

6. Ingeniería de requisitos

Índice

- Referencias.
- Introducción.
- Requisitos de sistema.
- El proceso de ingeniería de requisitos.
- Gestión de requisitos.
- IEEE Std. 830-1998.
 - 1. Introducción.
 - 2. Descripción general.
 - 3. Requisitos específicos.
- Conclusiones.

Índice

- Especificación de requisitos en UML
 - Introducción
 - Diagramas de casos de uso
 - Diagramas de actividades

Referencias

- IEEE Std. 830-1998. Recommended Practice for Software Requirements Specification, 1998
- Sommerville, I. *Ingeniería del Software*. 7^a edición. Addison-Wesley, 2005
- Pressman, R.S. *Ingeniería del Software. Un Enfoque Práctico. Sexta Edición*. McGraw-Hill, 2005

Introducción

- Entender la naturaleza de un problema por lo general es algo difícil → es difícil establecer que debe hacer un sistema software
- La descripción de los servicios y restricciones son los *requisitos* del sistema
- La *ingeniería de requisitos* es el proceso que permite identificar dichos requisitos

Introducción

- Podemos identificar dos tipos de requisitos:
 - Requisitos de usuario.
 - Requisitos de sistema.
- Los *requisitos de usuario* son frases en lenguajes natural junto a diagramas de los servicios que el sistema debe proporcionar, así como las restricciones bajo las que debe operar

Introducción

- Los *requisitos de sistema* determinan los servicios del sistema y las restricciones en detalle. Sirven como contrato.
- Es decir, son los mismo, pero a distinto nivel de detalle
 - e.g. req. usuario: el sistema debe hacer préstamos
 - e.g. reg. sistema: función préstamo; entrada: cód. socio, cód. ejemplar; salida: fecha devolución;

Requisitos de sistema

- Hay tres tipos de requisitos de sistema:
 - Requisitos funcionales.
 - Requisitos no funcionales
 - Requisitos del dominio.
- Los *requisitos funcionales* describen:
 - Los servicios que proporciona el sistema (funciones).
 - La respuesta del sistema ante determinadas entradas.
 - El comportamiento del sistema en situaciones particulares.

Requisitos de sistema

- En algunos casos, también determinan lo que no debería hacer el sistema.

- e.g.

5.4 función: alta usuario

descrita por: Pepe Sánchez, Biblioteca Ciencias Políticas

prioridad: alta

estabilidad: media

descripción: da de alta a un usuario de la biblioteca

entrada: nombre, primer apellido, segundo apellido, DNI, teléfono

salida: identificador de usuario

origen: interfaz usuario - bibliotecario

destino: sistema

necesita: *base de datos* de usuarios

acciones: comprobar que los datos son sintácticamente correctos y que no hay otro usuario con el mismo DNI. En caso positivo, dar de alta al usuario

precondición: no hay identificadores ni DNIs repetidos

postcondicion: no hay identificadores ni DNIs repetidos

efectos laterales: si falla la conexión enviar un mensaje al departamento de sistemas

Requisitos de sistema

- Los *requisitos no funcionales* son restricciones de los servicios o funciones que ofrece el sistema (e.g. tiempo, proceso de desarrollo, etc.)
 - e.g. la biblioteca debe ser capaz de atender simultáneamente a todas las bibliotecas de la UCM

Requisitos de sistema

- A su vez, hay tres tipos de requisitos no funcionales:
 - *Requisitos del producto*. Especifican el comportamiento del producto (e.g. prestaciones, memoria, tasa de fallos, etc.)
 - *Requisitos organizativos*. Se derivan de las políticas y procedimientos de las organizaciones de los clientes y desarrolladores (e.g. estándares de proceso, lenguajes de programación, etc.)
 - *Requisitos externos*. Se derivan de factores externos al sistema y al proceso de desarrollo (e.g. requisitos legislativos, éticos, etc.)

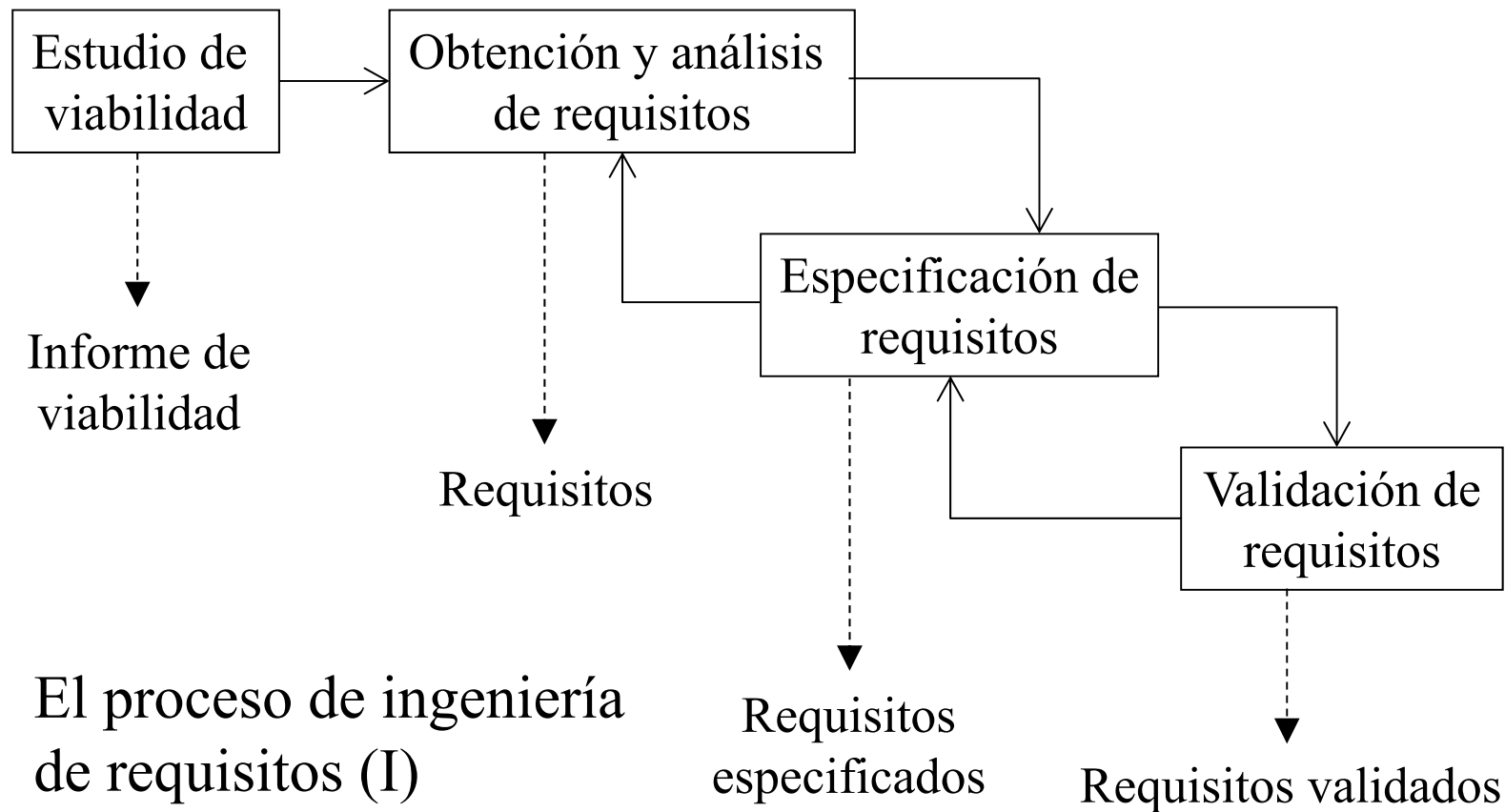
Requisitos de sistema

- Los *requisitos del dominio* se derivan del dominio de la aplicación y reflejan características de dicho dominio. Pueden ser funcionales o no funcionales
 - e.g. El sistema de biblioteca de la UCM debe ser capaz de exportar datos mediante el Lenguaje de Intercomunicación de Bibliotecas de España (LIBE).
 - e.g. El sistema de biblioteca no podrá acceder a bibliotecas con material *censurado*.

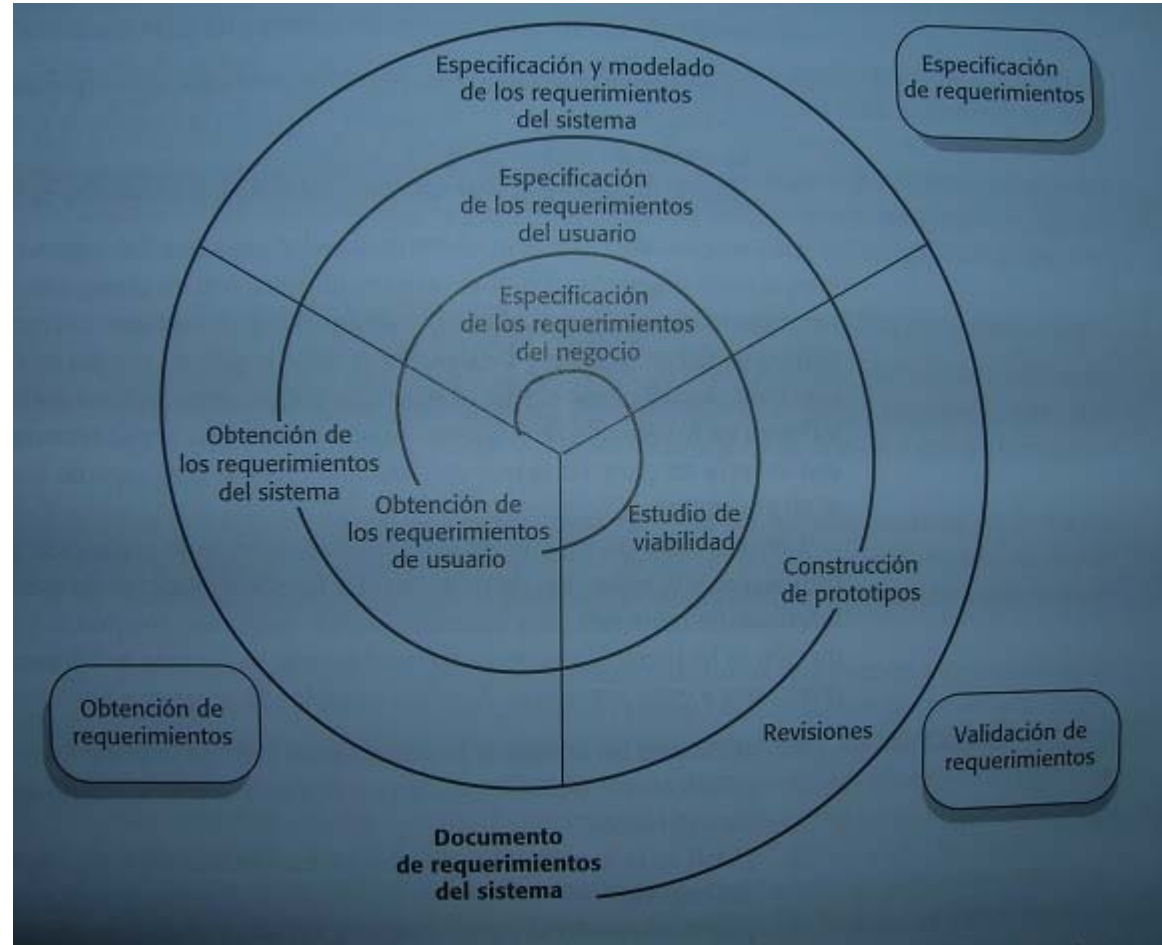
El proceso de ingeniería de req.

- La *ingeniería de requisitos* debe centrarse sobre lo *qué* hay que hacer, no sobre el *cómo*
- Tiene distintas fases que producen resultados diferentes

El proceso de ingeniería de req.



El proceso de ingeniería de req.



El proceso de ingeniería de req.

- Estudio de viabilidad
 - Estimación sobre si se puede resolver el problema del usuario con las tecnologías software y hardware disponible
 - El estudio decide si el sistema es rentable y puede desarrollarse cumpliendo las restricciones económicas
 - Se origina tras una especificación preliminar de los requisitos

El proceso de ingeniería de req.

- El estudio debe responder:
 - ¿Contribuye el sistema a los objetivos generales de la organización?
 - ¿Se puede implementar el sistema utilizando la tecnología actual y dentro de las restricciones de coste y tiempo?
 - ¿Puede integrarse el sistema con otros sistemas existentes en la organización?

El proceso de ingeniería de req.

- Análisis de requisitos
 - Interacción con los *stakeholders* (interesados) para determinar el dominio de la aplicación, así como los requisitos funcionales y no funcionales de la aplicación
 - Un *stakeholder* es cualquier persona que puede tener influencia directa o indirecta sobre los requisitos:
 - Clientes.
 - Usuarios.
 - Otros.

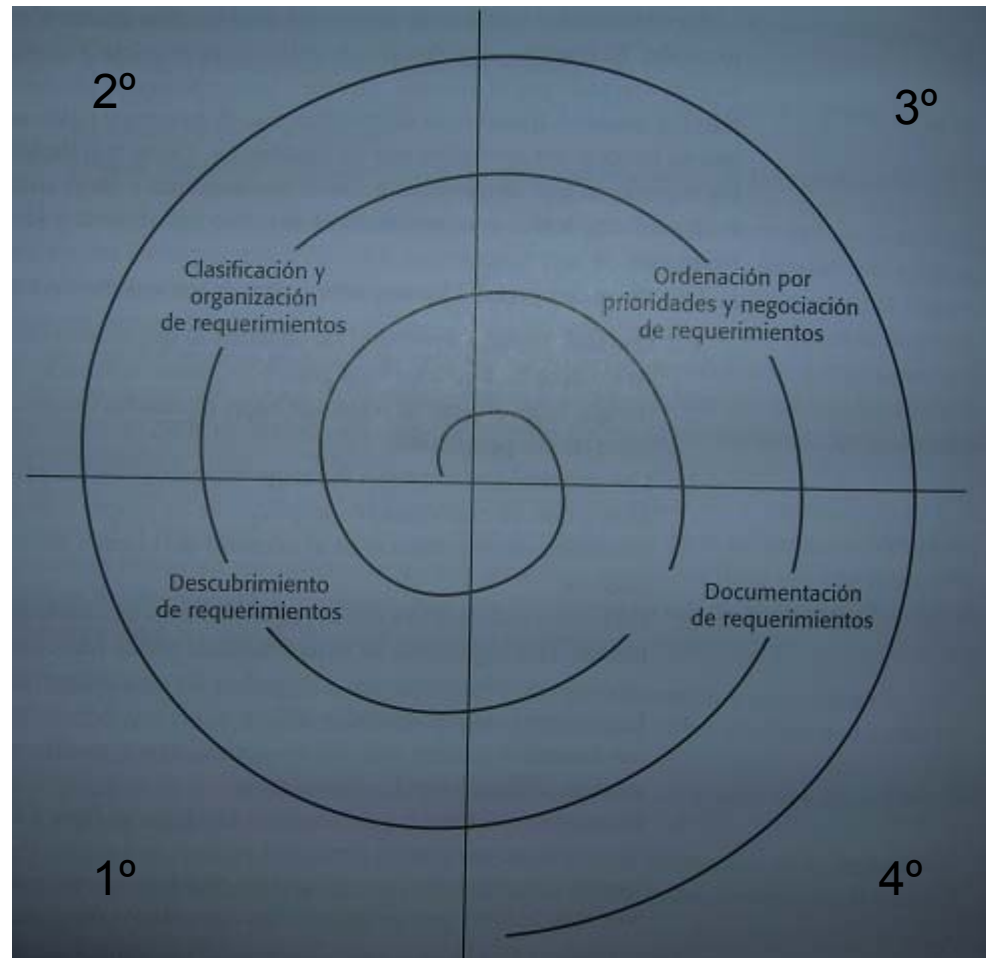
El proceso de ingeniería de req.

- Obtener y comprender los requisitos de los *stakeholders* es difícil por:
 - Pueden desconocer lo que esperan del sistema informático.
 - Es posible que no se conozca el dominio del sistema, familiar para los *stakeholders*.
 - Distintos *stakeholders* pueden tener distintos requisitos.
 - Influencia de factores políticos.
 - Entorno de negocio dinámico del sistema informático.

El proceso de ingeniería de req.

- El análisis de requisitos tiene el siguiente proceso:

El proceso de
obtención y
análisis de
requisitos



El proceso de ingeniería de req.

- Especificación de requisitos
 - Se fija una descripción detallada y precisa de los requisitos, de tal forma que sirva de base para un contrato entre el cliente y el desarrollador de software
 - En función del ciclo, los requisitos serán preliminares del negocio, de usuario o de sistema.
 - El objetivo es realizar la *especificación de requisitos software* SRS (*Software Requirements Specification*)

El proceso de ingeniería de req.

- Pueden utilizarse distintos lenguajes para especificar los requisitos del sistema:
 - Lenguaje natural.
 - Lenguaje natural estructurado.
 - Lenguaje de descripción de diseño.
 - Lenguaje de especificación de requisitos.
 - Notaciones gráficas.
 - Especificaciones matemáticas.

El proceso de ingeniería de req.

- Validación de requisitos
 - Se encarga de comprobar si los requisitos se ajustan a los deseos de los *stakeholders*.
 - Si la validación es inadecuada, se propagarán errores al diseño e implementación.
 - Evidentemente, tiene una fuerte repercusión sobre el coste.

El proceso de ingeniería de req.

- Hay distintos tipos de verificaciones en el proceso de validación de requisitos:
 - *Validez*. Comprobar la necesidad de las funciones definidas en la SRS.
 - *Consistencia*. Comprobar que no hay restricciones contradictorias o distintas descripciones de la misma función.
 - *Compleitud*. Comprobar que todos los requisitos y restricciones están incluidos en la SRS.
 - *Realismo*. Comprobar que se pueden implementar los requisitos con las tecnologías y plantilla disponible.
 - *Verificabilidad*. Comprobar que los requisitos pueden ser verificables con el fin de evitar problemas entre clientes y desarrolladores.

El proceso de ingeniería de req.

- Hay distintas técnicas para validar los requisitos, que pueden utilizarse conjunta o individualmente
 - *Revisión de requisitos.* Análisis sistemático de los requisitos por parte de un equipo de revisores.
 - *Prototipado.* Creación de un modelo ejecutable del sistema.
 - *Generación de casos de prueba.* Si no somos capaces de diseñar un caso de prueba para un requisitos, probablemente el requisito sea problemático.
 - *Análisis automático de la consistencia.* Supuesto que se haya utilizado una herramienta formal de especificación de requisitos.

Gestión de requisitos software

- Como ya hemos comentado, los requisitos pueden cambiar y evolucionar, ya que:
 - Puede haber requisitos variables en función del usuario.
 - El cliente no suele ser el usuario.
 - El producto y/o el entorno evoluciona.

Gestión de requisitos software

- La *gestión de requisitos* es el proceso de entender y controlar los cambios en los requisitos del sistema
- Tiene en cuenta aspectos técnicos no contemplados por la Gestión de la Configuración Software

Gestión de requisitos software

- Desde un punto de vista *evolutivo* los requisitos pueden ser:
 - *Estables*. Se derivan de la actividad principal del sistema y están directamente relacionados con el dominio.
 - *Volátiles*. Probablemente cambiarán durante el desarrollo del sistema o después de su entrega.

Gestión de requisitos software

- A su vez los volátiles pueden ser:
 - *Mutable*s. Cambian debido a cambios en el entorno en el que la organización opera (e.g. un nuevo sistema de numeración de firmas de las bibliotecas).
 - *Emergentes*. Aparecen durante el desarrollo del sistema por el mayor entendimiento por parte del cliente.
 - *De consecuencia*. Aparecen por la introducción del sistema.
 - *De compatibilidad*. Dependen de los procesos de negocio de la organización.

Gestión de requisitos software

- En la gestión de requisitos debemos ser capaces de:
 - Identificar los requisitos.
 - Tener un proceso de gestión del cambio.
 - Disponer de políticas de traza.
 - Disponer de soporte CASE

Gestión de requisitos software

- La *identificación* de cada requisito *se supone* al disponer de una SRS
- El *proceso de gestión de cambio* es común al de cualquier *Elemento de Configuración Software* (ECS)
- Las *políticas de traza* deben llevar cuenta de diversa *información de traza*:

Gestión de requisitos software

- *De origen*. Liga el requisito al *stakeholder*.
- *De requisitos*. Determina las dependencias entre requisitos.
- *De diseño*. Liga los requisitos a los módulos de diseño.
- La información de traza de requisitos puede representarse mediante una matriz de traza

Gestión de requisitos software

| Req. id | 1.1 | 1.2 | 1.3 | 2.1 | 2.2 | 2.3 | 3.1 | 3.2 |
|---------|-----|-----|-----|-----|-----|-----|-----|-----|
| 1.1 | | D | R | | | | | |
| 1.2 | | | D | | | R | | D |
| 1.3 | R | | | R | | | | |
| 2.1 | | | R | | D | | | D |
| 2.2 | | | | | | | | D |
| 2.3 | | R | | D | | | | |
| 3.1 | | | | | | | | R |
| 3.2 | | | | | | | R | |

D: depende de
R: relacionado con

Una matriz de traza

Gestión de requisitos software

- Las *herramientas CASE* facilitan:
 - El almacenamiento de requisitos.
 - La gestión del cambio.
 - La gestión de la traza.
- Pueden ser herramientas específicas, o gestionar el cambio de la SRS como el cambio de cualquier otro ECS

IEEE Std. 830-1998

Introducción

- El IEEE Std. 830-1998 se encarga de proporcionar unas normas para la creación de la Especificación de Requisitos Software (SRS: *Software Requirements Specification*)
- Objetivos de la SRS:
 - Establecer la base para un acuerdo entre clientes y desarrolladores sobre qué debe hacer el software.

IEEE Std. 830-1998

Introducción

- Reducir el esfuerzo de desarrollo.
- Proporcionar una base para la estimación de costes y planificación.
- Proporcionar una guía para la validación y verificación
- Facilitar la transferencia de software.
- Servir como base para mejoras posteriores.
- Según el IEEE una SRS debería identificar las siguientes características del sistema:

IEEE Std. 830-1998

Introducción

- *Funcionalidad*. Lo que hace el sistema.
- *Interfaces externas*. Forma de interactuar del sistema con las personas, el hardware del sistema, otro hardware y otro software.
- *Prestaciones*. Entre otras, velocidad, disponibilidad y tiempo de respuesta del sistema.
- *Atributos*. Portabilidad, corrección, mantenibilidad, seguridad y otros atributos.

IEEE Std. 830-1998

Introducción

- *Restricciones de diseño impuestas a la implementación.* Entre otras, estándares, lenguajes de implementación, entornos operativos, etc.
- Hay otras cuestiones que deben *excluirse* de la SRS:
 - Detalles de diseño o implementación.
 - Restricciones adicionales sobre el software.

IEEE Std. 830-1998

Introducción

- La SRS debe centrarse en *qué* hace el sistema, no en *cómo* lo hace, ni en *cómo* se construirá
- El estándar identifica una serie de *características* de una buena SRS:
 - *Correcta*. Cada requisito identificado es un requisitos del sistema.
 - *No ambigua*. Cada requisito identificado tiene una única interpretación.

IEEE Std. 830-1998

Introducción

- *Completa.* La SRS debe incluir:
 - Todos los requisitos relativos a funcionalidad, prestaciones, restricciones de diseño, atributos o interfaces externos.
 - Definición de las respuestas del sistema a todos los tipos posibles de datos de entrada en todas las posibles situaciones.
 - Todas las etiquetas y referencias a todas las figuras, tablas y diagramas en la SRS, así como definiciones de todos los términos y unidades de medida.

IEEE Std. 830-1998

Introducción

- *Consistente*. Consistencia interna con todos los requisitos en todos los documentos (i.e. requisitos de usuario vs. requisitos de sistema).
- *Priorizada*. La SRS deber priorizar la importancia y/o la estabilidad de cada requisito mediante un identificador de importancia y/o estabilidad para cada requisito.
- *Verificable*. Cada requisito identificado es verificable (i.e. se puede comprobar que el sistema implementa el requisito).

IEEE Std. 830-1998

Introducción

- *Modificable*. La estructura y estilo permiten cambios en los requisitos de forma fácil, completa y consistente manteniendo la estructura y estilo.
- *Trazable*. El origen de cada requisito está claro y facilita la referencia de cada requisito en desarrollos futuros o en mejoras de la documentación.

IEEE Std. 830-1998

Introducción

- Los interfaces entre distintos sistemas software, tales como interfaces de comunicación, interfaces software o interfaces de BDs pueden especificarse en un documento de especificación de interfaces que puede ser referenciado por diferentes SRSs.

IEEE Std. 830-1998

Introducción

- No es buena idea utilizar los requisitos de usuario en sustitución de la SRS, ya que
 - Puede ser correcta, consistente, verificable y modificable.
 - Pero no es no ambigua, completa y trazable.
- Los *beneficios* de la especificación de requisitos software se derivan del concepto de *calidad*

IEEE Std. 830-1998

Introducción

- El IEEE Std. 610.12 define *calidad* como:
 - Grado en que un sistema, componente o proceso cumple las especificaciones.
 - Grado en que un sistema, componente o proceso cumple las necesidades o deseos de clientes y usuarios.
- La SRS es vital para la calidad ya que es el documento que captura los criterios mediante los cuales se mide la calidad

IEEE Std. 830-1998

Introducción

- Sin una SRS es difícil o imposible:
 - Diseñar software.
 - Validar el software.
 - Gestionar el proyecto de software.
- La SRS es la base del desarrollo técnico
- Consecuentemente la SRS está inherentemente ligada al desarrollo de software

IEEE Std. 830-1998

Introducción

- Audiencia:
 - *Cliente*. La persona o personas que *pagan* por el producto y usualmente (pero no necesariamente) *deciden* los requisitos.
 - *Desarrollador*. La persona o personas que *producen* un producto para un cliente. El cliente y el desarrollador pueden pertenecer a la misma organización.
 - *Usuario*. La persona, o personas que *utilizan* o interactúan directamente con el producto. El usuario y el cliente habitualmente no son la misma persona.

IEEE Std. 830-1998

Introducción

- Clientes y desarrolladores deben realizar conjuntamente la SRS, ya que:
 - Frecuentemente, los clientes no entienden suficientemente bien las actividades de diseño y desarrollo como para proporcionar una SRS.
 - Frecuentemente, los desarrolladores no entienden el problema del cliente ni el dominio para especificar los requisitos de un sistema satisfactorio.

IEEE Std. 830-1998

Introducción

- Según el IEEE, la SRS se desarrolla durante la fase de *Análisis* de Requisitos
- Dicha SRS es la entrada de la fase de *Diseño*
- La SRS también se utiliza en las fases de *Implementación y Prueba*
- Es posible que la SRS tenga que ser actualizada según avanza el proyecto (GCS)

IEEE Std. 830-1998

Introducción

- Descripción detallada:

Tabla de contenidos

1. Introducción
 - 1.1 Propósito
 - 1.2 Alcance
 - 1.3 Definiciones, acrónimos y abreviaturas
 - 1.4 Referencias
 - 1.5 Resumen
2. Descripción general
 - 2.1 Perspectiva del producto
 - 2.2 Funciones del producto
 - 2.3 Características del usuario
 - 2.4 Restricciones
 - 2.5 Supuestos y dependencias
 - 2.6 Requisitos futuros

IEEE Std. 830-1998

Introducción

- 3. Requisitos específicos
 - 3.1 Interfaces externos
 - 3.2 Funciones
 - 3.3 Requisitos de rendimiento
 - 3.4 *Logical database requirements* (modelo del dominio)
 - 3.5 Restricciones de diseño
 - 3.6 Atributos del sistema software

Apéndices

Índice

IEEE Std. 830-1998

1. Introducción

- La *Introducción* (1) de la SRS proporciona una descripción de dicha SRS.
- La Introducción debe incluir:
 - 1.1 Propósito
 - Describe el propósito de la SRS.
 - Especifica la audiencia de la SRS.
 - 1.2 Alcance
 - Identifica el producto software por un nombre.

IEEE Std. 830-1998

1. Introducción

- Explica lo que hará, y si es necesario, lo que no hará el producto.
- Describe el uso del software desarrollado, incluyendo beneficios relevantes y objetivos.
- Es consistente con el resto de la SRS.
- 1.3 Definiciones acrónimos y abreviaturas
 - Proporciona las definiciones de todos los términos.

IEEE Std. 830-1998

1. Introducción

- 1.4 Referencias

- Proporciona una lista completa de todos los documentos referenciados en la SRS.
- Identifica cada documento por título, número de informe (si es necesario), fecha y organización que lo publica.
- Especifica las fuentes donde se pueden obtener las referencias.

- 1.5 Resumen

- Describe el resto de la SRS.
- Explica cómo está organizada la SRS.

IEEE Std. 830-1998

2. Descripción general

- La *Descripción General* (2) describe los factores generales que afectan al producto y sus requisitos
- No define requisitos específicos
- Proporciona un *background* para estos requisitos y los hace más comprensibles

IEEE Std. 830-1998

2. Descripción general

- La Descripción General debe incluir:
 - 2.1 Perspectiva del producto
 - Define el contexto de implantación del producto.
 - Si es parte de un sistema mayor, debe describir las relaciones entre ambos sistemas
 - Describe además:
 - *Interfaces del sistema*. Un resumen de los distintos interfaces del sistema
 - *Interfaces de usuario*. Características de configuración y uso de la interfaz de usuario
 - *Interfaces hardware*. Características de cada interfaz entre el software y el hardware.

IEEE Std. 830-1998

2. Descripción general

- *Interfaces software*. Uso de otros productos software e interfaces (nombre, alias, numero especificación, número versión, origen)
- *Interfaces de comunicación*. Tales como protocolos de red local, etc.
- *Memoria*. Características y límites de las memorias primaria y secundaria.
- *Operaciones*. Modos normales y especiales de operación por el usuario (e.g. niveles, operaciones de backup)
- *Requisitos de adaptación*. De instalaciones particulares.

IEEE Std. 830-1998

2. Descripción general

- 2.2 Funciones del producto
 - Describe las principales funciones del producto.
 - Podemos asimilarla a los *requisitos de usuario*.
- 2.3 Características de usuario
 - Nivel educativo, experiencia, capacidades.
- 2.4 Restricciones
 - Políticas de regulación.
 - Limitaciones hardware.
 - Interfaces con otras aplicaciones.

IEEE Std. 830-1998

2. Descripción general

- Operaciones en paralelo.
- Funciones de auditoria.
- Funciones de control.
- Requisitos de lenguaje de alto nivel.
- Protocolos de *signal handshake*.
- Requisitos de fiabilidad.
- Criticidad de la aplicación.
- Consideraciones de robustez y seguridad.

IEEE Std. 830-1998

2. Descripción general

- 2.5 Supuestos y dependencias
 - Factores que afectan a los requisitos.
- 2.6 Requisitos futuros
 - Versiones futuras.

IEEE Std. 830-1998

3. Requisitos específicos

- Los *Requisitos Específicos* (3) contienen todos los requisitos del sistema
- Es el núcleo de la SRS
- Características de los requisitos específicos:
 - Deben cumplir las características de toda SRS (t38).
 - Referencian a la información previa.
 - Son unívocamente identificables.
 - Deben organizarse de tal forma que facilite su legibilidad.

IEEE Std. 830-1998

3. Requisitos específicos

- Los Requisitos Específicos deben incluir:
 - 3.1 Interfaces externas
 - Descripción detallada de todas las entradas y salidas del sistema software.
 - Debe complementar la descripciones de interfaces de la sección 2 (Descripción General).
 - Debe incluir:
 - Nombre del elemento.
 - Descripción del propósito.
 - Origen de entrada o destino de salida.

IEEE Std. 830-1998

3. Requisitos específicos

- Rango válido, precisión y/o tolerancia.
- Unidades de medida.
- Temporización.
- Relaciones con otras entradas/salidas.
- Organización/formatos de pantalla.
- Organización/formatos de ventana.
- Formato de fechas.
- Formato de comandos.
- Mensajes de fin.

IEEE Std. 830-1998

3. Requisitos específicos

- 3.2 Funciones

- Definen las acciones fundamentales que deben tener lugar en el software durante la aceptación y procesamiento de la entrada y durante el procesamiento y generación de la salida.

- Incluyen:

- Controles de validez para las entradas.
- Secuencia exacta de operaciones.
- Respuestas ante situaciones anormales (desbordamiento, comunicación y manejo y recuperación de errores)

IEEE Std. 830-1998

3. Requisitos específicos

- Efectos de los parámetros.
- Relaciones de salidas a las entradas (secuencias de entrada/salida y formulas de conversión de entrada en salida).
- 3.3 Requisitos de rendimiento
 - Requisitos numéricos estáticos (e.g. número de terminales, número de usuarios, etc.) y dinámicos (e.g. número de transacciones por segundo) de rendimiento

IEEE Std. 830-1998

3. Requisitos específicos

- 3.4 *Logical database requirements* (modelo del dominio)

- Requisitos lógicos para cualquier información que resida en una base de datos (e.g.):

- Tipos de información utilizada por diversas funciones.
- Frecuencia de uso.
- Capacidades de acceso.
- Entidades de datos y sus relaciones.
- Restricciones de integridad.
- Requisitos de retención de datos.

- No es el modelo de los datos, es el modelo del dominio

IEEE Std. 830-1998

3. Requisitos específicos

- *Dominio* (RAE): ámbito real o imaginario de una actividad
- *Base de datos* (IEEE Std. 610.12-1990): colección de datos interrelacionados almacenados juntos en uno o más ficheros electrónicos

IEEE Std. 830-1998

3. Requisitos específicos

- 3.5 Restricciones de diseño
 - Impuestas por otros estándares, limitaciones hardware, etc.
 - Debe indicar si se ajusta a algún estándar.
- 3.6 Atributos del sistema software (e.g.)
 - Fiabilidad.
 - Disponibilidad.
 - Seguridad.
 - Mantenibilidad.
 - Portabilidad.

IEEE Std. 830-1998

3. Requisitos específicos

- Los Requisitos Específicos (3) se pueden organizar de diferentes formas:
 - *Modo del sistema*. Caracteriza al sistema en función de su modo de operación.
 - *Clase de usuario*. Funciones en base al tipo de usuario.
 - *Objetos*. Objetos del mundo real.

IEEE Std. 830-1998

3. Requisitos específicos

- *Característica*. Servicio externo del sistema que puede requerir una secuencia de entradas para producir el resultado esperado (e.g. llamada local).
- *Estímulo*. Describe las funciones en términos de *estímulos* (entradas).
- *Respuesta*. Agrupa las funciones en base a respuestas similares (e.g. altas, bajas, listados, etc.).
- *Jerarquía funcional*. Descrito en términos de una descomposición funcional.

IEEE Std. 830-1998

3. Requisitos específicos

- Se puede elegir la organización que más se ajuste a cada proyecto
- El estándar proporciona plantillas para cada organización

IEEE Std. 830-1998

3. Requisitos específicos

- e.g., para la organización por estímulos:

3. Requisitos específicos

3.1 Requisitos de interfaces externas

3.2 Requisitos funcionales

3.2.1 Estímulo 1

3.2.1.1 Requisito funcional 1.1

.....

3.2.1.n Requisito funcional 1.n

.....

3.2.m Estímulo m

3.2.m.1 Requisito funcional m.1

.....

3.2.m.k Requisito funcional m.k

IEEE Std. 830-1998

3. Requisitos específicos

3.3 Requisitos de rendimiento

3.4 Requisitos lógicos de la base de datos

3.5 Restricciones de diseño

3.6 Atributos del sistema software

- El estándar no determina el lenguaje en el que se debe describir cada requisito funcional (t63)

IEEE Std. 830-1998

3. Requisitos específicos

- Supuesto que elijamos *Lenguaje Natural Estructurado*, para cada requisito funcional deberíamos incluir:
 - Prioridad y nivel de estabilidad.
 - Descripción de la función.
 - Descripción de la entrada y su origen.
 - Descripción de la salida y su destino.
 - Indicación de otras entidades utilizadas.
 - Acción detallada de la función.
 - Pre y post condiciones.
 - Efectos laterales, si hay.

IEEE Std. 830-1998

3. Requisitos específicos

- e.g.
 - 5.4 función:** alta usuario
 - descrita por:** Pepe Sánchez, Biblioteca Ciencia Políticas
 - prioridad:** alta
 - estabilidad:** media
 - descripción:** da de alta a un usuario de la biblioteca
 - entrada:** nombre, primer apellido, segundo apellido, DNI, teléfono
 - salida:** identificador de usuario
 - origen:** interfaz usuario - bibliotecario
 - destino:** sistema
 - necesita:** *base de datos* de usuarios
 - acciones:** comprobar que los datos son sintácticamente correctos y que no hay otro usuario con el mismo DNI. En caso positivo, dar de alta al usuario
 - precondición:** no hay identificadores ni DNIs repetidos
 - postcondicion:** no hay identificadores ni DNIs repetidos
 - efectos laterales:** si falla la conexión enviar un mensaje al departamento de sistemas

IEEE Std. 830-1998

3. Requisitos específicos

- Donde:

función: *nombre del requisito funcional*

descrita por: *persona que especifica el requisito*

prioridad: *prioridad del requisito*

estabilidad: *estabilidad del requisito*

descripción: *descripción de alto nivel del requisito*

entrada: *datos de entrada a través de la interfaz que invoca la operación*

salida: *respuesta proporcionada por el sistema*

origen: *interfaz de entrada y usuario que la invoca*

destino: *destino de la petición del usuario*

necesita: *información que no está en la entrada y es necesaria para ejecutar la operación*

acciones: *descripción detallada del flujo de ejecución sin nombrar ningún elemento concreto de implementación*

precondición: *lo que se puede afirmar antes de introducir los datos*

postcondicion: *lo que se puede afirmar tras ejecutar la operación*

efectos laterales: *acciones no relacionadas directamente con la operación*

IEEE Std. 830-1998

3. Requisitos específicos

- Cada requisito funcional se puede representar como un *caso de uso* UML
- A su vez los casos de uso UML pueden especificarse:
 - Con lenguaje natural estructurado.
 - **Con un *diagrama de actividad* UML**
 - Otro.

IEEE Std. 830-1998

3. Requisitos específicos

- Nota:
 - ¿Hay que volver a introducir el código de socio cada vez si se va a llevar tres ejemplares?
 - ¿Cuál es el límite de préstamos por socio?

Espec. Requisitos en UML

Introducción

- El *Unified Modeling Language*, UML es una notación de modelado que nos permite especificar:
 - Requisitos
 - Diseños
 - Arquitecturas
 - Despliegues

Espec. Requisitos en UML

Introducción

- Veremos aquí la parte de la notación utilizada para especificar requisitos:
 - Diagramas de casos de uso
 - Diagramas de actividades

Espec. Requisitos en UML

Diagramas de casos de uso

- Un *diagrama de casos de uso* es un diagrama que muestra un conjunto de casos de uso, actores y sus relaciones
- Un *caso de uso* es una descripción de un conjunto de secuencias de acciones, incluyendo variantes, que ejecuta un sistema para producir un resultado observable de valor para un actor

Espec. Requisitos en UML

Diagramas de casos de uso

- Un *actor* representa un conjunto coherente de roles que los usuarios de los casos de uso juegan al interactuar con estos. Normalmente un actor es un rol que es jugado por una persona, un dispositivo hardware o incluso otro sistema al interactuar con nuestro sistema
- Un caso de uso es un requisito funcional y el actor su usuario

Espec. Requisitos en UML

Diagramas de casos de uso

- Los elementos más comunes de los diagramas de casos de uso son:

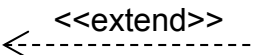
- Caso de uso: 

- Actor: 
nombre actor

- Relación:

- Generalización: 

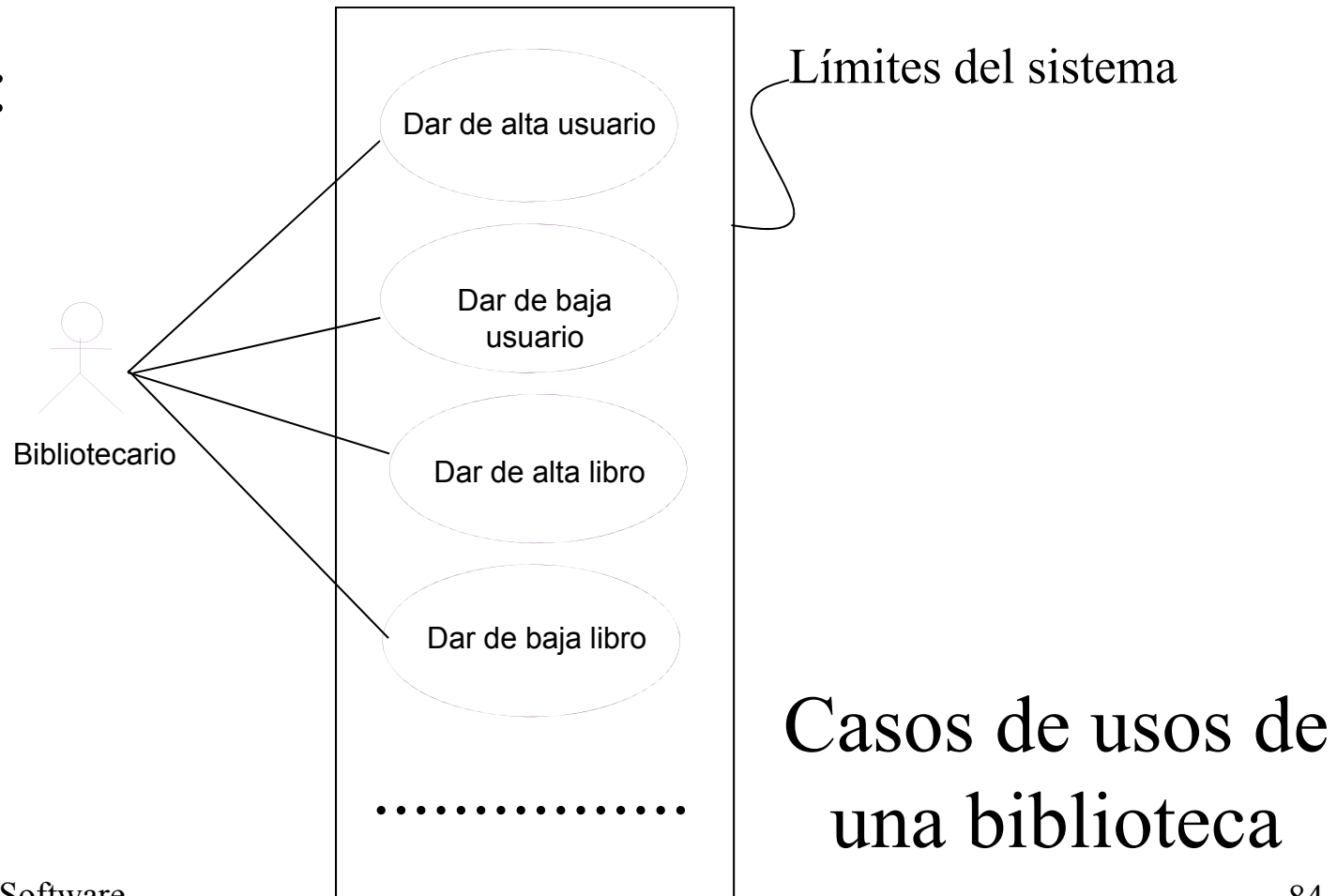
- Inclusión: 

- Extensión: 

Espec. Requisitos en UML

Diagramas de casos de uso

- Ej.:



Espec. Requisitos en UML

Diagramas de casos de uso

- Un caso de uso describe *qué* hace un sistema, pero no *cómo* lo hace
- El comportamiento de un caso de uso se puede especificar describiendo un flujo de eventos
- Dicho flujo de eventos se puede especificar:
 - Con lenguaje natural estructurado.
 - Pseudocódigo.
 - Diagramas de actividades

Espec. Requisitos en UML

Diagramas de casos de uso

- Cuando se describe el flujo de eventos se debe incluir:
 - Cómo y cuándo empieza y acaba el caso de uso.
 - Cuándo interactúa el sistema con los actores.
 - Qué elementos se intercambian.
 - El flujo principal y los flujos alternativos* de comportamiento.

*Si se desean se pueden clasificar (e.g., Datos incompletos, datos erróneos, comportamiento alternativo, etc.)

Espec. Requisitos en UML

Diagramas de casos de uso

- Ej. Dar alta usuario
 - Flujo de eventos principal:
 1. Introducir nombre, primer apellido, segundo apellido, DNI, tel.
 2. Comprobar corrección sintáctica datos
 3. Comprobar que no hay ya un usuario en la BD con ese DNI
 4. Introducir usuario en la BD el ejemplar.
 5. Obtener ID usuario nuevo
 6. Mostrar ID al usuario
 - Flujo de eventos alternativo:
 - En cualquier momento se puede cancelar la operación.
 - Si algún dato no es correcto, se comunica al usuario.
 - Si el usuario ya existía, se comunica al usuario.

Espec. Requisitos en UML

Diagramas de casos de uso

- El flujo de evento son las *acciones* de la plantilla que vimos en IEEE Std. 830-1998
- También se pueden incluir pre y postcondiciones en los casos de uso
- Ej.: dar de alta libro

Precondición: no hay identificadores ni DNIs repetidos

.....

Postcondición: no hay identificadores ni DNIs repetidos

Espec. Requisitos en UML

Diagramas de actividades

- Un *diagrama de actividades* muestra un flujo de actividades
- Están formados por nodos y arcos
- En UML 1.x se utilizaba la nomenclatura de *estados y transiciones*
- Los utilizaremos para describir casos de uso, i.e., las acciones de la plantilla que vimos en el IEEE Std. 830-1998

Espec. Requisitos en UML

Diagramas de actividades

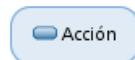
- Los nodos pueden ser:
 - De acción: unidades atómicas de trabajo dentro de la actividad
 - De actividad: conjunto de acciones
 - De control: controlan el flujo en la actividad
 - De objetos: objetos usados en la actividad

Espec. Requisitos en UML

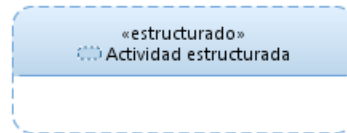
Diagramas de actividades

- Los elementos más comunes de los diagramas de actividades son:

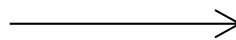
- Acción:



- Actividad



- Transición:



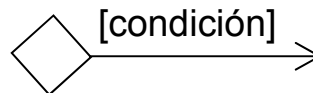
- Nodo inicial:



- Nodo final:



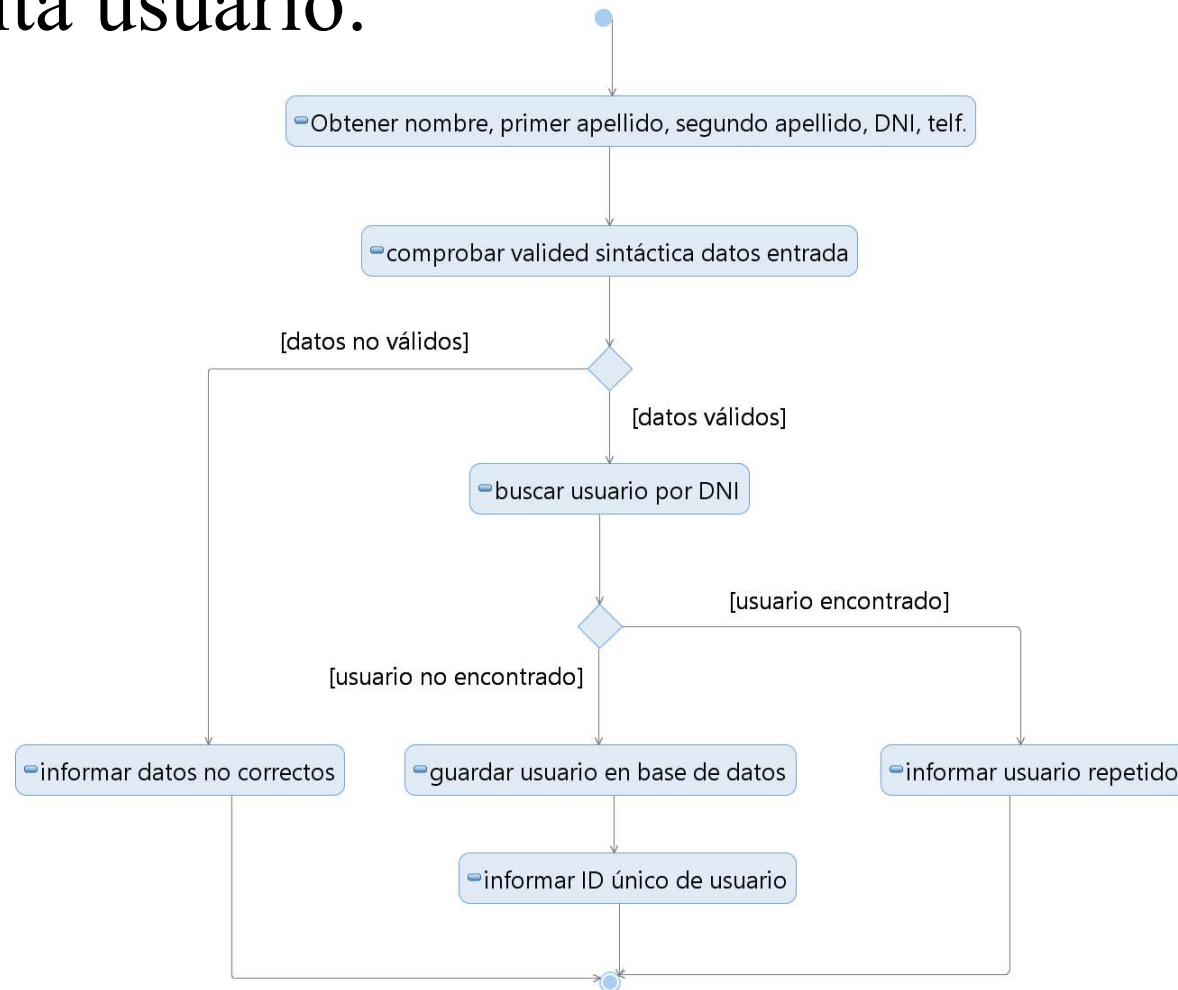
- Decisión:



Espec. Requisitos en UML

Diagramas de actividades

- Ej. Alta usuario:



Conclusiones

- Ingeniería de requisitos: fundamental en IS
- Requisitos de usuario y del sistema
- Requisitos funcionales, no funcionales y del dominio
- Fases en ingeniería de requisitos
- Gestión de requisitos del software
- Requisitos estables y volátiles
- IEEE Std. 830-1998
- Lenguaje específicos requisitos