# Ingeniería de Requerimientos

# ¿Qué es un requerimiento?

Dos diferentes enfoques:

- Declaración abstracta y de alto nivel de lo que un sistema debe proveer, en un determinado contexto (requerimientos de usuario).

- Definición detallada y formal de una función (requerimientos del sistema).

# Requerimientos: usuario vs sistema

Se desea saber desde Bizagi todas las bajas de los equipos (producto: enlaces de fibra, internet dedicado y transmisión de datos) y su estado (activo, pendiente de desconexión por mora, pendiente de desconexión voluntaria, pendiente de traslado, etc) dada una fecha N.

1. Dada una fecha N, filtrar los productos por su estado y quedarse con los pendiente de desconexión por mora y los pendiente de desconexión voluntaria.
2. Por cada producto, agregar los tipos de productos, cliente y domicilio de instalación.
3. Filtrar los productos por su tipo y quedarse con los de los tipos que coincidan con: enlace de fibra, internet dedicado y transmisión de datos.
4. Devolver el resultado a Bizagi:
    a. Devolver los productos agrupados por domicilio y cliente (I.E: un grupo es el conjunto de productos que comparten domicilio de instalación y cliente).
    b. Por cada grupo, crear un nodo para cada producto donde va estar la información del producto que devuelva ODCIM.

# Requerimientos funcionales y no funcionales

**Req. Funcionales**: servicios que el sistema debe implementar, como debería reaccionar a ciertas entradas y en determinadas condiciones de contexto, limitantes, etc.

**Req. No Funcionales:** restricciones en los servicios y funciones del sistema. Aplica a todo el sistema y no solamente a determinadas funcionalidades.

Por lo general no existe una independencia entre ambos tipos de requerimientos, ya que un req. no funcional puede generar otros requerimientos funcionales y viceversa.

# Requerimientos Funcionales

Describen lo que el sistema debería hacer.

Cuando se expresan como requerimientos de usuario,  se escriben en lenguaje natural para los usuarios del sistema y personal involucrado en la gestión.

Sin embargo, este tipo de requerimientos expanden los requerimientos de usuario cuando se escriben para desarrolladores:

- Funciones del sistema, entradas y salidas, excepciones, etc con gran detalle.
- Completos y coherentes.
- Vulnerabilidades: Sistemas complejos.

# Requerimientos No Funcionales

- No están directamente relacionados con servicios y funcionalidades particulares.
- Implican propiedades como confiabilidad, tiempo de respuesta, uso de memoria, rendimiento, etc.
- Pueden ser más críticos que los requerimientos funcionales porque pueden implicar que el sistema no pueda ser utilizado.

Ejemplos: requerimientos de producto, organización y externos como grandes categorías. Dentro de ellos: regulatorios, ambientales, usabilidad, recursos (memoria, cpu, etc), auditables, etc.

# Requerimientos No Funcionales (continuacion)

| Propiedad | Ejemplos de unidad de medida |
|---|---|
| Velocidad | Procesamiento: Transacciones / segundo<br>Tiempo de respuesta usuario / evento |
| Tamaño | MB / Cantidad de chips (ROM)<br>Espacio en disco |
| Facilidad de Uso | Tiempo de entrenamiento |
| Confiabilidad | Tiempo medio entre fallas<br>Disponibilidad |
| Robustez | Tiempo de reinicio después de un error<br>Posibilidad de corrupción de datos después de un error |
| Portabilidad | Número de sistemas que soportan la solución |

# Procesos de la Ingeniería de Requerimientos

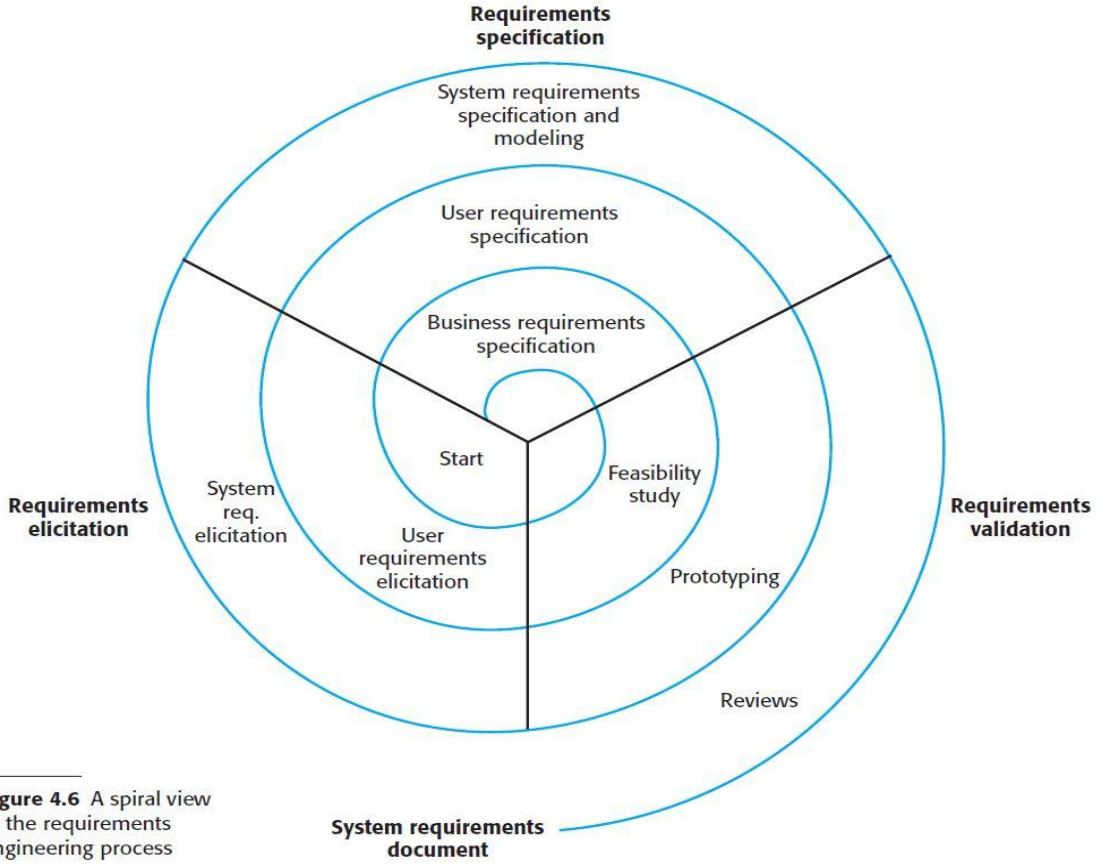


**Figure 4.6** A spiral view of the requirements engineering process

# Validacion y analisis

**Obtención**: el proceso de recopilación de requisitos de varias partes interesadas, incluidos clientes, usuarios finales, analistas comerciales y desarrolladores. Esto implica la realización de entrevistas, talleres y encuestas para recopilar información sobre las necesidades del sistema.

**Análisis**: El proceso de analizar los requisitos recopilados para garantizar que sean completos, consistentes y factibles. Los requisitos deben ser claros, inequívocos y verificables, sin contradicciones ni ambigüedades.

# Especificación y Validación

**Especificación**: El proceso de documentar los requisitos de una manera clara, concisa y sin ambigüedades. Esto implica definir los requisitos funcionales y no funcionales del sistema, incluidos los requisitos de rendimiento, seguridad y usabilidad.

**Validación**: El proceso de garantizar que los requisitos documentados sean precisos, completos y satisfagan las necesidades de las partes interesadas. Esto implica revisar los requisitos con los stakeholders, realizar pruebas y simulaciones, utilizar herramientas como prototipos y modelos para validar los requisitos.

# Epicas e Historias de usuario

**Épica**: una épica es una gran cantidad de trabajo que es demasiado grande para completarse en un solo ciclo de desarrollo o sprint. Las épicas generalmente se dividen en historias más pequeñas y manejables para ayudar a los equipos a planificar y priorizar mejor su trabajo.

**Historia de usuario:** en el desarrollo de software, una historia se refiere a una historia de usuario, que es una descripción de una característica o funcionalidad desde la perspectiva de un usuario final. Las historias de usuarios ayudan a los equipos a comprender lo que los usuarios quieren y necesitan, lo que a su vez les ayuda a crear un mejor software.

# Técnicas para crear historias de usuario

Uso de plantillas, reglas y protocolos: Se pueden usar para responder una serie de preguntas que van a facilitar identificar lo que se espera del desarrollo, tales como:

- As a / *Como*, I Want / *Quiero*, So that / *Para*:
- Para el alcance: lo que está dentro (scope) y fuera (limitation)
- Reglas de negocio (business rules)
- Documentación (uso de herramientas tales como Confluence, Figma)
- Criterios de aceptación (Gherkin): Dado (given), Cuando (when) y Entonces (then)
- 3 C (card, conversation y confirmation)
- INVEST (Independent, Negotiable, Valuable, Estimable, Small, Testable)

# Recursos

- Criterios de aceptación (Gherkin): Gherkin Reference - Cucumber Documentation
- 3 C: 3 C's For Writing User Stories
- Invest: What Does Agile INVEST Stand For? | Wrike Agile Guide
- Dividir historias de usuario: The Humanizing Work Guide to Splitting User Stories
- Reporte de bugs: https://marker.io/blog/bug-report-template
- Plantillas para historias de usuario: plantillas-historias-usuario
- Example Mapping: example-mapping-introduction