Curso de Diseño de Software – Impartido por Mario Chacón

Semestre V

Contenido

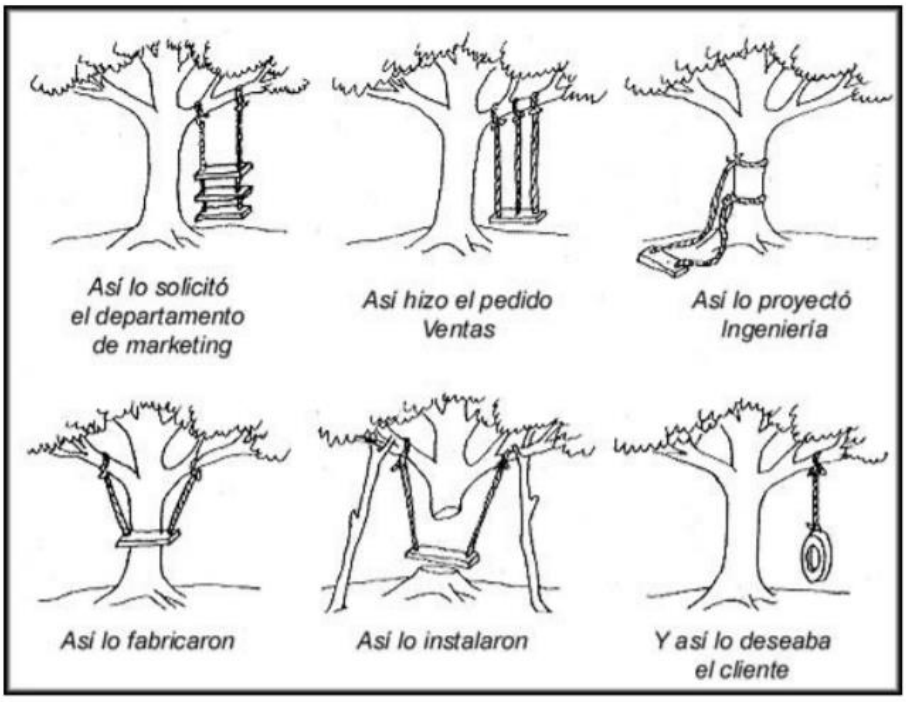
1. Cuestionamientos al Desarrollo de Software

1.1 Cuestionamientos al desarrollo de Software

- Los sistemas no responden a las expectativas de los usuarios.
- Los programas fallan con cierta frecuencia.

- Los costos del software son difíciles de prever y normalmente superan las estimaciones.
- La modificación del software es una tarea difícil y costosa.
- El software se suele presentar fuera del plazo establecido y con menos características de las consideradas inicialmente.
- Normalmente, es difícil cambiar el entorno hardware usando el mismo software.
- El aprovechamiento óptimo de los recursos (personas, tiempo, dinero, herramientas, entre otros.) no suele cumplirse.

1.2 Eterno problema del diseño



1.3 Contexto del profesional en Computación

El contexto abarca las siguientes áreas:

- Ingeniería en Computadores
- Ciencias de la Computación
- Tecnologías de Información
- Ingeniería de Software

- Sistemas de Información

1.4 Problemas tradicionales del Desarrollo de Software

- Estimación de Tiempos
- Gestión de Riesgos
- Control de Calidad
- Diseño Inadecuado
- Poca experiencia en tecnologías
- Poca motivación
- Asignar personal nuevo a proyecto retrasado
- Cambio vertiginoso de TI
- Comunicación de equipos
- Procesos - metodologías – modelos

1.4.1 Estimación de Tiempos

- Estimaciones de tiempo poco fiables
- Basado en empirismo
- Con poca documentación
- Es poco probable que al final del proyecto se reflexione y compare la estimación con lo real
    - Escasa documentación de bitácoras y progreso.
    - Escasas estadísticas de las estimaciones y progreso.
- Conocimiento de técnicas
    - Puntos de Fusión
    - Casos de uso
    - Delphi
    - Juicio Experto

1.4.2 Gestión de Riesgos

Riesgo: evento o condición incierta que en caso de que suceda incide en uno o varios de los objetivos del proyecto.

Tipos de Riesgos:

- Líder del proyecto
- Ejecución del proyecto
- Usuarios
- Entorno organizacional y político

1.4.2.1 Riesgos de Líder de Proyecto

- Experiencia profesional
- Experiencia en el negocio del proyecto
- Competencias de estimación
  - Tiempo
  - Recursos
  - Presupuestos
- Liderazgo
- Identificación de funcionalidades
- Identificación con el proyecto y la organización

1.4.2.2 Riesgos de Ejecución de Proyecto

- Selección y contratación de personal (inadecuado)
- Idoneidad de Metodología de Desarrollo
- Definición de roles y responsabilidades
- Planeación efectiva del proyecto
- Control del proyecto
- Cultura organizacional
- Cumplimiento de objetivos

1.4.2.3 Riesgos del Usuario

- Compromiso de la gerencia
- Identificación de los usuarios claves para el proyecto
- Participación activa y motivada de usuarios
- Generalmente no están al alcance de los líderes del proyecto

1.4.2.4 Riesgos del entorno organizacional y Político

- Cambios en la gerencia organizacional
- Políticas nacionales o internacionales
- Dependerán del margen de acción de la organización
- A nivel nacional:
    - Presupuestos y ejecución presupuestaria
    - Presupuestos anuales →proyectos con presupuesto plurianual
- A nivel internacional:
    - Cultura
    - Industria
    - Conformación de equipos interculturales

1.4.3 Control de Calidad

- Definición de calidad en la organización
- Posibilidad de inversión
- Medición de costos reales y monitoreo de proyecto
- Control vs Aseguramiento de Calidad
- Apoyos metodológicos
    - ISO
    - CMMI

1.4.4 Diseño Inadecuado

- Dimensión de etapa de Diseño
- Modelos de Diseños
- Innovación vs aplicaciones tradicionales
- Diseño de Interfaz ←→ Diseño de Arquitectura ←→ Diseño de Pruebas

1.4.5 Experiencia en Tecnologías

- Experiencia del equipo es directamente proporcional en incidencia al éxito del proyecto
- Aprendizaje vs Productividad
- Investigación se paga en tiempo
- Experiencia en Tecnología puede presentar un impacto directo según la Experiencia que se tenga sobre el negocio

1.4.6 Motivación

- La motivación del equipo del proyecto incide más que elementos técnicos
- Identificar claramente los roles requeridos del proyecto y las competencias del equipo
- Impacto de los elementos personales

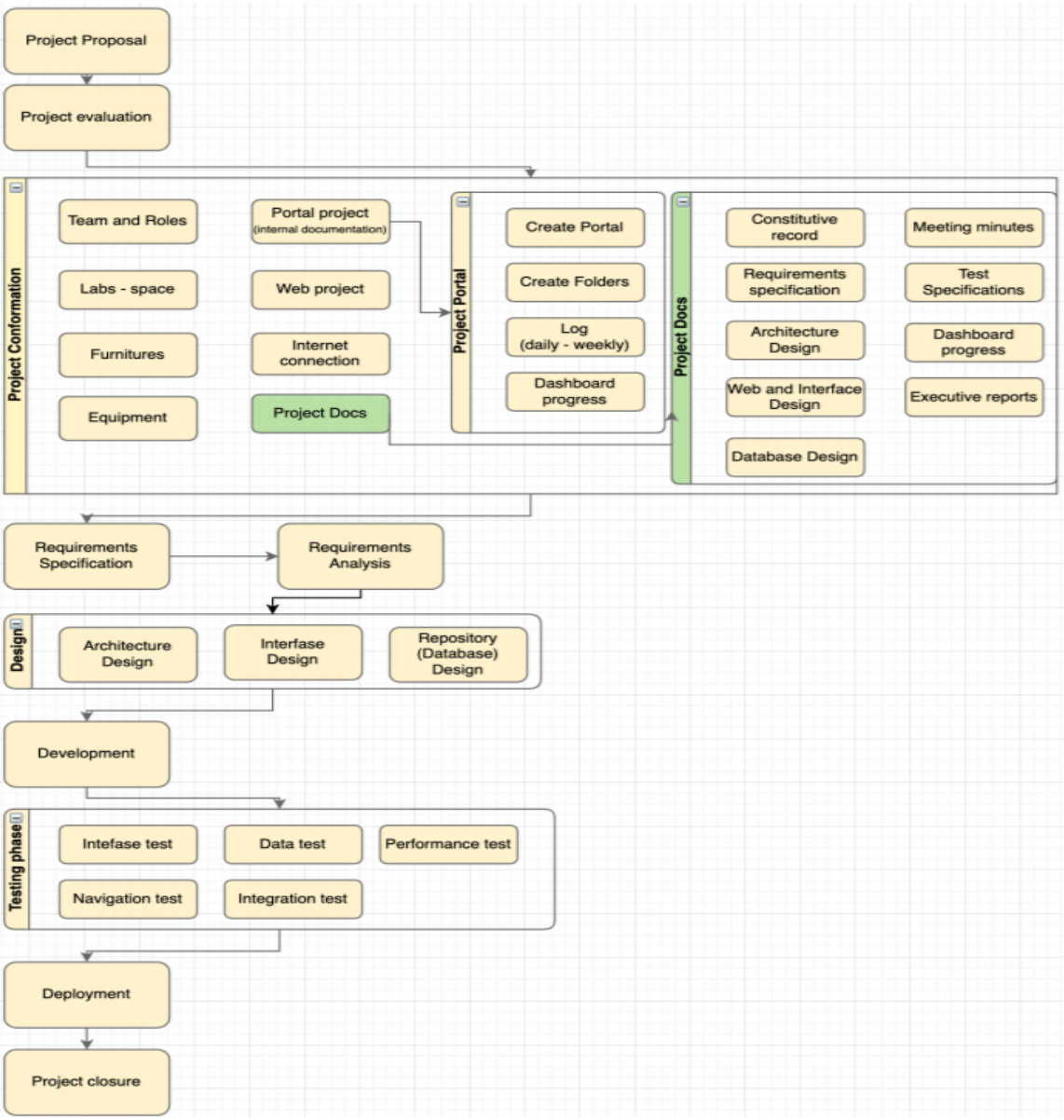
1.4.7 Asignación de personal nuevo

Este punto va a analizarse por medio del libro → "Mítico Mes-Hombre", específicamente en el segundo capítulo el cual tiene el mismo nombre.

El análisis de la lectura se localiza en el "Resumen del Mítico Mes-Hombre"

1.4.8 Fases del Desarrollo de Software

- Elicitación de Requerimientos
- Análisis
- Diseño
- Desarrollo
- Pruebas
- Implementación
- Entrega

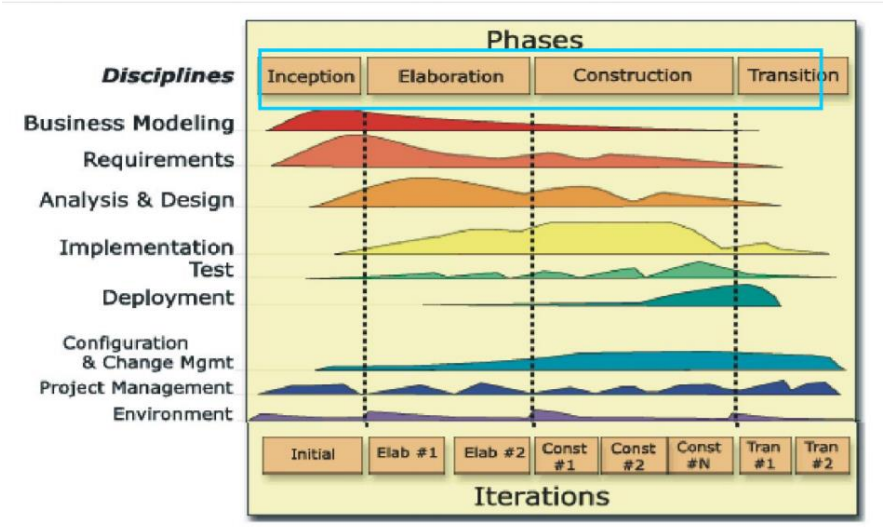


2. Procesos de Ingeniería de Requerimientos

2.1 ¿Qué es la Ingeniería de requerimientos?

- Actividades para descubrir, documentar y mantener un conjunto de requerimientos.
- Técnicas sistemáticas y repetibles.
- Documenta las necesidades de negocio y necesidades técnicas.
- Forma parte del proceso de desarrollo del Software (o de otras disciplinas).

2.2 Un proceso de desarrollo (referencia: RUP)



2.3 Disciplina de Requerimientos

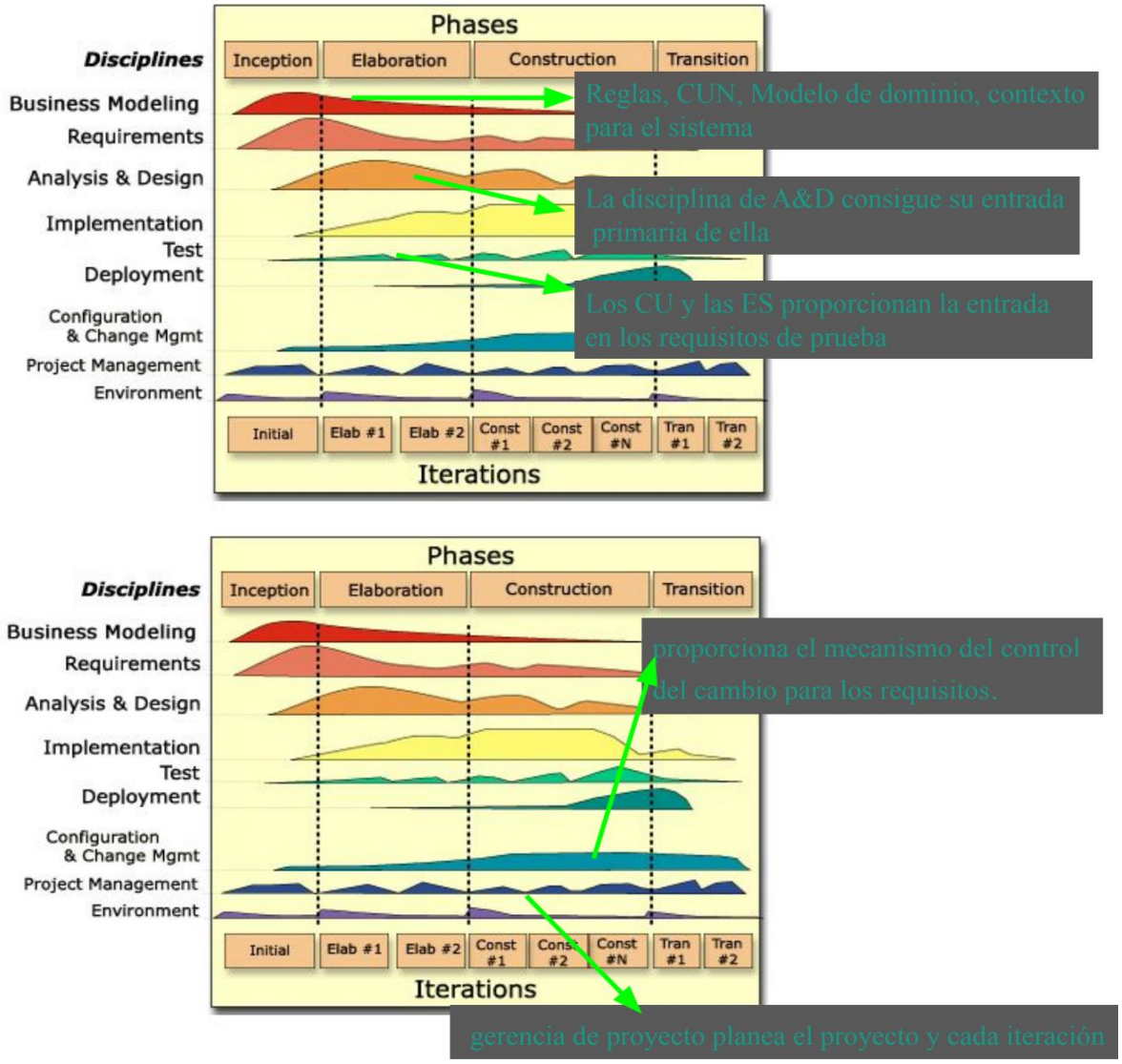
Principales motivos de la importancia de dicha disciplina:

- Para establecer y mantener el acuerdo con los clientes y otros tomadores de decisión de lo que debe hacer el sistema.

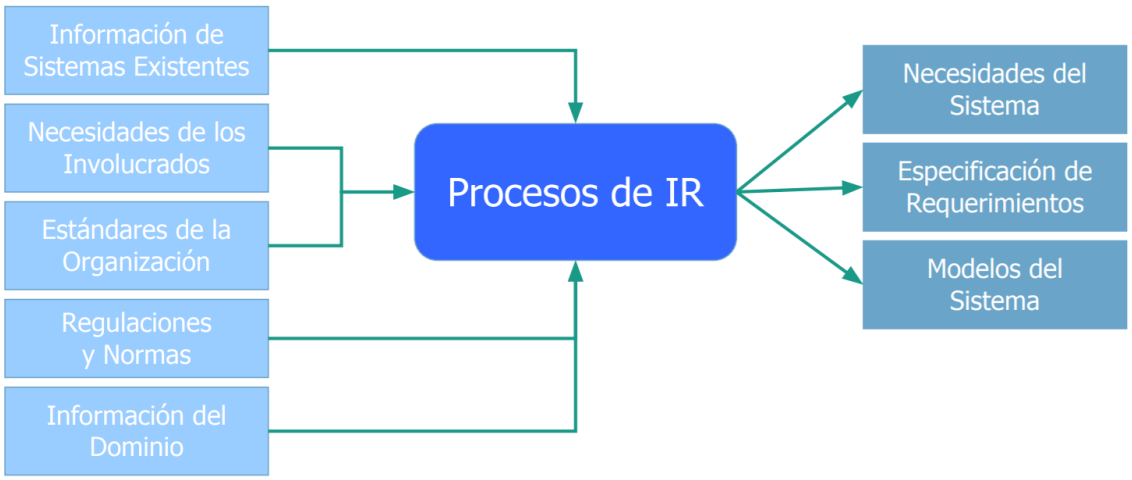
  Para que los desarrolladores del sistema obtengan una mejor comprensión de los requerimientos del sistema.
- Para definir los límites del sistema.
- Para proporcionar una base para planear el contenido técnico de las iteraciones.

- Para definir una interfaz usuario-sistema, centrándose en las necesidades y las metas del usuario.
- Para alcanzar estas metas, es importante que todos los involucrados entiendan la definición y el alcance del problema que estamos intentando solucionar con este sistema.
- Entradas necesarias:
    - Reglas de negocio
    - Modelos de caso de uso de negocio
    - Modelo del análisis de negocio

2.4 Relación con otras disciplinas

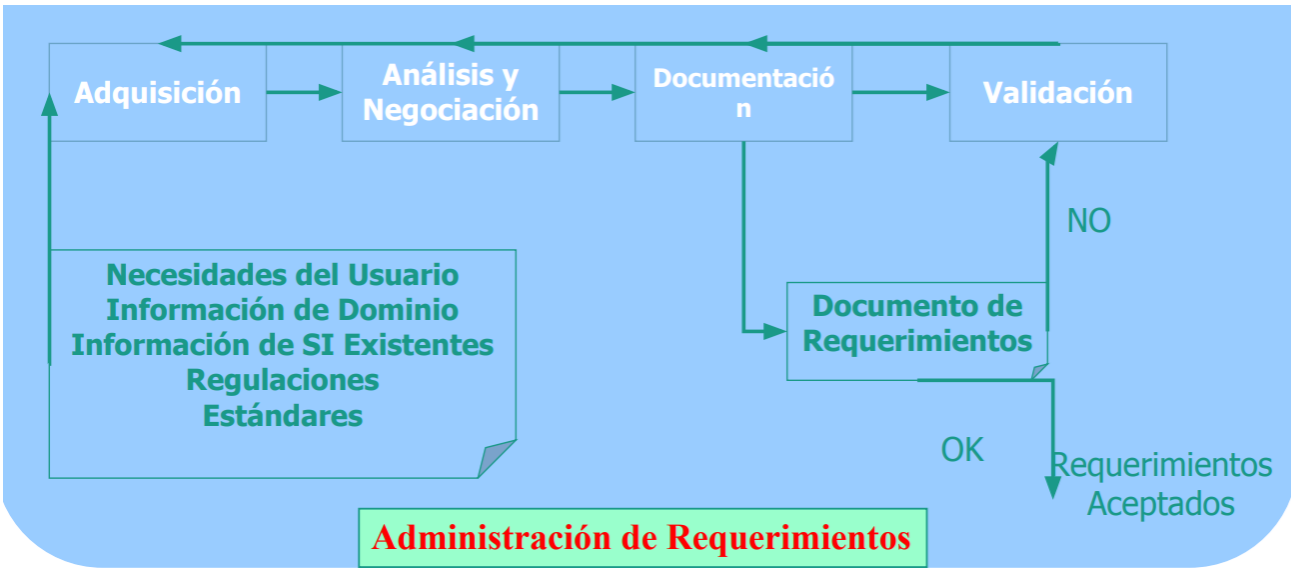


2.5 Entradas y Salidas



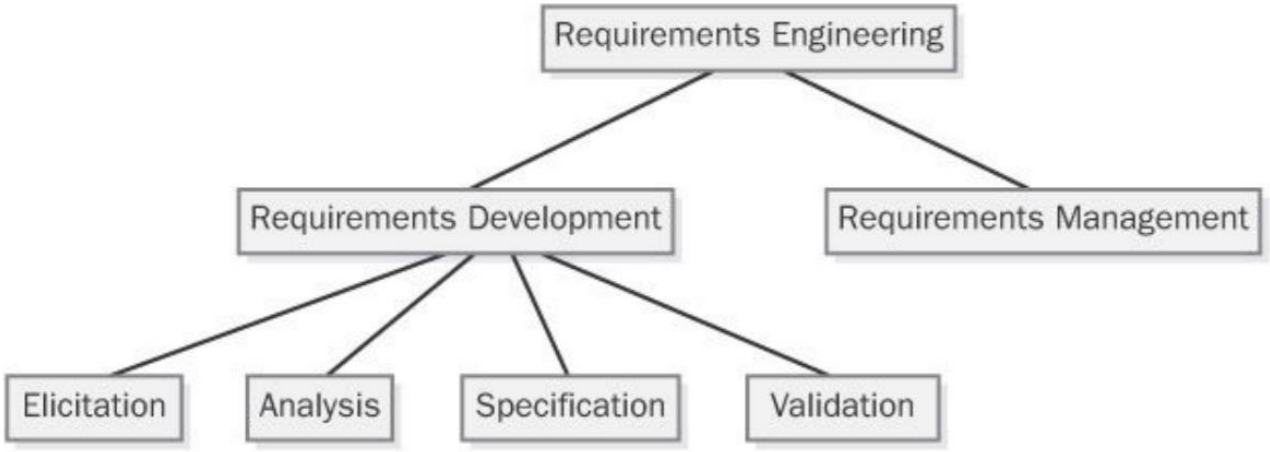
2.6 Procesos de la Ingeniería de Requerimientos

- Conjunto estructurado de actividades las cuales derivan, validan y mantienen un documento de requerimientos de sistema.
- Actividades.
- Roles y responsables.
- Calendarización de Actividades.
- Asignación de Responsables a las Actividades.
- Entregables (artefactos)

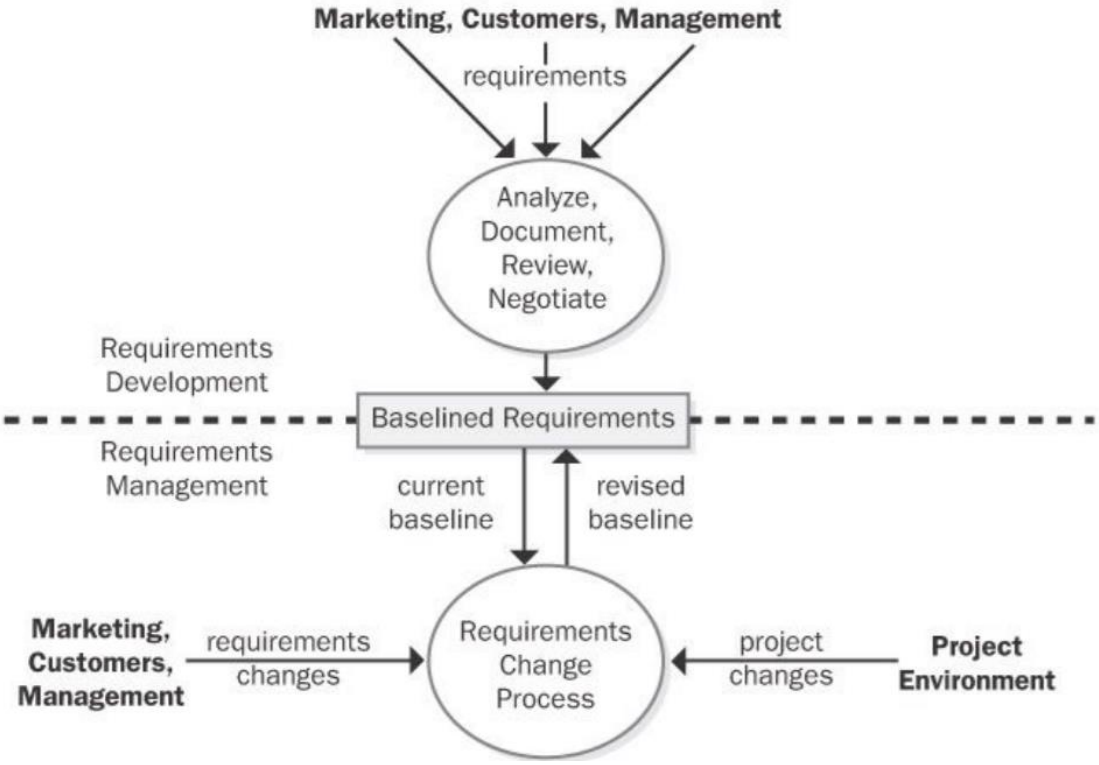


Enfoque sistemático para encontrar, documentar, organizar y dar seguimiento a los requerimientos cambiantes de un sistema.

2.7 Desarrollo y administración



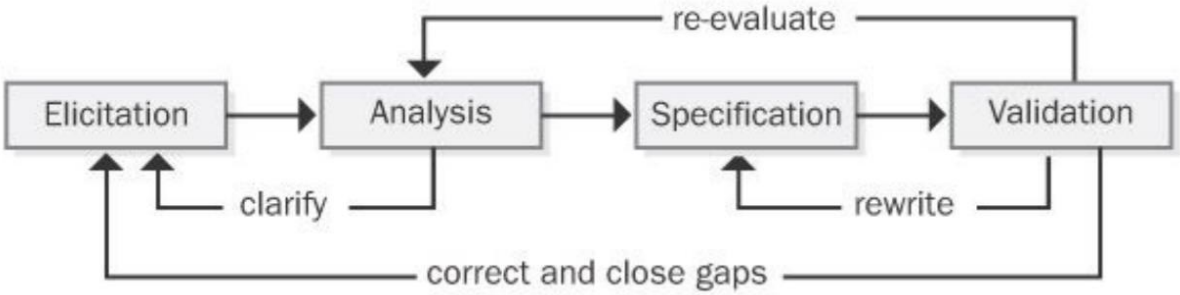
2.8 Relación entre desarrollo y administración

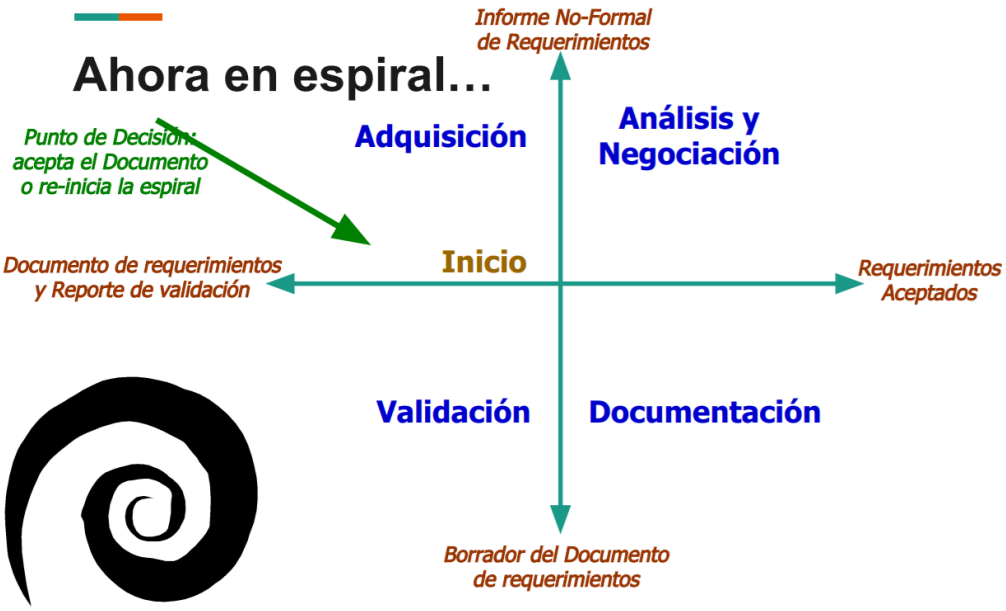


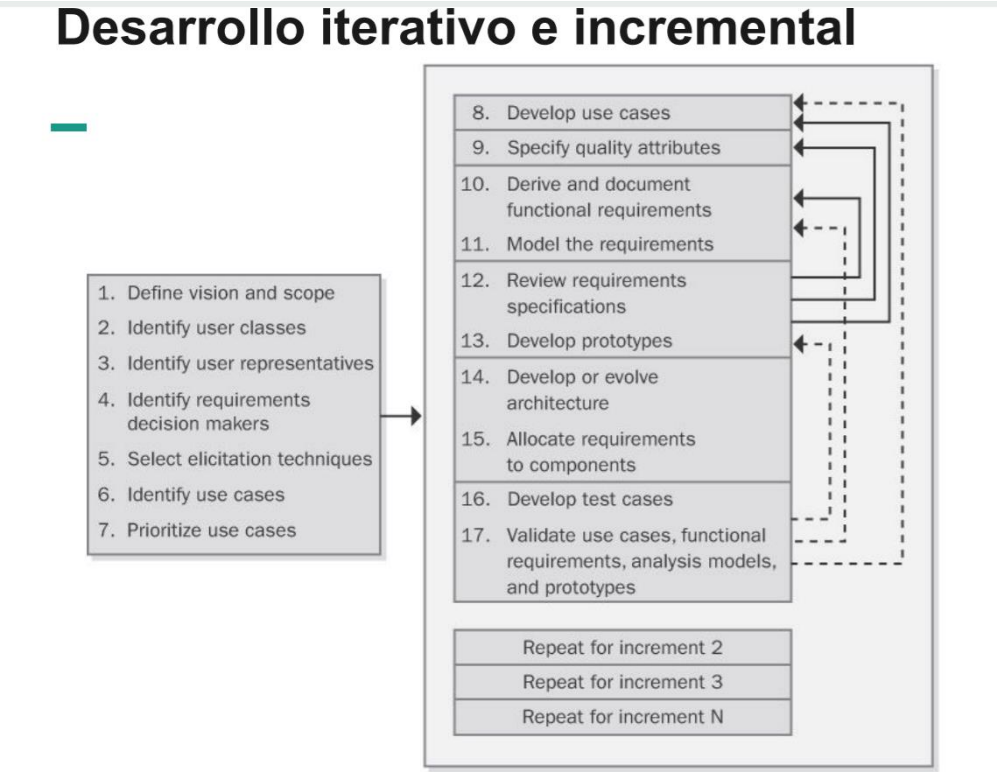
2.9 Desarrollo de requerimientos

- Educción (adquisición), análisis y negociación, documentación (especificación) y validación comprenden:

- Identificar las clases de usuario esperadas para el producto.
- Indagar las necesidades de los individuos que representan a cada clase de usuario.
- Comprender las tareas y metas de los usuarios, así como los objetivos del negocio con los que estas tareas se alinean.
- Analizar la información obtenida de usuarios para distinguir metas de sus tareas de los requerimientos funcionales, requerimientos no funcionales, reglas del negocio, soluciones sugeridas e información no atinente.
- Asignar partes de los requerimientos de alto nivel a los componentes del software que se hayan definido en la arquitectura del sistema.
- Comprender la importancia relativa de los atributos de calidad.
- Negociar prioridades de implementación.
- Traducir las necesidades del usuario recolectadas en las especificaciones de requerimientos y modelos.
- Revisar los requerimientos documentados para asegurar una comprensión común de los requerimientos enunciados por los usuarios y corregir cualesquiera problemas antes de que el grupo de desarrollo los acepte.



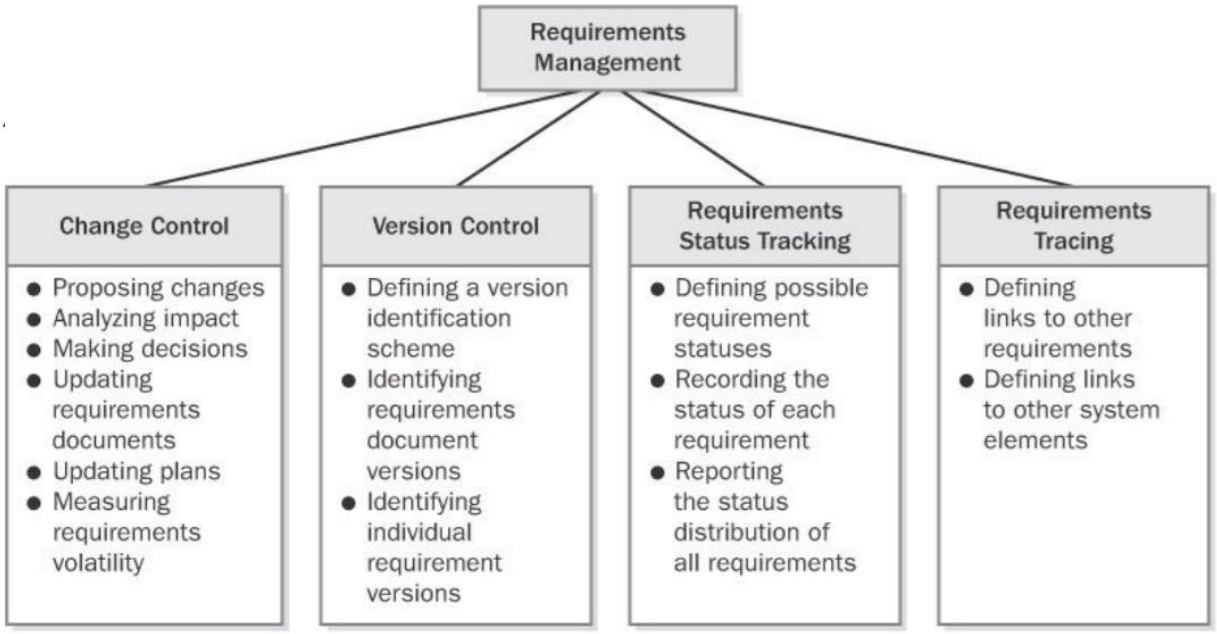


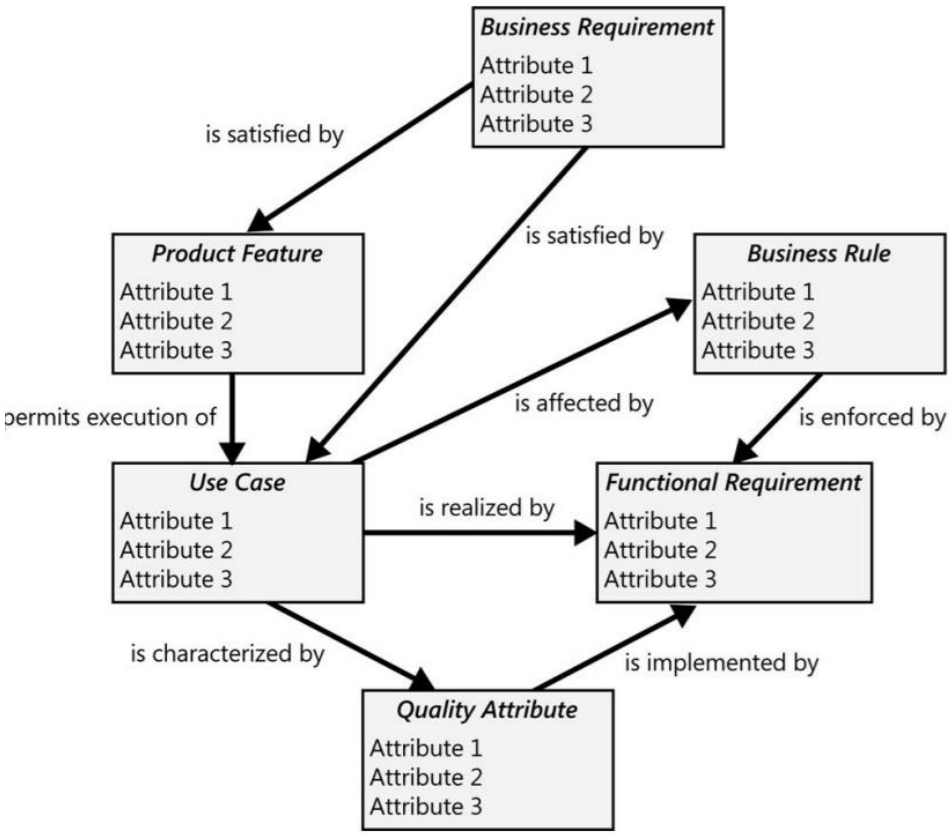


2.10 Administración de requerimientos

- Es el proceso para manejar los cambios a los requerimientos de sistemas, para mantener un acuerdo con el cliente respecto de los requerimientos del proyecto de software.

- Definir una línea base para los requerimientos.
- Revisar los cambios propuestos a los requerimientos y evaluar su posible impacto antes de aprobar cada cambio.
- Documentar cambios aprobados de una manera controlada.
- Mantener los planes del proyecto actualizados con los requerimientos.
- Negociar nuevos compromisos con base en el impacto estimado de los cambios en los requerimientos.
- Rastrear (“trazar”) los requerimientos a las partes correspondientes de casos de uso, diagramas, diseño, código y casos de prueba.
- Dar seguimiento al estado de los requerimientos y su actividad de cambio a lo largo del proyecto.
- Verificar que se implementen correctamente los requerimientos.
- Construir un Comité de Control de Cambios.





2.11 10 trampas a evitar en el proceso

1. Confusión entre requerimientos.
2. Involucramiento inadecuado del cliente.
3. Requerimientos vagos o ambiguos.
4. Requerimientos sin priorizar.
5. Especificar funcionalidades que el usuario no utilizará.
6. Síndrome: análisis-parálisis.
7. Incrementos descontrolados en el alcance.
8. Proceso inadecuado de control de cambios.
9. Análisis del impacto insuficiente.
10. Control de versiones inadecuado.

2.12 Beneficios de un buen proceso de IR[1]

- Menos defectos de requerimientos.
- Re-trabajo de desarrollo reducido.
- Menos características innecesarias.
- Menores costos de mejoras.
- Desarrollo más rápido.
- Menos malentendidos.
- Alcance “creciente sigiloso” reducido.
- Caos de proyecto reducido.
- Estimaciones de pruebas del sistema más exactas.
- Mayor satisfacción de clientes y miembros del equipo de desarrollo.

2.13 Verdades cósmicas sobre los requerimientos

- Cosmic Truth #1: If you don’t get the requirements right, it doesn’t matter how well you execute the rest of the project.
- Cosmic Truth #2: Requirements development is a discovery and invention process, not just a collection process.
- Cosmic Truth #3: Change happens.
- Cosmic Truth #4: The interests of all project stakeholders intersect in the requirements process.
- Cosmic Truth #5: Customer involvement is the most critical contributor to software quality.
- Cosmic Truth #6: The customer is not always right, but the customer always has a point.
- Cosmic Truth #7: The first question an analyst should ask about a proposed new requirement is, “Is this requirement in scope?”
- Cosmic Truth #8: Even the best requirements document cannot – and should not – replace human dialogue.

[1] IR: Ingeniería de Requerimientos

- Cosmic Truth #9: The requirements might be vague, but the product will be specific.
- Cosmic Truth #10: You are never going to have perfect requirements.

2.14 Buenas prácticas de IR

Existen distintas áreas en las cuales se requiere realizar buenas prácticas de Ingeniería de Requerimientos y son las siguientes:

- Conocimiento
- Educción (adquisición)
- Análisis
- Especificación (documentación)
- Validación
- Administración de requerimientos
- Administración de proyectos

2.14.1 Conocimiento

- Capacitar analistas.
- Educar a los representantes de los usuarios y a los administradores acerca de los requerimientos.
- Capacitar a los desarrolladores acerca del dominio de aplicación.
- Crear un glosario.

2.14.2 Educción (adquisición, indagación)

- Definir proceso de desarrollo de requerimientos.
- Definir visión y alcance.
- Identificar clases de usuarios.
- Seleccionar “campeones” del producto (“paladines”). →Usuarios clave
- Establecer grupos de enfoque.
- Identificar casos de uso.
- Identificar eventos y respuestas del sistema.
- Mantener talleres facilitados de educción.

- Observar usuarios realizar sus trabajos.
- Examinar informes de problemas.
- Reutilizar requerimientos.

2.14.3 Análisis

- Dibujar un diagrama de contexto.
- Crear prototipos.
- Analizar factibilidad.
- Dar prioridades a los requerimientos.
- Modelar los requerimientos.
- Crear un diccionario de datos.
- Asignar requerimientos a sub-sistemas.
- Aplicar QFD (Quality Function Deployment = Desplazamiento de la Función de Calidad).

2.14.4 Especificación (documentación)

- Adoptar una plantilla para la ERS.
- Identificar fuentes de los requerimientos.
- Etiquetar cada requerimiento de manera única.
- Registrar reglas del negocio.
- Especificar atributos de calidad.

2.14.5 Validación

- Inspeccionar los documentos de requerimientos.
- Probar los requerimientos.
- Definir criterios de aceptación.

2.14.6 Administración de requerimientos

- Definir un proceso de control de cambios.
- Establecer un Comité de Control de Cambios.
- Realizar análisis del impacto de los cambios.

- Hacer líneas base y controlar las versiones de los requerimientos.
- Mantener historia de los cambios.
- Dar seguimiento al estado de los requerimientos.
- Medir la volatilidad de los requerimientos.
- Utilizar una herramienta de administración de requerimientos.
- Crear una matriz de rastreabilidad de requerimientos.

2.14.7 Administración de proyectos

- Seleccionar un ciclo de vida apropiado.
- Basar los planes en los requerimientos.
- Renegociar compromisos.
- Administrar riesgos de los requerimientos.
- Dar seguimientos al esfuerzo en los requerimientos.
- Revisar lecciones aprendidas del pasado.

2.15 Implementación de las prácticas

Impact	Difficulty		
	High	Medium	Low
High	- Define requirements development process. - Base plans on requirements. - Renegotiate commitments.	- Identify use cases. - Specify quality attributes. - Prioritize requirements. - Adopt SRS template. - Define change-control process. - Establish CCB. - Inspect requirements documents. - Allocate requirements to subsystems. - Record business rules.	- Train developers in application domain. - Define vision and scope. - Identify user classes. - Draw context diagram. - Identify sources of requirements. - Baseline and control versions of requirements.
	- Educate user reps and managers about requirements.	- Train requirements analysts. - Select product champions.	- Analyze feasibility. - Create a glossary. - Create a data dictionary.

Medium	- Model the requirements. - Manage requirements risks. - Use a requirements management tool. - Create requirements traceability matrix. - Hold facilitated elicitation workshops.	- Establish focus groups. - Create prototypes. - Define acceptance criteria. - Perform change impact analysis. - Select appropriate life cycle.	- Observe users performing their jobs. - Identify system events and responses. - Unique label each requirement. - Test the requirements. - Track requirements status. - Review past lessons learned.
Low	- Reuse requirements. - Apply Quality Function Deployment. - Maintain change history volatility.	- Measure requirements. - Track requirements effort.	- Examine problem reports.