La Ingeniería de Requerimientos

Requirements Engineering (RE) is a set of activities concerned with identifying and communicating purpose of a software-intensive system, and the contexts in which it will be used. Hence, RE acts as the bridge between the real world needs of users, customers, and other constituencies affected by a software system, and the capabilities and opportunities afforded by software-intensive technologies

No es una fase o etapa!

Comunicación es tan importante como la recolección y análisis

Calidad significa que cumple con su propósito. No se puede decir nada acerca de calidad si no se entiende el propósito.

Diseñadores necesitan saber cómo y donde el sistema será utilizado.

Requerimientos tratan en parte de lo que se necesita...

...y en parte de lo que es posible

Necesidad de indentificar todas las partes involucradas - no sólo el usario y cliente

Cosas para llevarse…

- Hacer requerimientos parece fácil y banal
- Problemas en requerimientos son la causa numero 1 de problemas en proyectos de software
- Entender el problema (los objetivos) es clave para razonar sobre completitud y pertinencia de requerimientos
- Entender las presunciones es clave para razonar sobre riesgo.
- Hacer IR es diseñar sistemas

Bibliografía

- Software Requirements & Specifications: A Lexicon of Practice, Principles and Prejudices, Michael Jackson. Addison-Wesley and ACM Press, 1996.
- Requirements Engineering: From System Goals to UML Models to Software Specifications, Axel van Lamsweerde, Wiley, 2009
- *No Silver Bullet: Essence and Accidents of Software Engineering, Fred Brooks, 1987. In Mythical Man Month, Addison-Wesley, 1995.*
- *T. E. Bell, T. A. Thayer. Software Requirements: Are They Really a Problem? ICSE 1976*
- *B. W. Boehm, "Software Engineering Economics", Prentice Hall, 1981.*
- *B. W. Boehm, "Verifying and Validating Software Requirements and Design Specifications," IEEE Software, 1984*