



---

## Resumen Académico: Comprensión de los Requerimientos

### Capítulo 5 - Ingeniería de Software

**Autor:** Roger S. Pressman

**7ma Edición**

**Fecha:** 2024

---

#### 1. Introducción

La comprensión de los requerimientos constituye una de las tareas más desafiantes en la ingeniería de software. Como señala Brooks (citado en Pressman, 2010), "La parte más difícil al construir un sistema de software es decidir qué construir" (p. 102). Este capítulo aborda las técnicas y metodologías necesarias para establecer una base sólida en el proceso de requerimientos.

#### 2. Ingeniería de Requerimientos

##### 2.1 Definición y Alcance

La ingeniería de requerimientos se define como el conjunto amplio de tareas y técnicas que llevan a entender los requerimientos de software. Según Pressman (2010), esta disciplina "proporciona el mecanismo apropiado para entender lo que desea el cliente, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable" (p. 102).

##### 2.2 Componentes Principales

La ingeniería de requerimientos incluye siete tareas fundamentales:

1. **Concepción:** Establecimiento del entendimiento básico del problema
2. **Indagación:** Proceso de obtención de información de los participantes
3. **Elaboración:** Refinamiento y expansión de los requerimientos
4. **Negociación:** Resolución de conflictos entre requerimientos
5. **Especificación:** Documentación formal de los requerimientos
6. **Validación:** Verificación de la calidad y consistencia
7. **Administración:** Control y seguimiento de cambios

*Figura 1*

*Tareas fundamentales de la ingeniería de requerimientos*

### 3. Proceso de Establecimiento de Bases

Sommerville y Sawyer (citados en Pressman, 2010) definen participante como "cualquier persona que se beneficie en forma directa o indirecta del sistema en desarrollo" (p. 106). Los participantes típicos incluyen:

- ### 3.2 Múltiples Puntos de Vista

*Figura 2*

## Participantes típicos en el proceso de requerimientos y sus enfoques

Participante	Enfoque Principal
Mercadotecnia	Funciones que estimulen el mercado
Gerentes	Características dentro del presupuesto
Usuarios finales	Facilidad de uso y aprendizaje
Ingenieros de software	Infraestructura técnica
Soporte	Facilidad de mantenimiento

**Nota.** Tomado de Pressman (2010, p. 106), muestra los distintos enfoques según el tipo de participante.



---

### 3.3 Preguntas Fundamentales

Pressman (2010) propone tres categorías de preguntas iniciales:

#### Preguntas sobre participantes y beneficios:

- ¿Quién está detrás de la solicitud?
- ¿Quién usará la solución?
- ¿Cuál será el beneficio económico?

#### Preguntas sobre el problema:

- ¿Cuál sería una "buena" salida?
- ¿Qué problemas resolvería?
- ¿Hay restricciones de desempeño?

#### Metapreguntas:

- ¿Es usted la persona indicada?
- ¿Mis preguntas son relevantes?
- ¿Debería preguntar algo más?

## 4. Indagación de Requerimientos

### 4.1 Problemas Comunes

Christel y Kang (citados en Pressman, 2010) identifican tres categorías principales de problemas:

1. **Problemas de alcance:** Fronteras mal definidas del sistema
2. **Problemas de entendimiento:** Incertidumbre sobre necesidades
3. **Problemas de volatilidad:** Cambios constantes en los requerimientos

### 4.2 Recabación Colaborativa

El enfoque colaborativo sigue lineamientos específicos:

- Participación conjunta de ingenieros y participantes
- Establecimiento de reglas claras
- Uso de facilitadores
- Empleo de mecanismos de definición

#### *Figura 3*

*Fases de una reunión colaborativa para la recolección de requerimientos*



REUNIÓN COLABORATIVA
<p>Preparación:</p> <ul style="list-style-type: none"> <li>• Solicitud de producto</li> <li>• Listas de objetos</li> <li>• Servicios y restricciones</li> </ul> <p>Durante la reunión:</p> <ul style="list-style-type: none"> <li>• Presentación de listas</li> <li>• Análisis y consenso</li> <li>• Desarrollo de miniespecificaciones</li> </ul> <p>Resultado:</p> <ul style="list-style-type: none"> <li>• Lista consensuada</li> <li>• Especificaciones preliminares</li> </ul>

**Nota.** Adaptado de Pressman (2010, p. 111), se describe el proceso estructurado para realizar reuniones efectivas en la etapa de indagación.

## 5. Especificación de Requerimientos

### 5.1 Especificación de Requerimientos de Software (ERS)

Una ERS formal se justifica cuando:

- El desarrollo es realizado por terceros
- La falta de especificación crearía problemas severos
- El sistema es extremadamente complejo
- Se trata de un negocio de importancia crítica

### 5.2 Estructura de la ERS

Según el formato de Wiegers (citado en Pressman, 2010), una ERS debe incluir:

1. **Introducción:** Propósito, alcance y referencias
2. **Descripción general:** Perspectiva y características del producto
3. **Características del sistema:** Funcionalidades específicas
4. **Requerimientos de interfaz:** Usuario, hardware, software
5. **Requerimientos no funcionales:** Desempeño, seguridad
6. **Apéndices:** Glosario, modelos, conceptos

## 6. Validación de Requerimientos

### 6.1 Mecanismos de Validación

La validación se realiza principalmente a través de revisiones técnicas que evalúan:

- Claridad en el enunciado
- Identificación de fuentes
- Cuantificación de requerimientos
- Relaciones entre requerimientos
- Capacidad de prueba

## 6.2 Lista de Verificación

Pressman (2010) propone preguntas clave para la validación:

- ¿Los requerimientos están claramente enunciados?
- ¿Está identificada la fuente del requerimiento?
- ¿El requerimiento está acotado cuantitativamente?
- ¿Puede someterse a prueba?
- ¿Es posible rastrearlo hasta los objetivos del sistema?

## 7. Administración de Requerimientos


La administración de requerimientos es definida como "el conjunto de actividades que ayudan al equipo del proyecto a identificar, controlar y dar seguimiento a los requerimientos y a sus cambios" (Pressman, 2010, p. 105).

### 7.1 Herramientas Tecnológicas

*Figura 4*

*Ingeniería de requerimientos*

HERRAMIENTAS DE SOFTWARE



**Ingeniería de requerimientos**

**Objetivo:** Las herramientas de la ingeniería de los requerimientos ayudan a reunir éstos, a modelarlos, administrarlos y validarlos.

**Mecánica:** La mecánica de las herramientas varía. En general, éstas elaboran varios modelos gráficos (por ejemplo, UML) que ilustran los aspectos de información, función y comportamiento de un sistema. Estos modelos constituyen la base de todas las demás actividades del proceso de software.

**Herramientas representativas:**  
En el sitio de Volere Requirements, en [www.volere.co.uk/tools.htm](http://www.volere.co.uk/tools.htm), se encuentra una lista razonablemente amplia (y actualizada) de herramientas para la ingeniería de requerimientos. En los capítulos 6 y 7 se estudian las herramientas que sirven para modelar aquéllos. Las que se mencionan a continuación se centran en su administración.

*EasyRM*, desarrollada por Cybernetic Intelligence GmbH ([www.easy-rm.com](http://www.easy-rm.com)), construye un diccionario/glosario especial para proyectos, que contiene descripciones y atributos detallados de los requerimientos.

*Rational RequisitePro*, elaborada por Rational Software ([www-306.ibm.com/software/awdtools/reqpro/](http://www-306.ibm.com/software/awdtools/reqpro/)), permite a los usuarios construir una base de datos de requerimientos, representar relaciones entre ellos y organizarlos, indicar su prioridad y rastrearlos.

En el sitio de Volere ya mencionado, se encuentran muchas herramientas adicionales para administrar requerimientos, así como en la dirección [www.jiludwig.com/Requirements\\_Management\\_Tools.html](http://www.jiludwig.com/Requirements_Management_Tools.html)

**Nota.** Imagen tomada de *Ingeniería de software: Un enfoque práctico* (7ma ed., p. 106), por R. S. Pressman, 2010, McGraw-Hill. Identifica los principales problemas de alcance, entendimiento y volatilidad (citados en Pressman, 2010).

Las herramientas de ingeniería de requerimientos incluyen:



- **EasyRM:** Diccionario/glosario especializado
- **Rational RequisitePro:** Base de datos de requerimientos
- **Volere Requirements:** Conjunto integral de herramientas

## 8. Consideraciones Finales

### 8.1 Importancia Crítica

Como advierte Jones (citado en Pressman, 2010), "Las semillas de los desastres enormes del software por lo general se vislumbran en los tres primeros meses del inicio del proyecto" (p. 102). Esto subraya la importancia crítica de una ingeniería de requerimientos rigurosa.

### 8.2 Desafíos Continuos

Los requerimientos enfrentan desafíos constantes:

- Cambios durante el desarrollo
- Múltiples interpretaciones
- Conflictos entre participantes
- Limitaciones técnicas y presupuestarias

## 9. Conclusiones

La comprensión de los requerimientos constituye el fundamento del éxito en proyectos de software. Pressman (2010) enfatiza que "diseñar y construir un elegante programa de cómputo que resuelva el problema equivocado no satisface las necesidades de nadie" (p. 101).

Las técnicas presentadas no constituyen una solución definitiva, pero proporcionan un enfoque estructurado para enfrentar uno de los aspectos más desafiantes de la ingeniería de software. La colaboración efectiva entre todos los participantes, junto con la aplicación sistemática de las técnicas de ingeniería de requerimientos, aumenta significativamente las probabilidades de éxito del proyecto.

La comprensión de los requerimientos constituye una fase fundamental en el proceso de ingeniería de software. Este capítulo aborda las técnicas y metodologías esenciales para la identificación, análisis y documentación de los requerimientos del sistema, estableciendo las bases para el desarrollo exitoso de software.

### *Figura 5*

*Conducción de una reunión para recabar requerimientos*



### CASA SEGURA



#### Conducción de una reunión para recabar los requerimientos

**La escena:** Sala de juntas. Está en marcha la primera reunión para recabar los requerimientos.

**Participantes:** Jamie Lazar, integrante del equipo de software; Vinod Raman, miembro del equipo de software; Ed Robbins, miembro del equipo de software; Doug Miller, gerente de ingeniería de software; tres trabajadores de mercadotecnia; un representante de ingeniería del producto, y un facilitador.

#### La conversación:

**Facilitador (apunta en un pizarrón):** De modo que ésta es la lista actual de objetos y servicios para la función de seguridad del hogar.

**Persona de mercadotecnia:** Eso la cubre, desde nuestro punto de vista.

**Vinod:** ¿No dijo alguien que quería que toda la funcionalidad de CasaSegura fuera accesible desde internet? Eso incluiría la función de seguridad, ¿o no?

**Persona de mercadotecnia:** Sí, así es... tendremos que añadir esa funcionalidad y los objetos apropiados.

**Facilitador:** ¿Agrega eso algunas restricciones?

**Jamie:** Sí, tanto técnicas como legales.

**Representante del producto:** ¿Qué significa eso?

**Jamie:** Nos tendríamos que asegurar de que un extraño no pueda ingresar al sistema, desactivarlo y robar en el lugar o hacer algo peor. Mucha responsabilidad sobre nosotros.

**Doug:** Muy cierto.

**Mercadotecnia:** Pero lo necesitamos así... sólo asegúrense de impedir que ingrese un extraño.

**Ed:** Eso es más fácil de decir que de hacer.

**Facilitador (interrumpe):** No quiero que debatamos esto ahora. Anotémoslo como un aspecto y continuemos.

(Doug, que es el secretario de la reunión, toma debida nota.)

**Facilitador:** Tengo la sensación de que hay más por considerar aquí.

(El grupo dedica los siguientes 20 minutos a mejorar y aumentar los detalles de la función de seguridad del hogar.)

Nota. Imagen tomada de Ingeniería de software: Un enfoque práctico (7ma ed., cap. 5, p. 111), por R. S. Pressman, 2010, McGraw-Hill.

## 2. Despliegue de la Función de Calidad (DFC)

### 2.1 Concepto y Propósito

El Despliegue de la Función de Calidad (DFC) es una técnica de administración de la calidad que traduce las necesidades del cliente en requerimientos técnicos específicos para el software. Su objetivo principal es maximizar la satisfacción del cliente a través del proceso de ingeniería de software (Pressman, 2010).

### 2.2 Tipos de Requerimientos según el DFC

El DFC identifica tres categorías fundamentales de requerimientos:

#### Requerimientos Normales:

- Objetivos y metas establecidos durante las reuniones con el cliente
- Su presencia garantiza la satisfacción básica del cliente
- Ejemplos: tipos de gráficos, funciones específicas del sistema, niveles de rendimiento

#### Requerimientos Esperados:

- Están implícitos en el producto o sistema
- Su ausencia causa gran insatisfacción
- Ejemplos: facilidad de interacción humano-máquina, operación confiable, facilidad de instalación

#### Requerimientos Emocionantes:



- Características que van más allá de las expectativas del cliente
- Su presencia genera alta satisfacción
- Ejemplos: funcionalidades innovadoras como pantalla táctil, correo de voz visual

## **2.3 Metodología del DFC**

El DFC utiliza diversas técnicas para la recopilación de información:

- Entrevistas con clientes
- Observación directa
- Encuestas
- Análisis de datos históricos

Esta información se organiza en la "tabla de la voz del cliente", que posteriormente se analiza mediante diagramas, matrices y métodos de evaluación.

## **3. Escenarios de Uso**

### **3.1 Definición y Propósito**

Los escenarios de uso, también conocidos como casos de uso, proporcionan descripciones detalladas sobre cómo los usuarios finales utilizarán las funciones y características del sistema. Estos escenarios son fundamentales para comprender la naturaleza de los usos previstos para el sistema.

### **3.2 Proceso de Desarrollo**

El desarrollo de escenarios de uso implica:

1. Identificación de actores y usuarios
2. Definición de interacciones específicas
3. Descripción de flujos de trabajo
4. Documentación de excepciones y variaciones

## **4. Productos del Trabajo de la Indagación**

### **4.1 Elementos Principales**

La indagación de requerimientos genera los siguientes productos del trabajo:

- Enunciado de la necesidad y su factibilidad
- Enunciado acotado del alcance del sistema
- Lista de participantes en la indagación
- Descripción del ambiente técnico
- Lista organizada de requerimientos y restricciones
- Conjunto de escenarios de uso
- Prototipos para definición de requerimientos





---

## 4.2 Proceso de Revisión

Todos los productos del trabajo deben ser revisados por todas las personas que participan en la indagación de requerimientos, garantizando así la validación y el consenso.

## 5. Desarrollo de Casos de Uso

### 5.1 Concepto Fundamental

Según Cockburn, un caso de uso captura un contrato que describe el comportamiento del sistema bajo diferentes condiciones en respuesta a peticiones de sus participantes (Pressman, 2010). Esencialmente, narra una historia estilizada sobre la interacción usuario-sistema.

### 5.2 Actores en los Casos de Uso

**Definición de Actor:** Un actor es cualquier entidad externa que se comunica con el sistema. Representa los roles que desempeñan las personas o dispositivos durante la operación del sistema.

#### **Tipos de Actores:**

- **Actores Principales:** Interactúan directamente para lograr la función requerida
- **Actores Secundarios:** Proporcionan apoyo al sistema

### 5.3 Elementos Clave de un Caso de Uso

Un caso de uso efectivo debe responder a las siguientes preguntas:

- ¿Quién es el actor principal y secundario?
- ¿Cuáles son los objetivos de los actores?
- ¿Qué precondiciones deben existir?
- ¿Qué tareas principales realiza el actor?
- ¿Qué excepciones deben considerarse?
- ¿Qué variaciones son posibles?
- ¿Qué información maneja el actor?

### 5.4 Formato Estructurado

El formato recomendado para casos de uso detallados incluye:

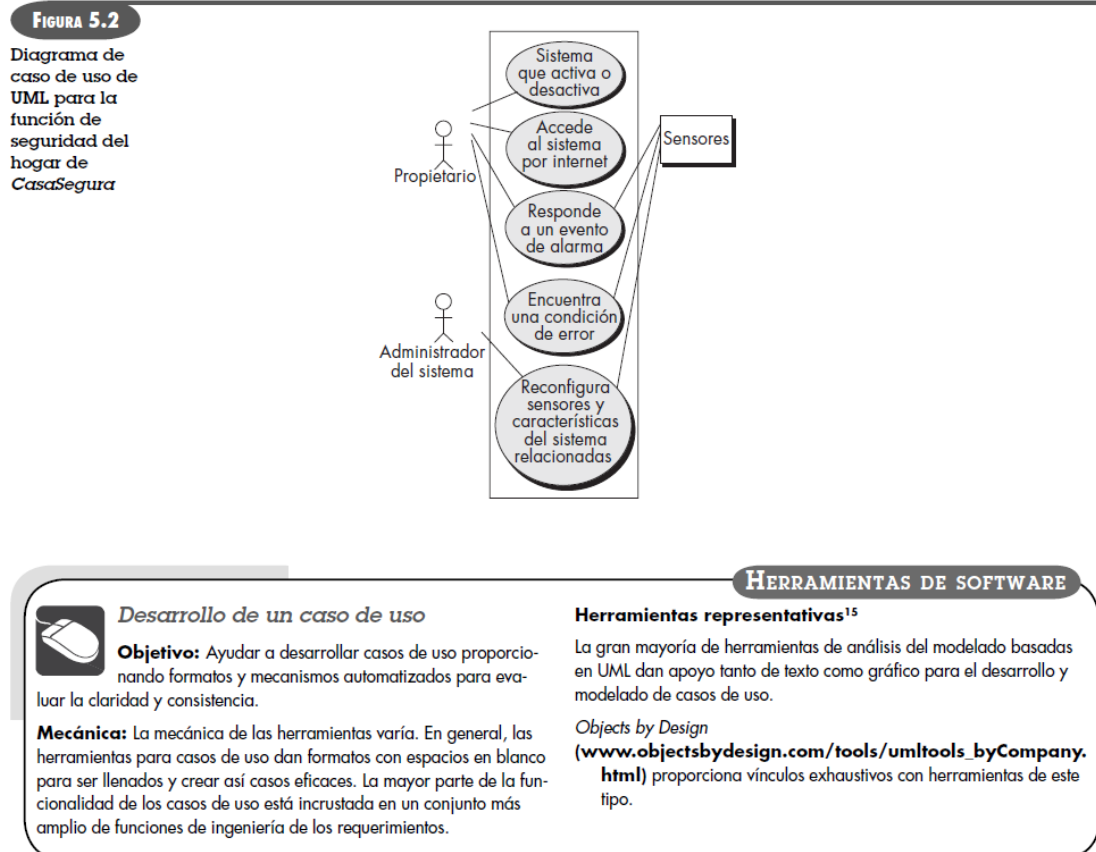
- **Caso de uso:** Nombre identificativo
- **Actor principal:** Usuario primario
- **Objetivo en contexto:** Propósito específico
- **Precondiciones:** Condiciones iniciales
- **Disparador:** Evento que inicia el caso de uso
- **Escenario:** Secuencia de acciones



- **Excepciones:** Situaciones alternativas
- **Prioridad:** Importancia relativa
- **Frecuencia de uso:** Periodicidad esperada

Figura 6

Diagrama de caso de uso UML



Nota. Nota. Imagen tomada de Ingeniería de software: Un enfoque práctico (7ma ed., cap. 5, p. 117), por R. S. Pressman, 2010, McGraw-Hill

## 6. Elaboración del Modelo de Requerimientos

### 6.1 Propósito del Modelo

El modelo de requerimientos describe los dominios de información, función y comportamiento necesarios para un sistema basado en computadora. Es dinámico y evolutivo, cambiando a medida que se comprende mejor el sistema.

### 6.2 Elementos del Modelo

**Elementos Basados en Escenarios:**

- Descripción del sistema desde la perspectiva del usuario



- Casos de uso básicos y elaborados
- Diagramas de casos de uso

#### **Elementos Basados en Clases:**

- Identificación de objetos y su clasificación
- Definición de atributos y operaciones
- Diagramas de clases UML

#### **Elementos de Comportamiento:**

- Diagramas de estado
- Representación de modos de comportamiento
- Eventos y transiciones de estado

#### **Elementos Orientados al Flujo:**

- Modelado de transformación de información
- Flujos de entrada, procesamiento y salida
- Representación de la arquitectura de datos



Figura 7

## Diagrama y notación UML

FIGURA 5.3

Diagramas de actividad del UML para indagar los requerimientos

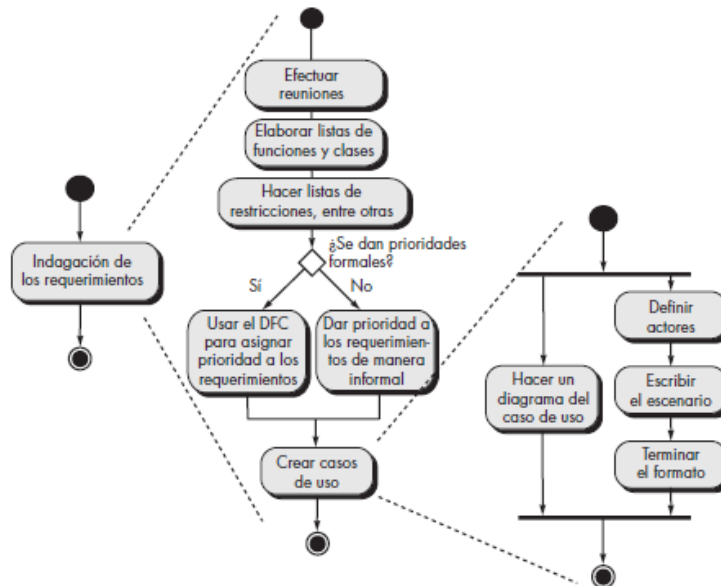


FIGURA 5.4

Diagrama de clase para un sensor

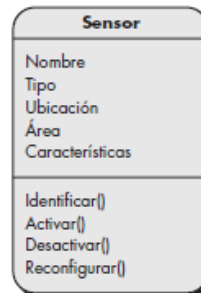
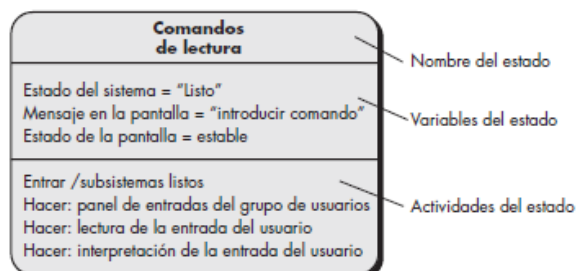


FIGURA 5.5

Notación UML del diagrama de estado



Nota. Imagen tomada de Ingeniería de software: Un enfoque práctico (7ma ed., cap. 5, p. 119), por R. S. Pressman, 2010, McGraw-Hill

## 6.3 Patrones de Análisis

Los patrones de análisis proporcionan soluciones reutilizables para problemas recurrentes en dominios específicos de aplicación. Ofrecen dos beneficios principales:



1. **Aceleración del desarrollo:** Proporcionan modelos reutilizables con ejemplos
2. **Mejora de la calidad:** Incluyen descripción de ventajas y limitaciones

## 7. Caso de Estudio: CasaSegura

### 7.1 Contexto del Sistema

A lo largo del capítulo se utiliza el sistema CasaSegura como ejemplo práctico, el cual incluye funcionalidades de seguridad para el hogar con las siguientes características:

- Sistema de vigilancia con sensores
- Panel de control para interacción del usuario
- Acceso remoto vía internet
- Múltiples modos de operación (permanece/fuera)

### 7.2 Actores Identificados

Para el sistema CasaSegura se identificaron los siguientes actores:

- **Propietario de la casa:** Usuario principal del sistema
- **Gerente de arranque:** Responsable de la configuración inicial
- **Sensores:** Dispositivos de detección
- **Subsistema de vigilancia:** Estación central de monitoreo

### 7.3 Caso de Uso Ejemplo: IniciarVigilancia

El caso de uso IniciarVigilancia ilustra el proceso completo de activación del sistema:

#### Escenario Principal:

1. El propietario observa el panel de control
2. Introduce la clave de acceso
3. Selecciona modo de operación (permanecer/fuera)
4. Confirma la activación mediante indicador luminoso

#### Excepciones Consideradas:

- Sistema no preparado
- Clave incorrecta
- Fallos de comunicación
- Diferentes modos de activación

## 8. Herramientas de Software

### 8.1 Herramientas para Desarrollo de Casos de Uso



---

Las herramientas especializadas proporcionan:

- Formatos estructurados para casos de uso
- Mecanismos de evaluación de claridad y consistencia
- Funcionalidades integradas de ingeniería de requerimientos

## **8.2 Herramientas de Modelado UML**

La mayoría de herramientas de análisis basadas en UML ofrecen:

- Soporte gráfico y textual
- Desarrollo de diagramas de casos de uso
- Integración con otros elementos del modelo

## **9. Conclusiones y Recomendaciones**

### **9.1 Importancia de la Comprensión de Requerimientos**

La comprensión adecuada de los requerimientos constituye la base fundamental para el éxito de cualquier proyecto de software. La aplicación sistemática de técnicas como el DFC, los casos de uso y el modelado de requerimientos garantiza:

- Mejor alineación con las necesidades del cliente
- Reducción de errores en fases posteriores
- Mejora en la satisfacción del usuario final
- Optimización del proceso de desarrollo

### **9.2 Mejores Prácticas**

Para una indagación efectiva de requerimientos se recomienda:

- Involucrar activamente a todos los participantes
- Utilizar múltiples técnicas de recopilación
- Documentar exhaustivamente los casos de uso
- Revisar continuamente los productos del trabajo
- Mantener la trazabilidad de los requerimientos

### **9.3 Consideraciones Futuras**

El modelo de requerimientos debe considerarse como un documento vivo que evoluciona a medida que se comprende mejor el sistema. La flexibilidad y adaptabilidad son características esenciales para el éxito del proceso de ingeniería de requerimientos.

## **5.6 Requerimientos De Las Negociaciones**

### **Introducción**



La comprensión de los requerimientos constituye una de las tareas más críticas y desafiantes en el desarrollo de software. Como establece Brooks: "La parte más difícil al construir un sistema de software es decidir qué construir" (Pressman, 2010, p. 102). Este capítulo aborda las técnicas y procesos necesarios para establecer una base sólida para el diseño y construcción de sistemas de software.

## Conceptos Fundamentales

### Definición de Ingeniería de Requerimientos

La **ingeniería de requerimientos** es el conjunto amplio de tareas y técnicas que conducen a la comprensión de los requerimientos del software. Constituye una acción fundamental de la ingeniería de software que inicia durante la actividad de comunicación y continúa en el modelado (Pressman, 2010).

### Características Principales

### Proceso de Ingeniería de Requerimientos

#### 1. Concepción

- **Definición:** Establece el entendimiento básico del problema y las personas que requieren una solución
- **Actividades principales:**
  - Identificación de necesidades del negocio
  - Análisis de factibilidad de alto nivel
  - Definición del alcance del proyecto
  - Establecimiento de comunicación preliminar

#### 2. Indagación

- **Propósito:** Recabar información sobre objetivos, funcionalidades y restricciones del sistema
- **Problemas comunes identificados por Christel y Kang:**
  - Problemas de alcance
  - Problemas de entendimiento
  - Problemas de volatilidad

#### 3. Elaboración

- **Objetivo:** Desarrollar un modelo refinado de los requerimientos
- **Elementos clave:**
  - Creación de escenarios de usuario
  - Identificación de clases de análisis
  - Definición de atributos y servicios
  - Establecimiento de relaciones entre clases

#### 4. Negociación



- **Necesidad:** Reconciliar requerimientos conflictivos entre diferentes participantes
- **Proceso:**
  - Priorización de requerimientos
  - Análisis de costo y riesgo
  - Resolución de conflictos internos
  - Búsqueda de soluciones que satisfagan a todas las partes

## 5. Especificación

- **Formatos posibles:**
  - Documento escrito
  - Modelos gráficos
  - Modelo matemático formal
  - Escenarios de uso
  - Prototipos

## 6. Validación

- **Métodos principales:**
  - Revisiones técnicas
  - Análisis de consistencia
  - Verificación de completitud
  - Detección de errores y omisiones

## 7. Administración

- **Propósito:** Identificar, controlar y dar seguimiento a los requerimientos y sus cambios

## Establecimiento de Bases

### Identificación de Participantes

Los **participantes** son todas las personas que se benefician directa o indirectamente del sistema en desarrollo (Sommerville & Sawyer, 1997). Incluyen:

- Gerentes de operaciones
- Personal de mercadotecnia
- Clientes internos y externos
- Usuarios finales
- Consultores
- Ingenieros de software
- Personal de mantenimiento

### Múltiples Puntos de Vista

Cada participante aporta una perspectiva única:





- **Mercadotecnia:** Funciones vendibles
- **Gerencia:** Presupuesto y tiempo
- **Usuarios finales:** Facilidad de uso
- **Ingenieros:** Infraestructura técnica

## Preguntas Iniciales

Gause y Weinberg (1989) proponen tres categorías de preguntas:

1. **Preguntas centradas en el cliente:**
  - ¿Quién está detrás de la solicitud?
  - ¿Quién usará la solución?
  - ¿Cuál será el beneficio económico?
2. **Preguntas sobre el problema:**
  - ¿Qué problemas resolverá?
  - ¿Cuál sería una buena salida?
  - ¿Hay restricciones especiales?
3. **Metapreguntas:**
  - ¿Es usted la persona indicada?
  - ¿Mis preguntas son relevantes?
  - ¿Debería preguntar algo más?

## Indagación Colaborativa de Requerimientos

### Lineamientos Básicos

1. Participación conjunta de ingenieros y participantes
2. Establecimiento de reglas de preparación
3. Agenda formal pero flexible
4. Facilitador para controlar la reunión
5. Mecanismos de definición apropiados

## Proceso de Reunión

### *Preparación*

- Distribución de solicitud de producto
- Elaboración de listas individuales:
  - Objetos del sistema
  - Servicios requeridos
  - Restricciones
  - Criterios de desempeño

### *Durante la Reunión*

- Presentación de listas individuales
- Creación de listas combinadas
- Eliminación de redundancias
- Desarrollo de miniespecificaciones



---

## **Ejemplo Práctico: Sistema CasaSegura**

### **Solicitud de Producto (Extracto)**

"La función de seguridad del hogar protegería varias 'situaciones' indeseables, como acceso ilegal, incendio y niveles de monóxido de carbono. Emplearía sensores inalámbricos para detectar cada situación."

### **Objetos Identificados**

- Panel de control
- Detectores de humo
- Sensores en ventanas y puertas
- Detectores de movimiento
- Alarma
- Eventos (activación de sensores)

### **Servicios Requeridos**

- Configurar el sistema
- Preparar la alarma
- Vigilar los sensores
- Marcar el teléfono
- Programar el panel de control

### **Especificación de Requerimientos de Software (ERS)**

#### **Estructura Propuesta por Wiegers**

- 1. Introducción**
  - Propósito
  - Alcance del proyecto
  - Referencias
- 2. Descripción General**
  - Perspectiva del producto
  - Características del producto
  - Clases de usuario
- 3. Características del Sistema**
  - Características específicas
- 4. Requerimientos de Interfaz Externa**
  - Interfaces de usuario
  - Interfaces de hardware
  - Interfaces de software
- 5. Requerimientos No Funcionales**
  - Desempeño
  - Seguridad
  - Calidad


### **Validación de Requerimientos**



## Lista de Verificación

*Figura 8*

### *Lista de verificación para validad requerimientos*



**Lista de verificación para validar requerimientos**

Con frecuencia es útil analizar cada requerimiento en comparación con preguntas de verificación. A continuación se presentan algunas:

INFORMACIÓN

- ¿Los requerimientos están enunciados con claridad? ¿Podrían interpretarse mal?
- ¿Está identificada la fuente del requerimiento (por ejemplo, una persona, reglamento o documento)? ¿Se ha estudiado el planteamiento final del requerimiento en comparación con la fuente original?
- ¿El requerimiento está acotado en términos cuantitativos?
- ¿Qué otros requerimientos se relacionan con éste? ¿Están comparados con claridad por medio de una matriz de referencia cruzada u otro mecanismo?

- ¿El requerimiento viola algunas restricciones del dominio?
- ¿Puede someterse a prueba el requerimiento? Si es así, ¿es posible especificar las pruebas (en ocasiones se denominan criterios de validación) para ensayar el requerimiento?
- ¿Puede rastrearse el requerimiento hasta cualquier modelo del sistema que se haya creado?
- ¿Es posible seguir el requerimiento hasta los objetivos del sistema o producto?
- ¿La especificación está estructurada en forma que lleva a entenderlo con facilidad, con referencias y traducción fáciles a productos del trabajo más técnicos?
- ¿Se ha creado un índice para la especificación?
- ¿Están enunciadas con claridad las asociaciones de los requerimientos con las características de rendimiento, comportamiento y operación? ¿Cuáles requerimientos parecen ser implícitos?

- Nota. Imagen tomada de Ingeniería de software: Un enfoque práctico (7ma ed., cap. 5, p. 105), por R. S. Pressman, 2010, McGraw-Hill
- ¿Están enunciados con claridad?
- ¿Está identificada la fuente?
- ¿El requerimiento está acotado cuantitativamente?
- ¿Puede someterse a prueba?
- ¿Es trazable hasta los objetivos del sistema?

## Mecanismos de Validación

- **Revisiones técnicas:** Método principal
- **Análisis de consistencia:** Verificación de coherencia
- **Verificación de completitud:** Detección de omisiones
- **Detección de conflictos:** Identificación de inconsistencias

## Herramientas de Apoyo

### Herramientas Representativas

- **EasyRM:** Construcción de diccionarios especializados
- **Rational RequisitePro:** Base de datos de requerimientos
- **Herramientas de modelado UML:** Representación gráfica

## Funcionalidades Principales

- Administración de requerimientos
- Trazabilidad
- Control de cambios
- Generación de reportes



---

## Consideraciones Especiales

### Administración de Requerimientos

La administración de requerimientos incluye:

- Identificación de cambios
- Control de modificaciones
- Seguimiento de evolución
- Mantenimiento de trazabilidad

### Factores Críticos de Éxito

1. **Comunicación efectiva** entre participantes
2. **Colaboración activa** de todos los involucrados
3. **Documentación clara** y precisa
4. **Validación continua** de requerimientos
5. **Gestión proactiva** de cambios

### Conclusiones

La comprensión de los requerimientos es fundamental para el éxito de cualquier proyecto de software. Como señala Jones: "Las semillas de los desastres enormes del software por lo general se vislumbran en los tres primeros meses del inicio del proyecto" (Pressman, 2010, p. 102).

Los elementos clave para una ingeniería de requerimientos exitosa incluyen:

- **Proceso sistemático:** Seguir las siete tareas fundamentales
- **Participación activa:** Involucrar a todos los stakeholders
- **Comunicación efectiva:** Establecer canales claros de información
- **Validación continua:** Verificar constantemente la precisión
- **Gestión de cambios:** Prepararse para la evolución de requerimientos

La implementación efectiva de estas técnicas proporciona una base sólida para el diseño y construcción de sistemas de software que verdaderamente satisfagan las necesidades del usuario final.

---

### Referencias

Pressman, R. S. (2010). *Ingeniería de software: Un enfoque práctico* (7ma ed.). McGraw-Hill.

Christel, M. G., & Kang, K. C. (1992). *Issues in requirements elicitation*. Software Engineering Institute.



Gause, D. C., & Weinberg, G. M. (1989). *Exploring requirements: Quality before design*. Dorset House.

Sommerville, I., & Sawyer, P. (1997). *Requirements engineering: A good practice guide*. John Wiley & Sons.

Wiegers, K. E. (2003). *Software requirements* (2nd ed.). Microsoft Press.