Ingeniería de Software

Ingeniería de Requisitos
Clase 4

Sommerville capítulo 4 – secciones 4.4.4, 4.5 y 4.6
Wiegers – capítulos 12, 16 y 17

Modelado del sistema

- El modelado del sistema es el proceso de desarrollar modelos abstractos del sistema, con cada modelo se presentan diferentes visiones o perspectivas.
- Ayuda a los analistas a entender las funcionalidades del sistema y facilita la comunicación con los clientes.
- En la actualidad es una actividad que se basa generalmente en la notación UML.

Modelado de sistemas existentes y nuevos

- Los modelos de sistemas existentes se utilizan durante la ingeniería de requisitos. Ayudan a comprender qué hacen y pueden ser usados para discusiones acerca de sus fortalezas y debilidades.
- Los modelos de nuevos sistemas son usados durante la ingeniería de requisitos para ayudar a explicar los requisitos propuestos a otros stakeholders del sistema. También para discutir propuestas de diseño y documentar el sistema para su implementación.
- En un proceso de ingeniería guiado por modelos (model-driven), es posible generar de forma total o parcial una implementación del modelo del sistema.

Perspectivas del sistema - Ejemplo

- **Una perspectiva externa** — se modela el contexto o entorno del sistema.
- **Una perspectiva de la interacción** — se modelan las interacciones entre un sistema y su entorno o entre un sistema y sus componentes.
- **Una perspectiva estructural** — se modela la organización del sistema o la estructura de los datos que se procesan por el sistema.
- **Una perspectiva del comportamiento** — se modela el comportamiento dinámico del sistema y la forma en que responde a los eventos.

Perspectivas del sistema - UML

Tipos de diagramas considerados como esenciales (encuesta 2007)

- **Diagramas de actividad** — muestran las actividades involucradas en un proceso o en el procesamiento de datos.
- **Diagramas de casos de uso** — muestran las interacciones entre el sistema y su entorno.
- **Diagramas de secuencia** — muestran las interacciones entre el sistema y los componentes del sistema.
- **Diagrama de clases** — muestran las clases de objetos del sistema y las asociaciones entre ellas.
- **Diagramas de estados** — muestran como el sistema reacciona ante eventos internos o externos.

Uso de modelos gráficos

- Como medio de facilitar la discusión acerca de sistemas existentes o propuestos.
  - Ya sea que los modelos estén incompletos o incorrectos está bien debido a que su papel es aportar a la discusión.
- Como una manera de documentar un sistema existente.
  - Los modelos deben ser una representación exacta del sistema pero no tiene que ser completa.
- Como una descripción detallada del sistema que se puede usar para generar una implementación del sistema.
  - Los modelos tienen que ser a la vez correctos y completos.

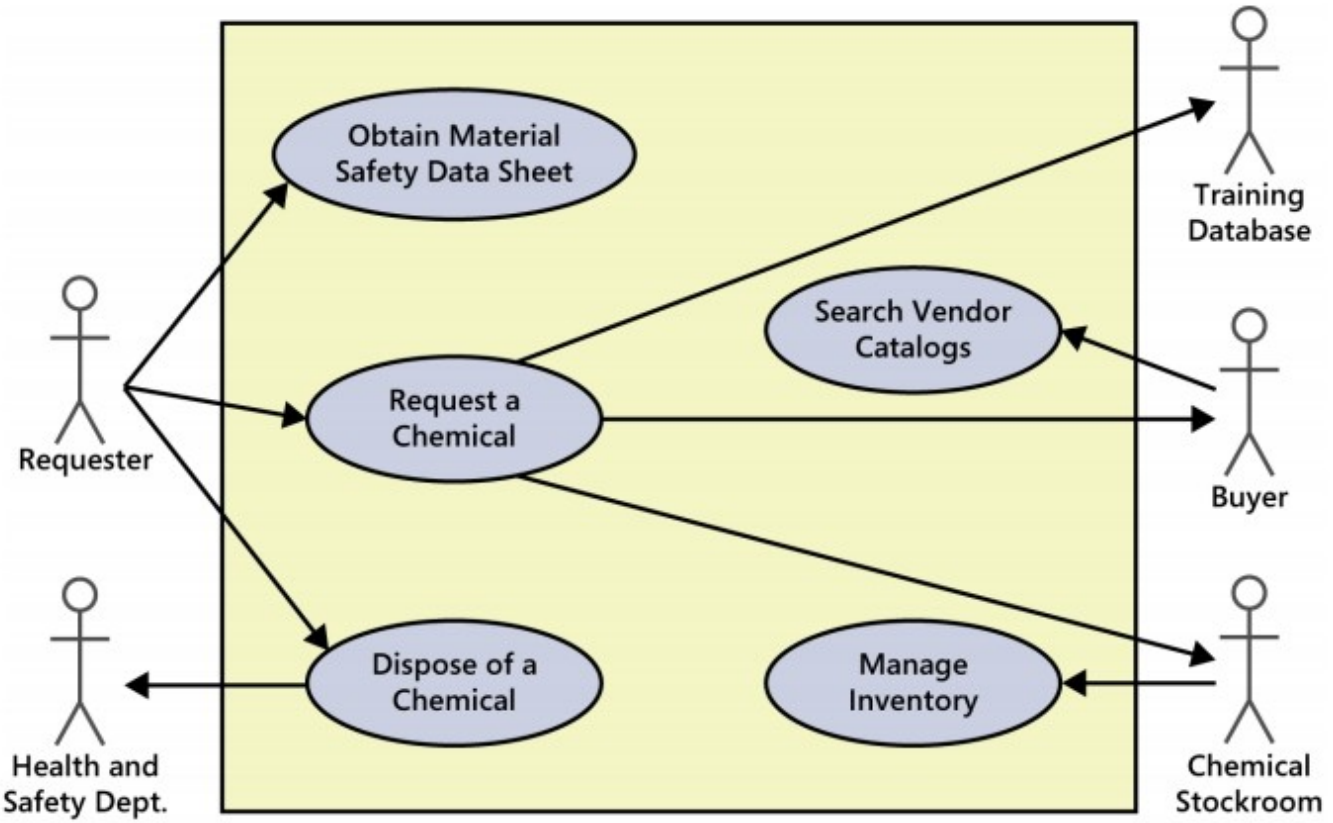
Modelos de contexto

- Son usados para ilustrar el contexto operacional del sistema – muestran lo que está fuera de los límites del sistema.
- Temas sociales y organizacionales pueden afectar la decisión sobre donde situar los límites del sistema.

- **Límites del Sistema**
  - Definen qué está dentro y qué está fuera del sistema.
  - La definición de los limites del sistema tiene un profundo efecto sobre los requisitos del sistema.

Modelos de contexto – Ejemplo

- Podemos utilizar un diagrama de casos de uso que incluya la frontera del sistema como modelo de contexto.



Modelos de interacción

- Todo sistema involucra interacción: interacción con usuarios, con otros sistemas o entre sus componentes.
- Modelar la interacción con los usuarios ayuda a identificar los requisitos de usuario.
- Modelar la interacción con otros sistemas permite explorar posibles problemas de comunicación.
- Modelar la interacción entre los componentes ayuda a entender sí la estructura propuesta del sistema es apropiada para los requisitos y atributos de calidad establecidos.

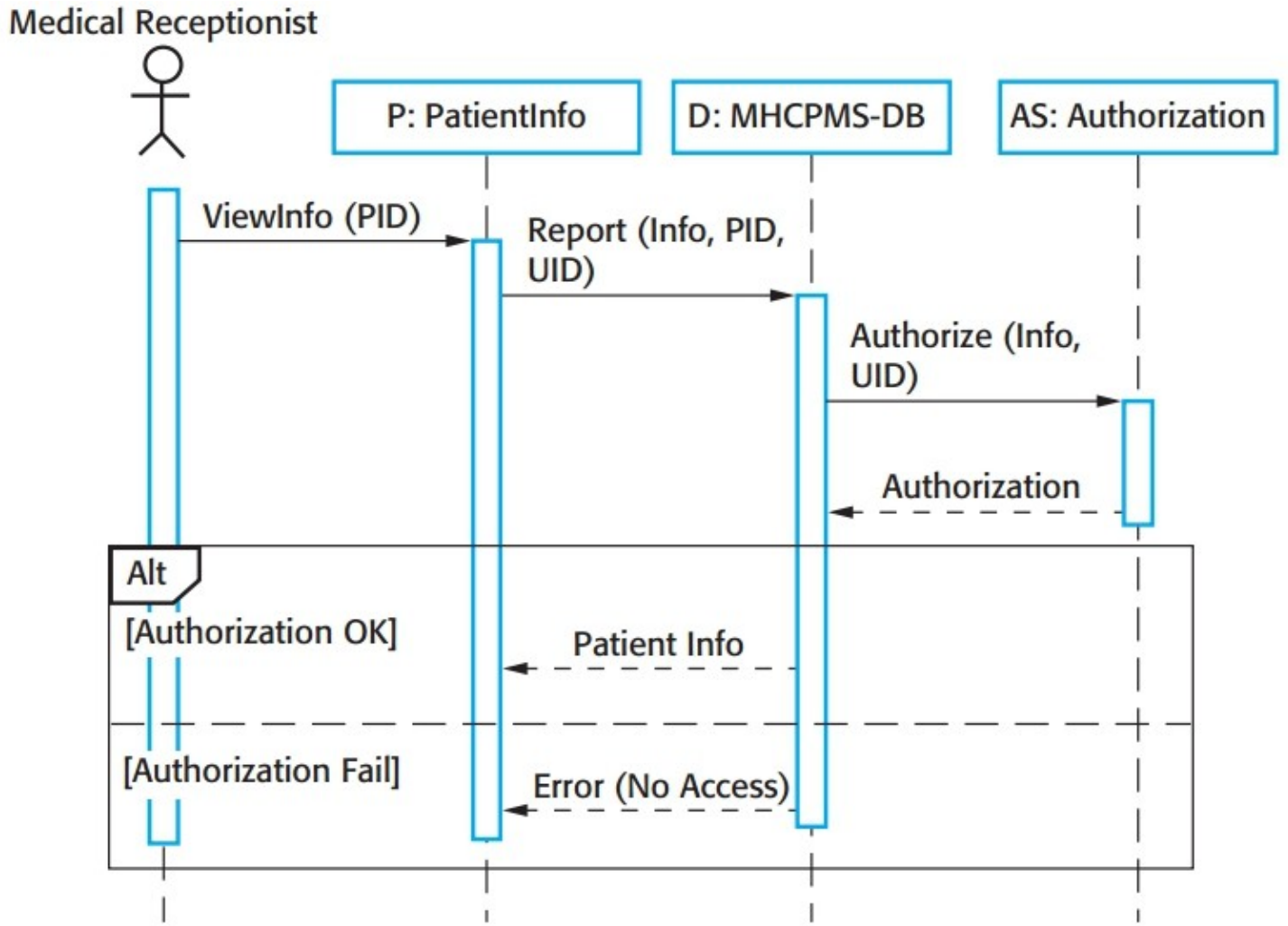
Modelos de interacción

- En UML podemos modelar la interacción de un sistema utilizando:
  - Modelado con casos de uso.
  - Diagramas de secuencia.

Modelos de interacción

- **Diagramas de secuencia**
  - Son usados para modelar las interacciones entre los actores y los objetos dentro de un sistema.
  - Un diagrama de secuencia muestra la secuencia de interacciones que tienen lugar durante un caso de uso particular o una instancia de caso de uso.
  - Los objetos y actores involucrados son enumerados a lo largo de la parte superior del diagrama con una línea vertical de puntos a partir desde allí hacía abajo.
  - Las interacciones entre objetos son indicadas mediante flechas con anotaciones.

Modelos de interacción



Modelos estructurales

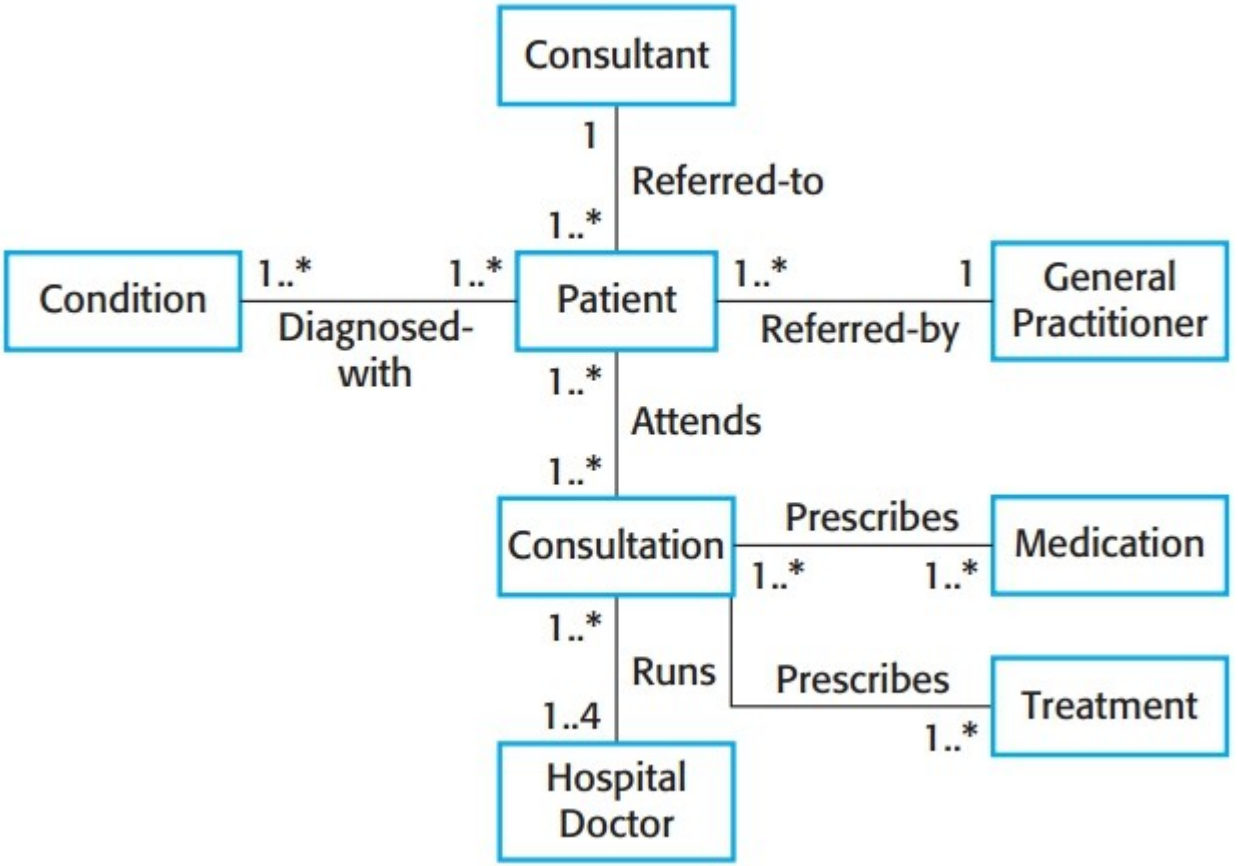
- Disponen la organización del sistema en términos de los componentes que lo componen y sus relaciones.
- Pueden ser modelos estáticos, los cuales muestran la estructura del sistema diseñado o modelos dinámicos los cuales muestran la organización del sistema cuando se está ejecutando.
- Se pueden crear modelos estructurales del sistema cuando se está discutiendo y diseñando la arquitectura del sistema.

Modelos estructurales

- **Diagramas de clases**
  - Muestran las clases de un sistema y sus asociaciones.
  - Una clase puede ser considerada como una definición general de un tipo de objeto del sistema.
  - Una asociación se representa como un enlace entre dos o más clases.
  - Cuando se desarrollan modelos durante fases tempranas los objetos representan algo del mundo real (modelo conceptual).

Modelos estructurales

- **Diagramas de clases – Ejemplo**



Modelos de comportamiento

- Modelan el comportamiento dinámico de un sistema en ejecución.
  - Muestran qué sucede o qué debe suceder cuando un sistema responde a los estímulos de su entorno.

- Se puede pensar en dos tipos de estímulos:
  - **Datos** — Ciertos datos que llegan y deben ser procesados.
  - **Eventos** — Ciertos eventos que funcionan como disparadores.

Modelos de comportamiento

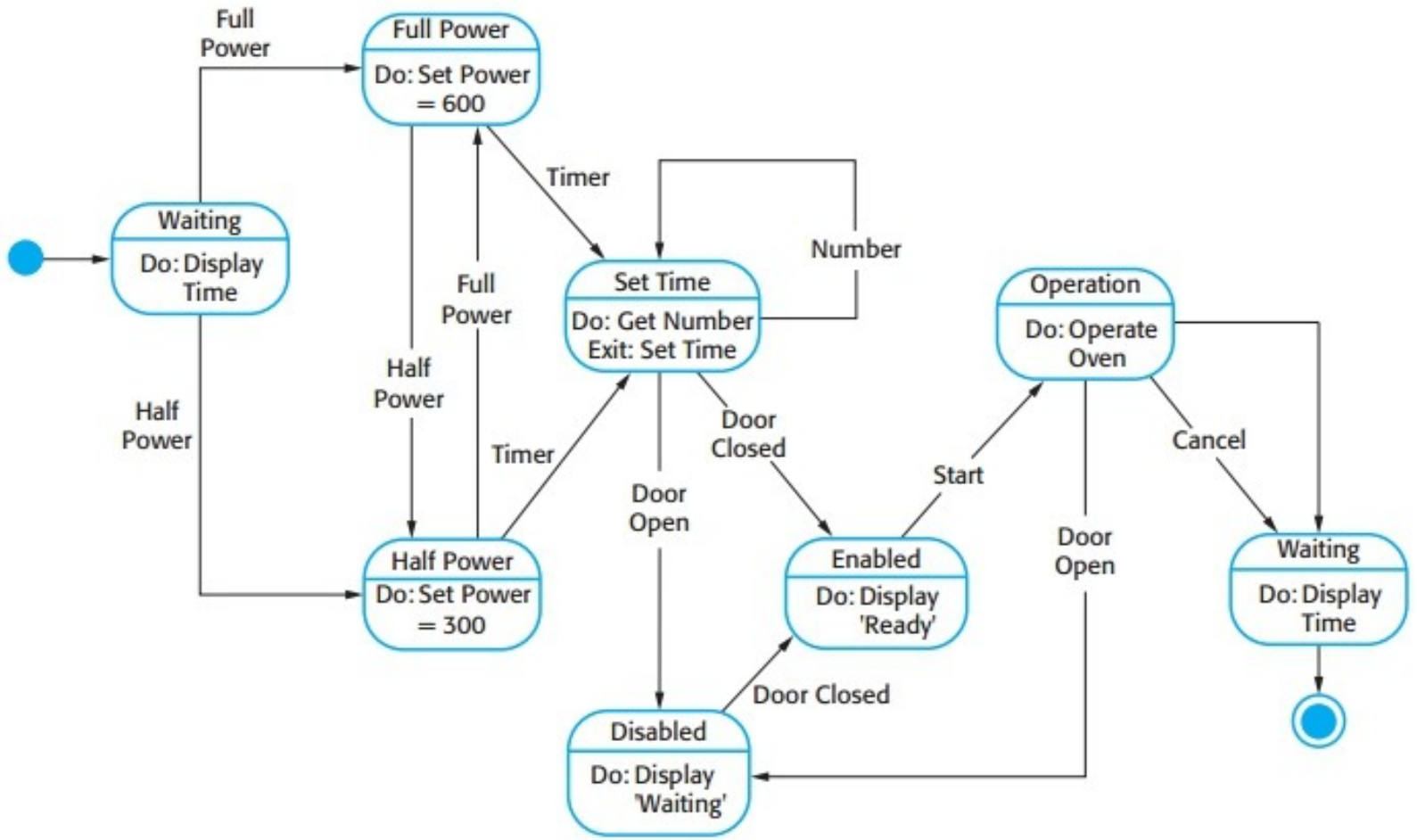
- **Modelado dirigido por datos (data-driven)**
  - Muestran la secuencia de acciones involucradas en procesar los datos de entrada y generar una salida asociada.
  - Los modelos de flujo de datos permiten realizar el seguimiento y documentar cómo los datos asociados a un proceso particular se mueven a través del sistema.
  - UML no soporta de forma natural los modelos de flujo de datos pero pueden utilizarse diagramas de secuencia.

Modelos de comportamiento

- **Modelado dirigido por eventos (event-driven)**
  - Muestran cómo un sistema responde a eventos externos e internos.
  - Se basan en la suposición de que un sistema tiene un número finito de estados y que eventos (estímulos) pueden causar una transacción de un estado a otro.
  - En UML se pueden utilizar diagramas de estado.

Modelos de comportamiento

- **Diagrama de estado - ejemplo**



Ingeniería dirigida por modelos (MDE)

- Es un enfoque para el desarrollo de software donde los modelos (y no los programas) son los principales resultados del proceso de desarrollo.
- Los programas que se ejecutan en una plataforma de hardware/software se generan automáticamente a partir de los modelos.
- La MDE se encuentra todavía en una etapa temprana de desarrollo y no está claro si tendrá o no un efecto significativo en la práctica de la ingeniería de software.

Otros modelos para requisitos - Wiegers

- Los modelos visuales de requisitos ayudan a identificar requisitos faltantes, extraños o inconsistentes.
- Los modelos sirven tanto para elaborar y explorar requisitos así como para diseñar soluciones de software.
- No hay que asumir que los clientes saben cómo interpretar los modelos de análisis.
- En general no es posible modelar todo el sistema, así que lo recomendable es enfocarse en las porciones del sistema más riesgosas.

Diagrama de flujo de datos

- Provee una visión de cómo se mueven los datos a través de un sistema.

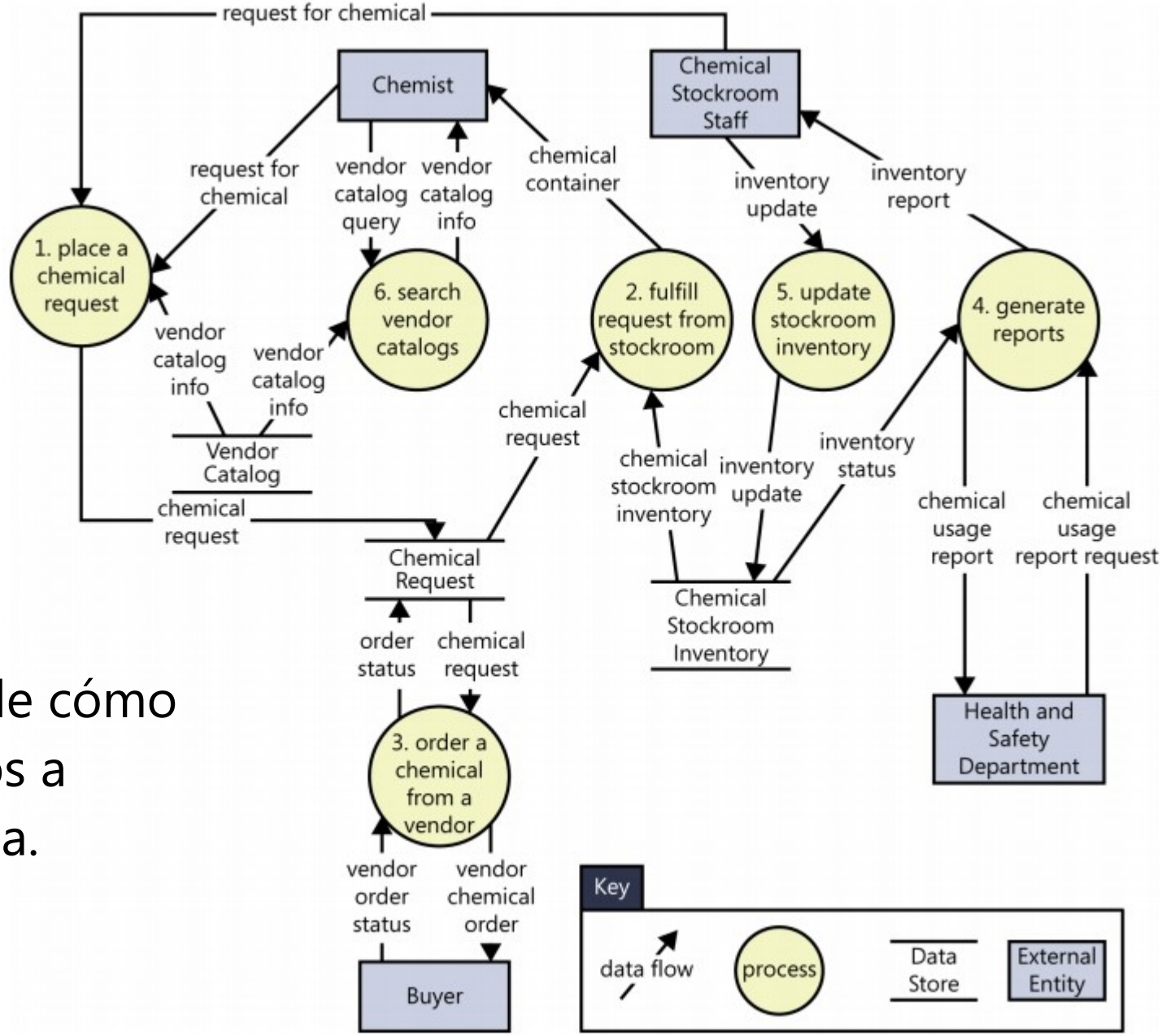


Diagrama de andariveles

- Proveen una manera de representar los pasos involucrados en un proceso o las operaciones de un nuevo sistema de software.
- Los carriles representan actores u otros sistemas que ejecutan un paso del proceso.
- Es uno de los modelos más simples de entender por los clientes.

Diagrama de andariveles

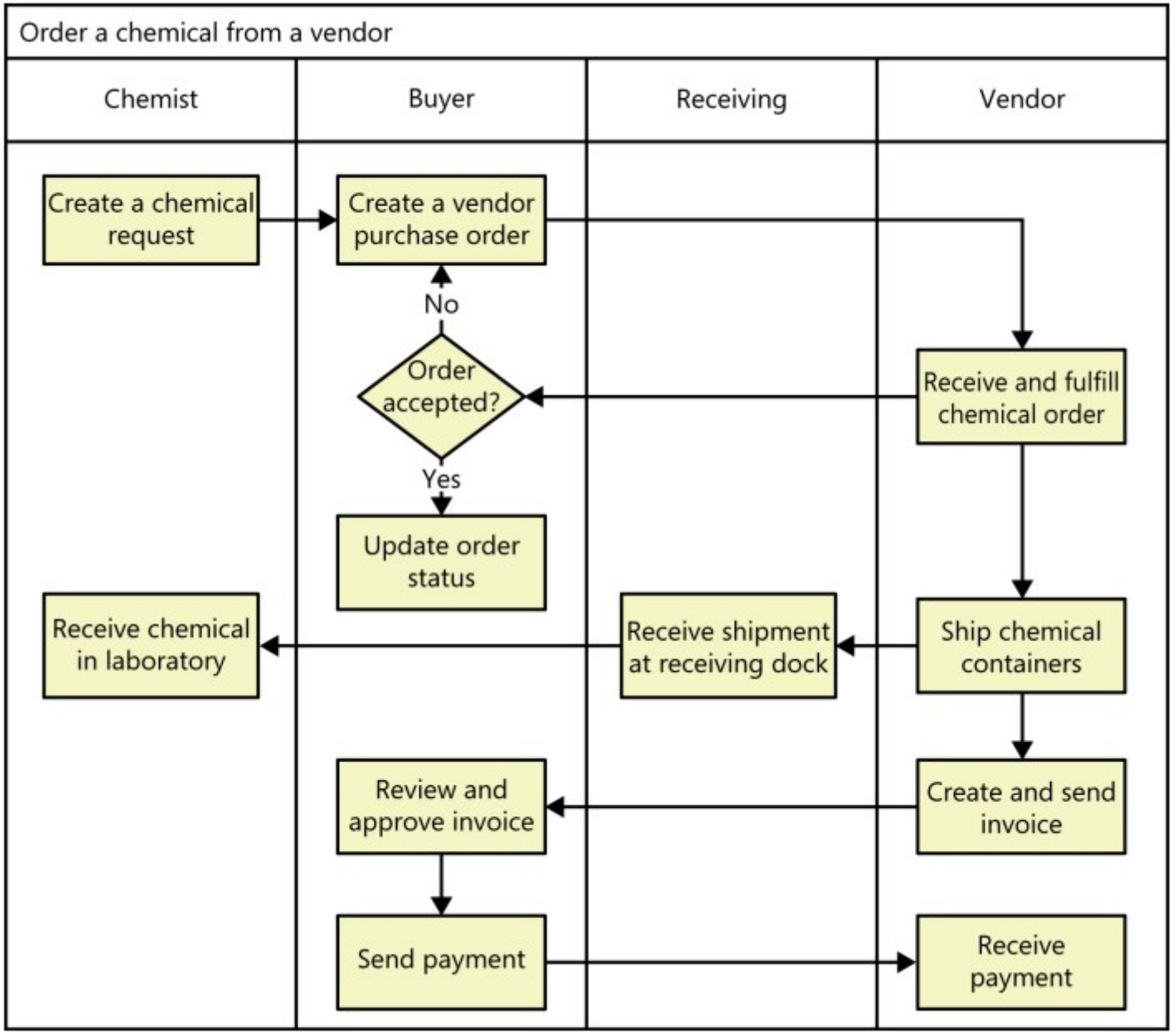


Diagrama de estados y transiciones y Tablas de estados

- Proveen una representación concisa, completa y no ambigua de los estados de un objeto o sistema.
- Los diagramas de estado contienen:
  - Los posibles estados se representan como rectángulos.
  - Las transiciones o los cambios de estado permitidos se representan como flechas entre dos rectángulos.
  - Los eventos o condiciones que causan los cambios de estado se muestran como etiquetas de texto sobre las flechas de transición.

Diagrama de estados y transiciones y Tablas de estados

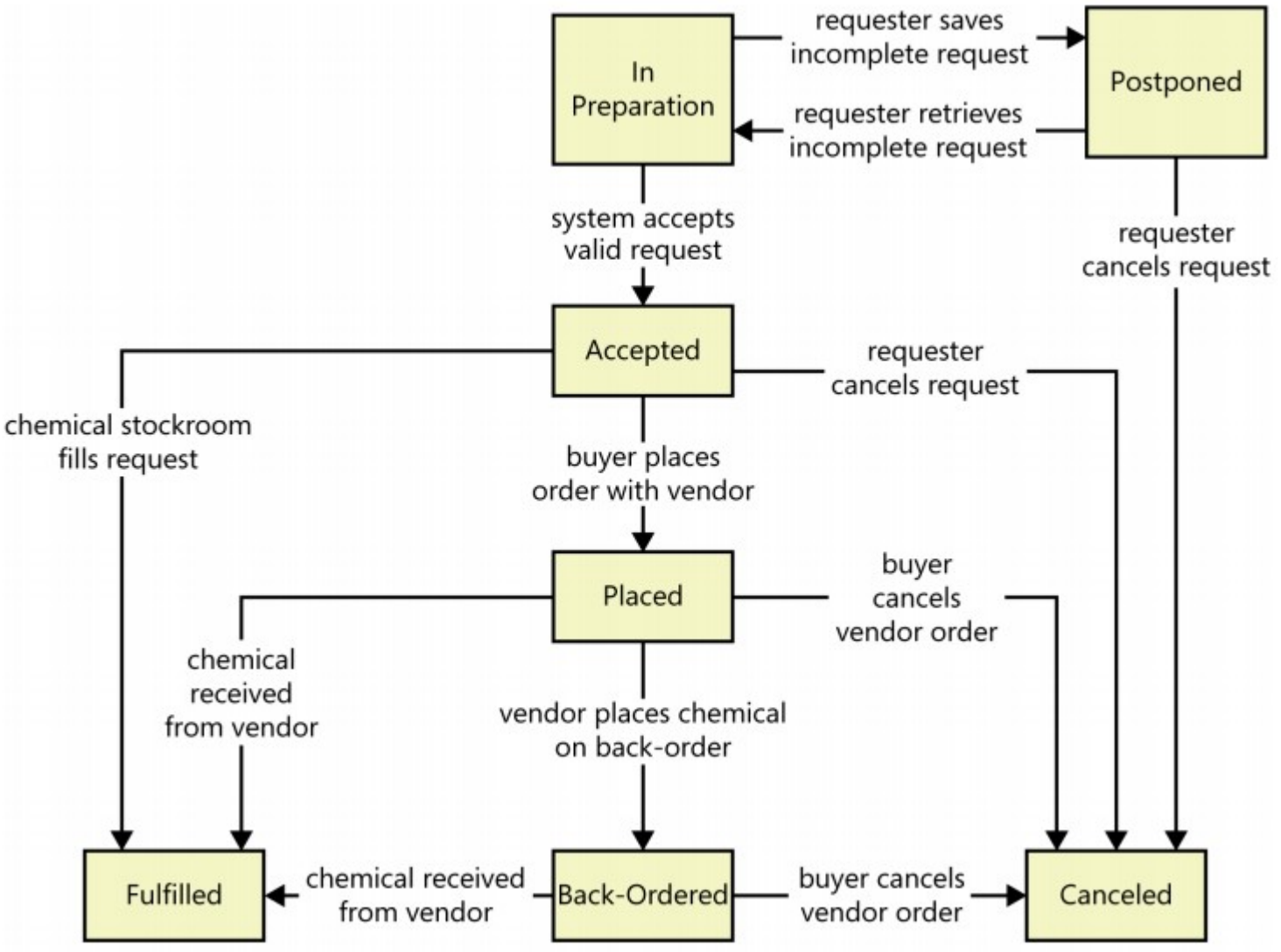


Diagrama de estados y transiciones y Tablas de estados

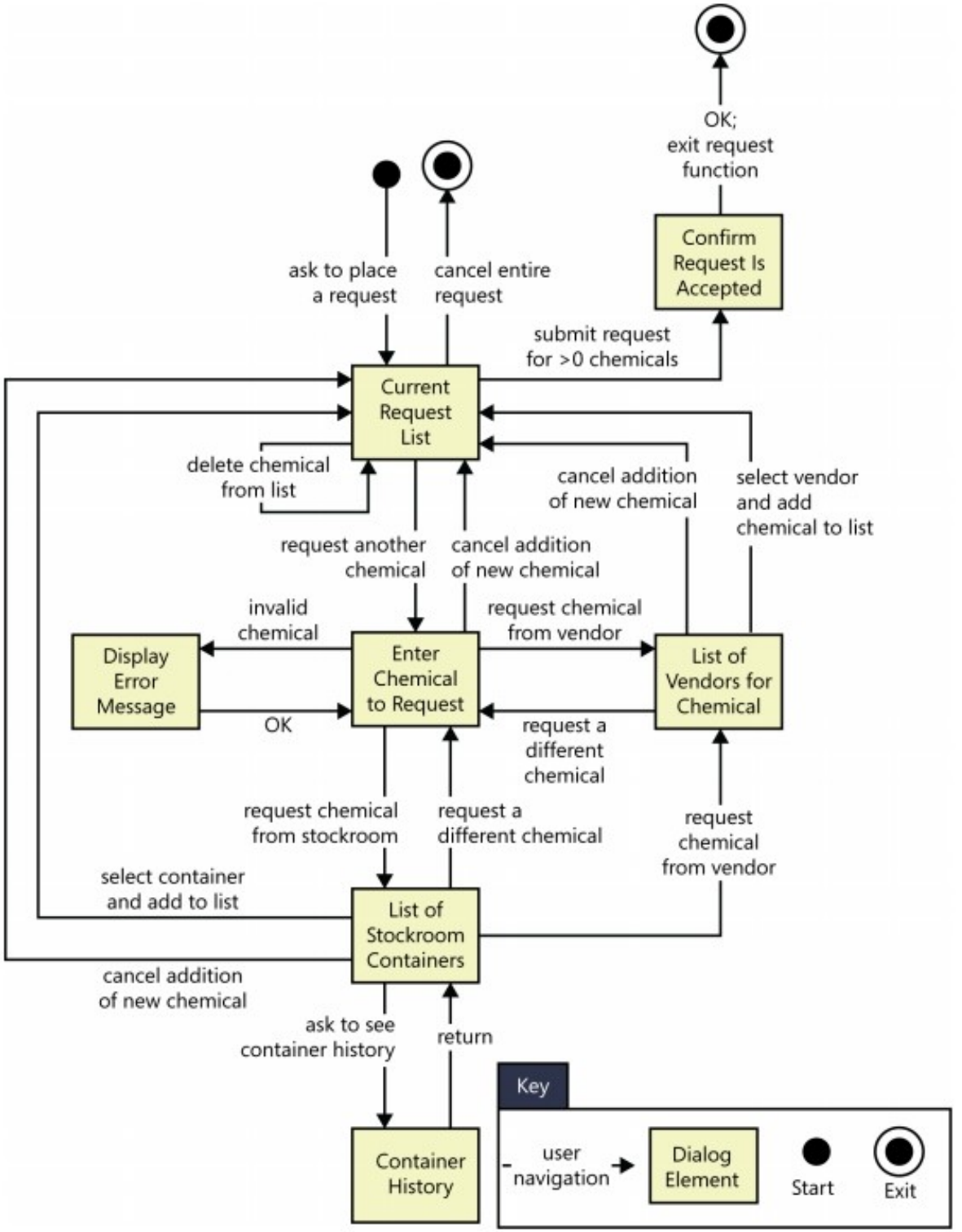
- Las tablas de estados muestran todas las posibles transiciones entre los estados en forma de una matriz.

	In preparation	Postponed	Accepted	Placed	Back-Ordered	Fulfilled	Canceled
In Preparation	no	user saves incomplete request	system accepts valid request	no	no	no	no
Postponed	user retrieves incomplete request	no	no	no	no	no	no
Accepted	no	no	no	buyer places order with vendor	no	chemical stockroom fills request	requester cancels request
Placed	no	no	no	no	vendor places chemical on back-order	chemical received from vendor	buyer cancels vendor order
Back-Ordered	no	no	no	no	no	chemical received from vendor	buyer cancels vendor order
Fulfilled	no	no	no	no	no	no	no
Canceled	no	no	no	no	no	no	no

Mapa de diálogo

- Un mapa de diálogo representa el diseño de la interfaz de usuario con un nivel alto de abstracción.
- La técnica permite modelar la interfaz de usuario en la forma de un diagrama de estados y transiciones.

Mapa de diálogo

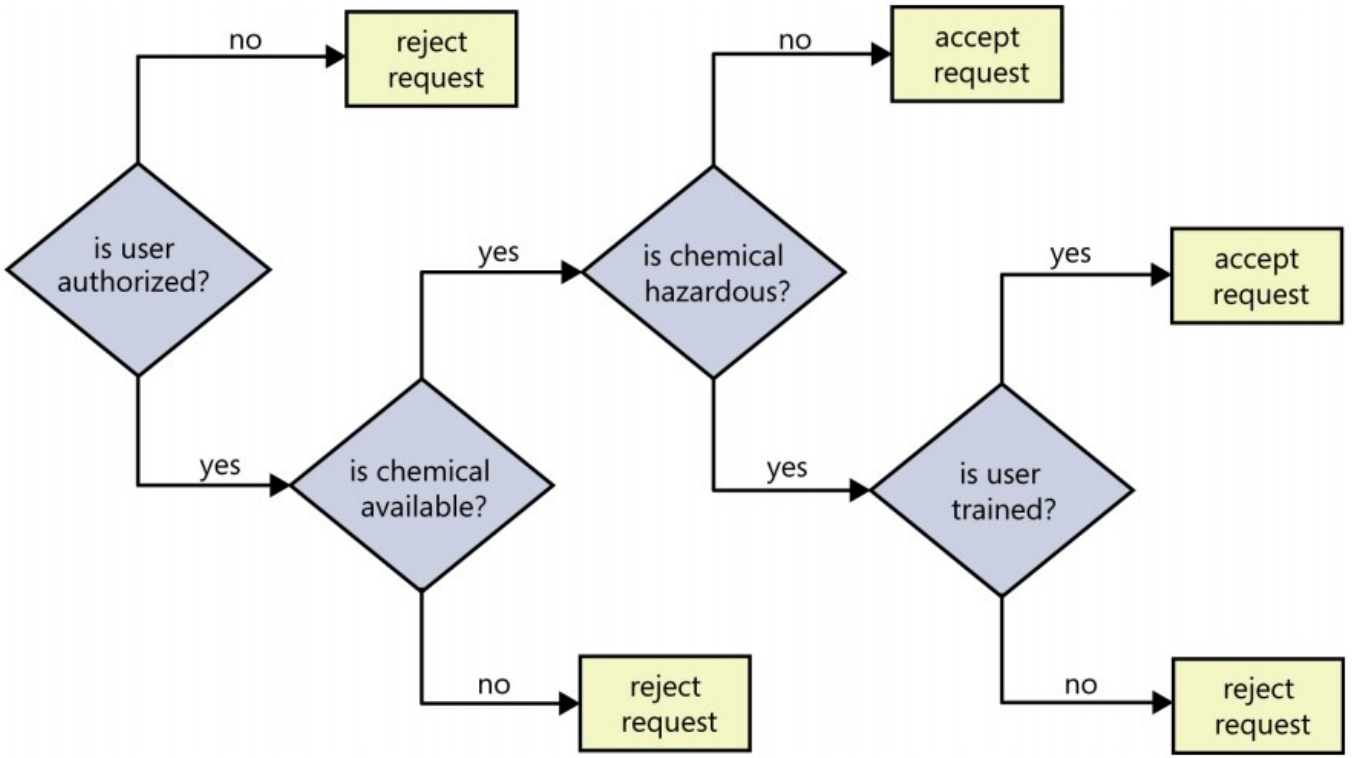


Tablas y árboles de decisión

- Las tablas y árboles de decisión son dos técnicas alternativas para representar qué debería hacer el sistema ante decisiones o lógica compleja.
- Una tabla de decisión lista los valores para todos los factores que influencian el comportamiento del sistema e indican la acción esperada del sistema en respuesta a cada combinación de factores.

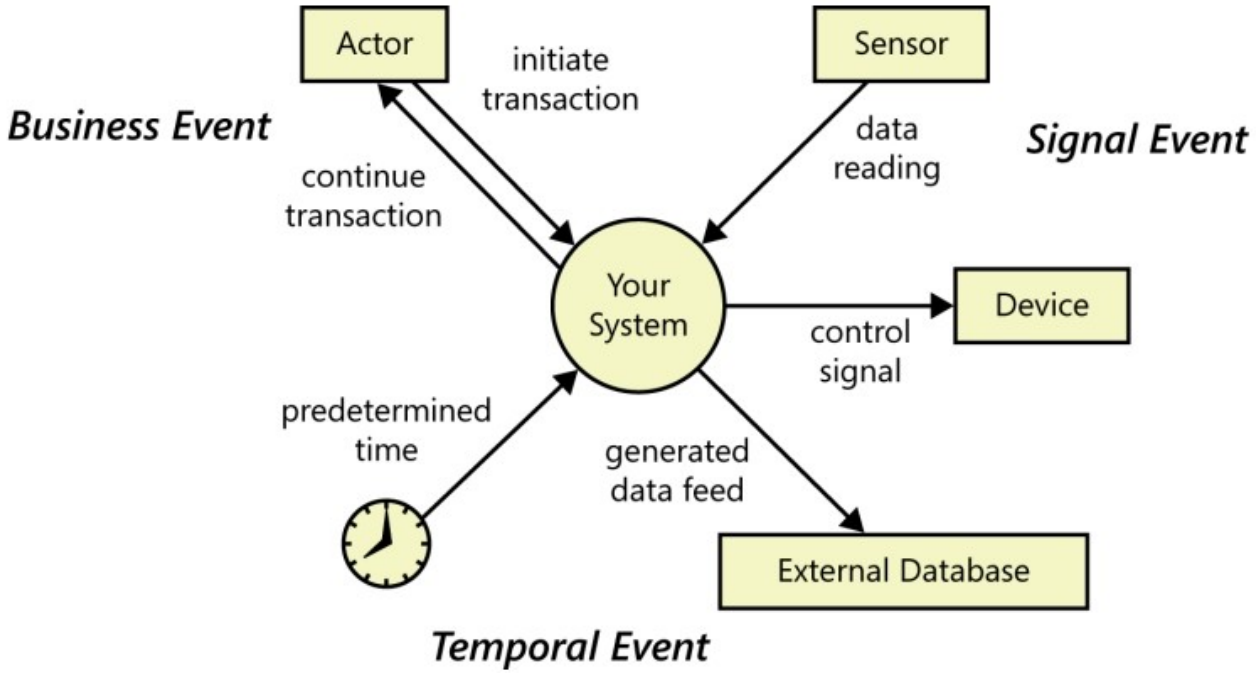
Tablas y árboles de decisión

Requirement Number					
Condition	**1**	**2**	**3**	**4**	**5**
User is authorized	F	T	T	T	T
Chemical is available	—	F	T	T	T
Chemical is hazardous	—	—	F	T	T
Requester is trained	—	—	—	F	T
Action					
Accept request			X		X
Reject request	X	X		X	



Tablas de evento-respuesta

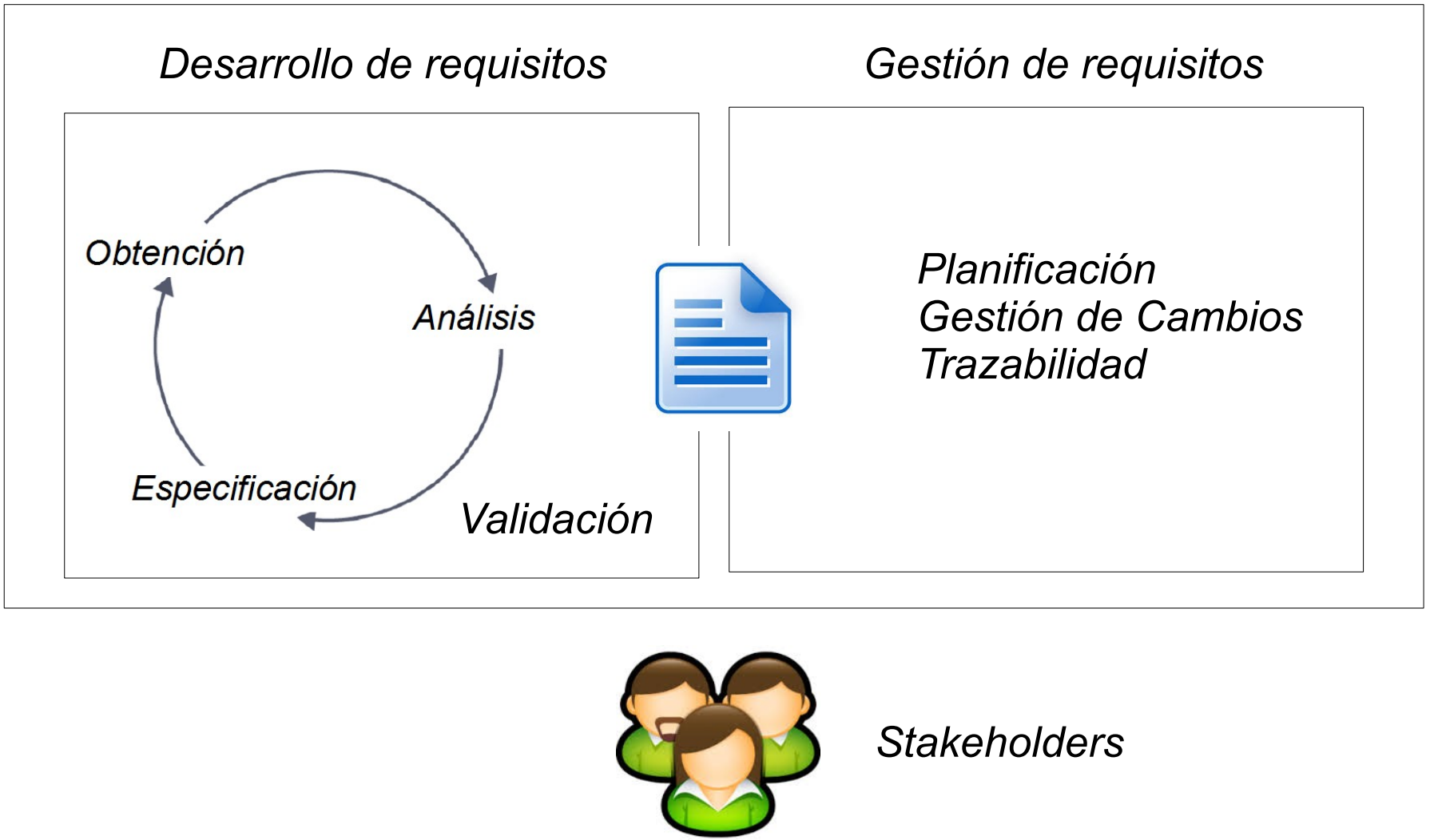
- Permiten representar las respuestas del sistema a los posibles eventos que puedan ocurrir.
- Un evento es un cambio o actividad que tiene lugar en el ambiente del usuario y que estimula una respuesta del sistema.



Tablas de evento-respuesta

ID	Event	System state	System response
1	Set wiper control to low speed	Wiper off, on high speed, or on intermittent	Set wiper motor to low speed
2	Set wiper control to high speed	Wiper off, on low speed, or on intermittent	Set wiper motor to high speed
3	Set wiper control to off	Wiper on high speed, low speed, or intermittent	1. Complete current wipe cycle 2. Turn wiper motor off
4	Set wiper control to intermittent	Wiper off	1. Perform one wipe cycle 2. Read wipe time interval setting 3. Initialize wipe timer
5	Set wiper control to intermittent	Wiper on low speed or on high speed	1. Complete current wipe cycle 2. Read wipe time interval setting 3. Initialize wipe timer
6	Wipe time interval has passed since completing last cycle	Wiper on intermittent	Perform one wipe cycle at low speed setting
7	Change intermittent wiper interval	Wiper on intermittent	1. Read wipe time interval setting 2. Initialize wipe timer
8	Change intermittent wiper interval	Wiper off, on high speed, or on low speed	No response
9	Immediate wipe signal received	Wiper off	Perform one low-speed wipe cycle

Actividades de la ingeniería de requisitos



Temario

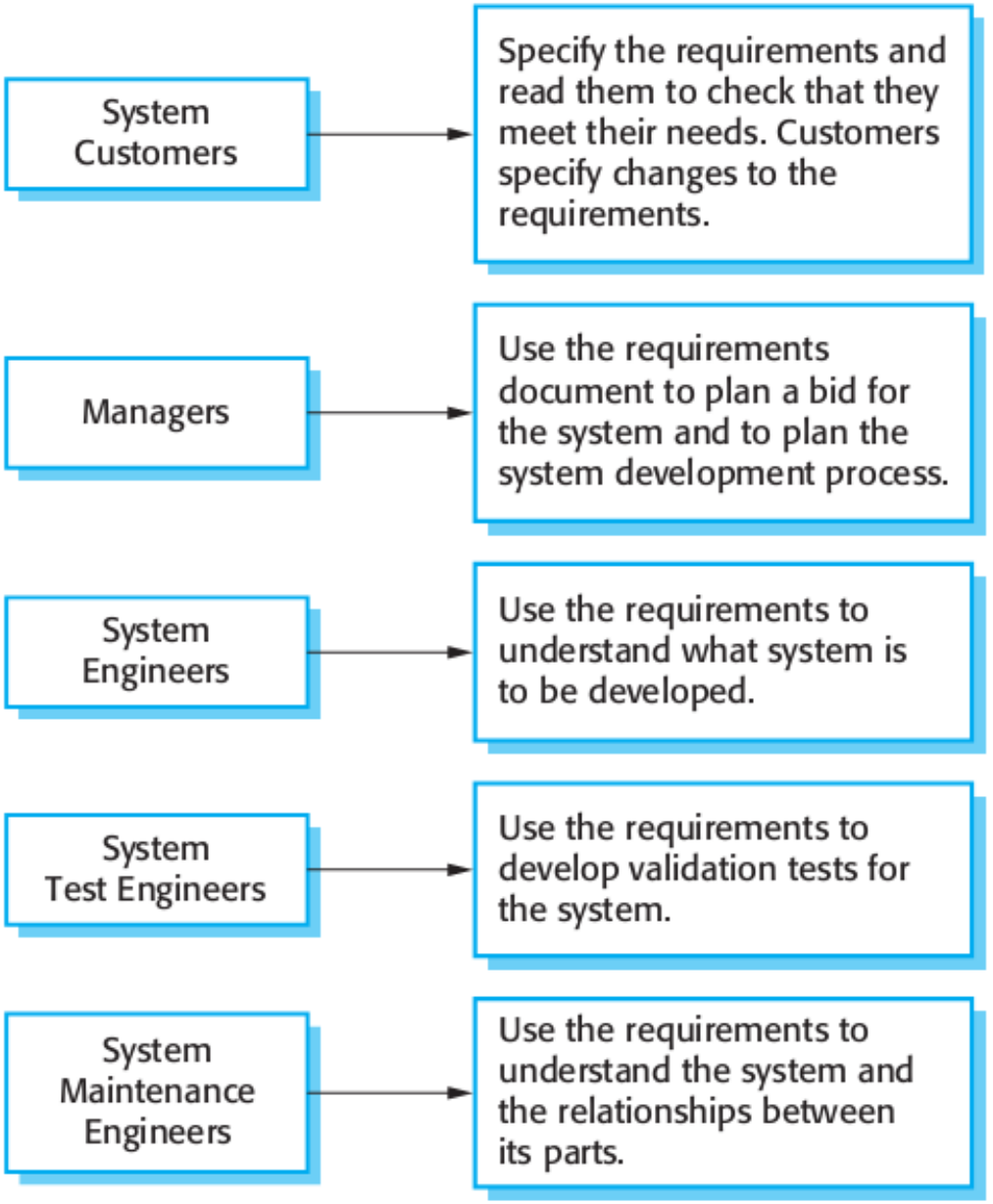
- Documento de requisitos
- Priorización de requisitos
- Validación de requisitos
- Gestión de requisitos

Documento de requisitos de software

- El documento de requisitos de software (SRS) es una declaración oficial de lo qué se requiere de los desarrolladores del sistema.
- Puede incluir una definición de los requisitos del usuario y una especificación de los requisitos del sistema.
- No es un documento de diseño. Se debe indicar lo que el sistema debe hacer y no la forma en la que lo debe hacer.

- Metodologías ágiles prefieren, en lugar de construir un documento formal, recolectar requisitos de forma separada. Por ejemplo: las historias de usuario en Extreme Programming (XP).

Usuarios de un documento de requisitos



→ COMPROMISO

Variabilidad del documento de requisitos

- La información del documento de requisitos depende del tipo de sistema y el enfoque de desarrollo usado.
- Los sistemas desarrollados de forma incremental tienen típicamente menos detalles en el documento de requisitos.
- Se han diseñado estándares para documentos de requisitos ej: estándar de IEEE 29148:2011 (antes el IEEE 830:1998).

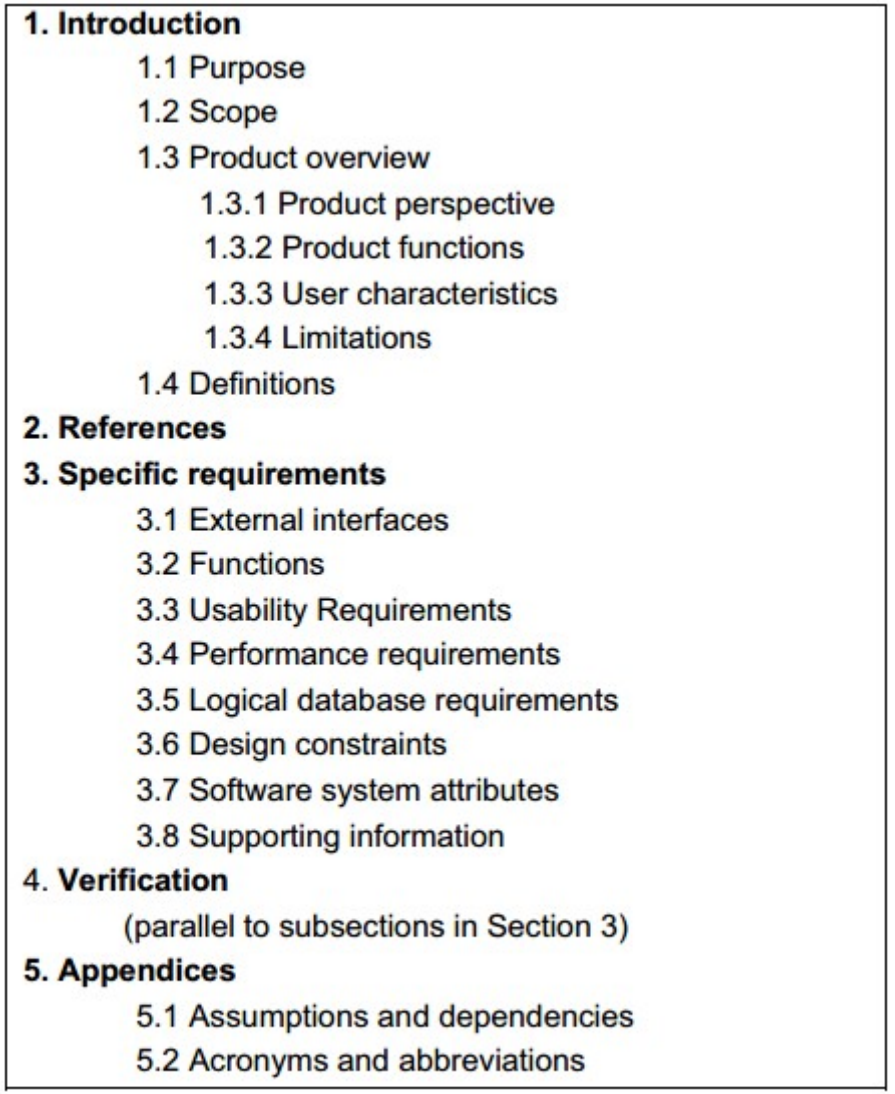
Estructura de un documento de requisitos

Capítulo	Descripción
Prefacio	Definir los lectores esperados y describir el historial de versiones, incluyendo una justificación para la creación de una nueva versión y un resumen de los cambios realizados en cada versión.
Introducción	Describir las necesidades y las funciones del sistema y explicar como va a funcionar con otros sistemas. También debe describir como el sistema se ajusta a los objetivos del negocio o a los planes estratégicos de la organización.
Glosario	Definir los términos técnicos usados en el documento. No se deben hacer suposiciones acerca de la experiencia o de los conocimientos del lector.
Definición de requisitos de usuario	Aquí, se describen los servicios prestados para el usuario. En esta sección también deben ser descritos los requisitos no funcionales del sistema. Esta descripción puede utilizar el lenguaje natural, diagramas u otra notación entendible por los clientes. Las normas de producto y proceso que deben seguir deben ser especificados.
Arquitectura del sistema	Este capitulo debe presentar una visión de alto nivel de la arquitectura del sistema, mostrando la distribución de las funciones sobre los módulos del sistema.

Estructura de un documento de requisitos

Capitulo	Descripción
Especificación de requisitos del sistema	Describir con más detalle los requisitos funcionales y no funcionales, así como interfaces con otros sistemas.
Los modelos del sistema	Aquí se puede incluir modelos gráficos del sistema que muestren las relaciones entre los componentes del sistema y la relación con su entorno.
Evolución del sistema	Describir los supuestos fundamentales en los cuales esta basado el sistema y cualquier cambio previsto debido a la evolución del hardware, cambios en las necesidades de los usuarios, etc. Esta sección es útil para los diseñadores del sistema ya que puede ayudar a evitar que las decisiones de diseño restrinjan posibles cambios futuros.
Apéndices	Debería proporcionar información detallada y especifica relacionada con la aplicación que se esta desarrollando, por ejemplo hardware y descripción de la base de datos.
Índice	Se pueden incluir varios índices. Entre ellos puede ser un índice alfabético, índice de diagramas, índice de funciones, etc.

Ejemplo de un SRS – IEEE 29148:2011



1. Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Product overview
  - 1.3.1 Product perspective
  - 1.3.2 Product functions
  - 1.3.3 User characteristics
  - 1.3.4 Limitations
- 1.4 Definitions

2. References

3. Specific requirements
- 3.1 External interfaces
- 3.2 Functions
- 3.3 Usability Requirements
- 3.4 Performance requirements
- 3.5 Logical database requirements
- 3.6 Design constraints
- 3.7 Software system attributes
- 3.8 Supporting information

4. **Verification**

(parallel to subsections in Section 3)

5. Appendices
- 5.1 Assumptions and dependencies
- 5.2 Acronyms and abbreviations

Figure 8 — Example SRS Outline

Documento de requisitos - Consideraciones

- Registrar los requisitos en términos del cliente.
- Características de una buena especificación:
  - Correcta / Válida
  - No ambigua
  - Completa
  - Consistente internamente
  - Agrupados y ordenados por importancia y/o estabilidad
  - Verificable
  - Modificable
  - Trazable
  - Factibles
  - Entendible

Priorización de requisitos

- Todo proyecto que tenga recursos limitados necesita definir prioridades en los requisitos que va a incluir en el producto de software.
- La priorización ayuda a entregar el mayor valor de negocio tan rápido como sea posible con las restricciones existentes.

- A veces los clientes no quieren priorizar requisitos pensando que no se hará lo que sea de prioridad baja.
- Los programadores por su lado no suelen priorizar porque eso daría la impresión de que no son capaces de realizar todos los requisitos.

Priorización de requisitos

- Se debe elegir un nivel de abstracción apropiado (casos de uso, historias de usuario, requisitos funcionales o quizás flujos dentro de un caso de uso).

- Se deben tener en cuenta:
  - Las necesidades de los clientes.
  - Las importancia relativa de los requisitos para los clientes.
  - Las relaciones de precedencia entre los requisitos.
  - Requisitos que deban ser implementados en grupo.
  - El costo de satisfacer cada requisito.

Priorización de requisitos

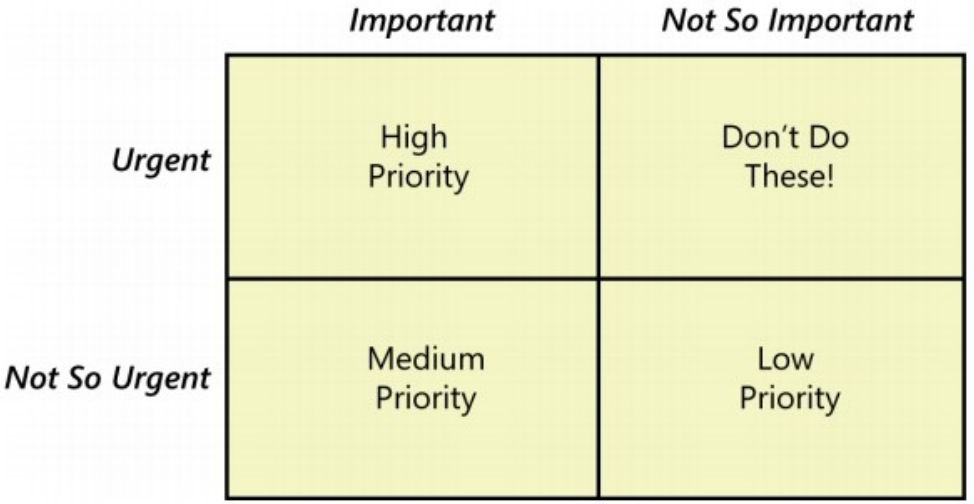
- Un estudio muestra que cerca de dos tercios de las características de los sistemas de software se usan raramente o nunca (The Standish Group 2009).

- Importante: sí terminamos el proceso de priorización con todos los requisitos con la misma prioridad entonces no priorizamos nada.

- Hay que evitar dos casos de priorización:
  - Por decibeles — los que gritan más definen las prioridades.
  - Por amenazas — los que tienen más poder definen las prioridades.

Priorización de requisitos

- Algunas técnicas posibles:
  - Adentro y afuera — se nombra cada requisito y se decide sí queda en el alcance o no.
  - Comparación por pares y ranking — se hace un ranking general de todos los requisitos mediante comparación de a pares.
  - Escala de tres niveles — se utilizan dos dimensiones.



Priorización de requisitos

- Algunas técnicas posibles:
  - 4 niveles (MoSCoW) — debe, debería, podría, no.
  - $100 — asignar dinero disponible y luego “invertirlo” en los requisitos a priorizar.
  - Priorización basado en valores, costos y riesgos.

Relative weights		**2**	**1**			**1**		**0.5**		
Feature		Relative benefit	Relative penalty	Total value	Value %	Relative cost	Cost %	Relative risk	Risk %	Priority
1.	Print a material safety data sheet.	2	4	8	5.2	1	2.7	1	3.0	1.22
10.	Import chemical structures from structure drawing tools.	7	4	18	11.6	9	24.3	7	21.2	0.33
	Totals	**53**	**49**	**155**	**100.0**	**37**	**100.0**	**33**	**100.0**	

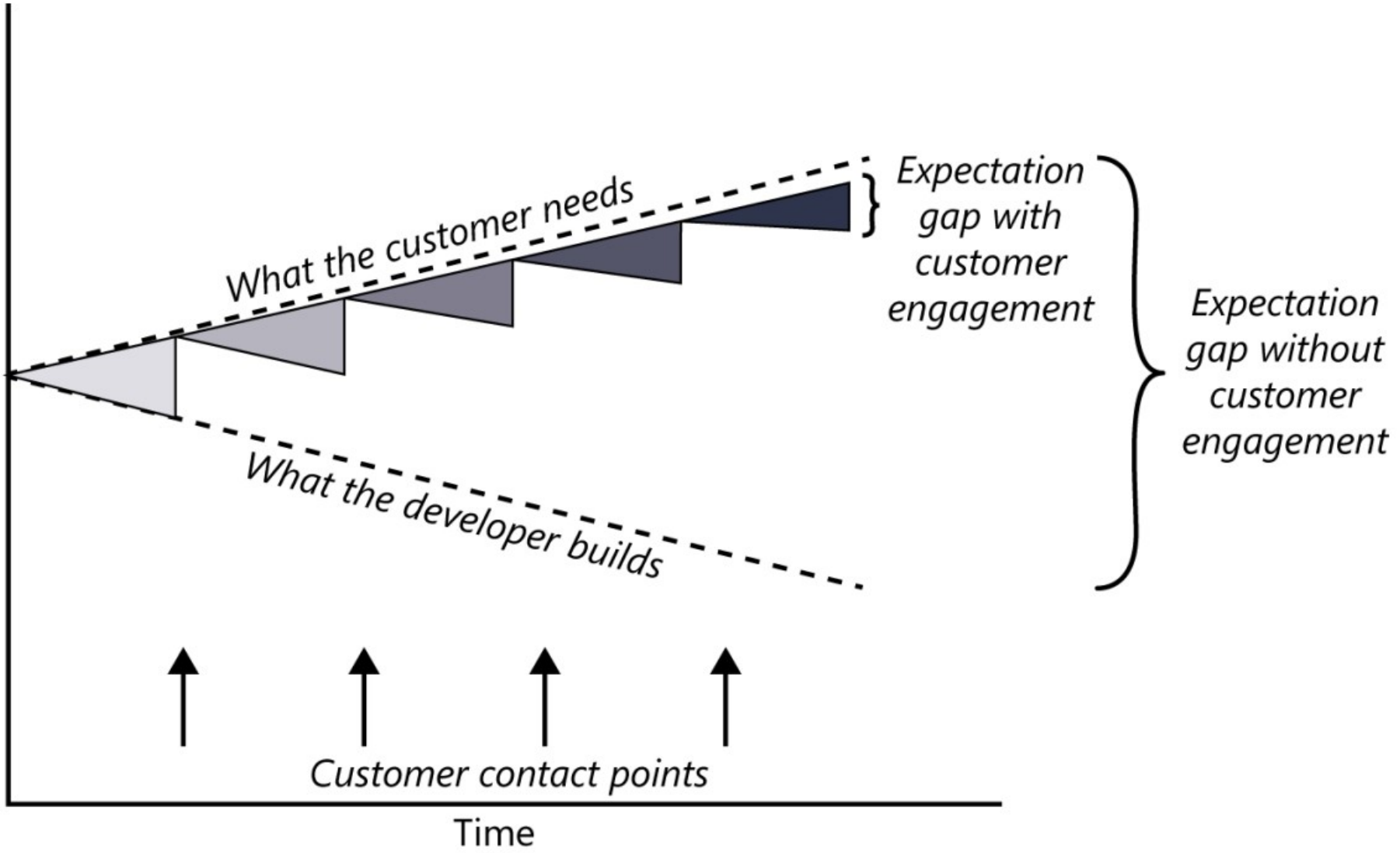
Validación de requisitos

- Proceso por el cual se determina si los requisitos relevados son consistentes con las necesidades del cliente.

- *Verificación — determina si el producto de alguna actividad de desarrollo cumple los requisitos (hacer las cosas bien).*
- *Validación — evalúa si un producto satisface las necesidades del cliente (hacer la cosa correcta).*

- Proceso:
  - Planificar quién (qué stakeholder) va a validar qué (artefacto) cómo (técnica).
  - Ejecutar
  - Registrar – Reporte de validación / Firma

Expectativas del cliente



Chequeo de requisitos

- **Validez**— ¿El sistema provee las funciones que mejor soportan las necesidades del cliente?
- **Consistencia**— ¿Hay algún requisito en conflicto?
- **Completitud**— ¿Están incluidas todas las funciones requeridas por el cliente?
- **Realismo**— ¿Los requisitos pueden ser implementados con el presupuesto y la tecnología disponible?
- **Verificabilidad**— ¿Los requisitos pueden ser chequeados?

Validación de requisitos no funcionales

- Son difíciles de validar.
- Se deben expresar de manera cuantitativa utilizando métricas que se puedan probar de forma objetiva (esto es ideal).

Propiedad	Medida
Rapidez	Transacciones por seg
Tamaño	KB
Fiabilidad	Tiempo promedio entre fallas
Portabilidad	Número de sistemas, especificar
Facilidad de uso	Tiempo de capacitación

- Para los usuarios es difícil especificarlos en forma cuantitativa.

Técnicas de validación de requisitos

- **Revisión de requisitos**
  - Análisis manual y sistemático de los requisitos.
- **Prototipado**
  - Uso de un modelo ejecutable del sistema para chequear los requisitos.
- **Generación de casos de prueba**
  - Desarrollo de pruebas para los requisitos que pueden ser testeados.

¿Cómo validar requisitos?

- Manuales
  - Lectura por parte del cliente.
  - Recorridas. Útiles con muchos stakeholders que no lo leerían de otra manera.
  - Entrevistas.
  - Chequeo manual de referencias cruzadas.
  - Instancias de validación formal:
    - Revisiones - Stakeholders revisan por separado y se reúnen para discutir problemas.
    - Inspecciones formales – roles y reglas.
  - Listas de comprobación.
  - Escenarios.
  - Generación de Casos de Prueba.

- Automatizadas
  - Chequeo automático de referencias cruzadas,
  - Ejecución de Modelos para verificar funciones y relaciones.
  - Construcción de Prototipos.
  - Simulaciones.

Ej. checklist de defectos en requisitos (wiegers 339)

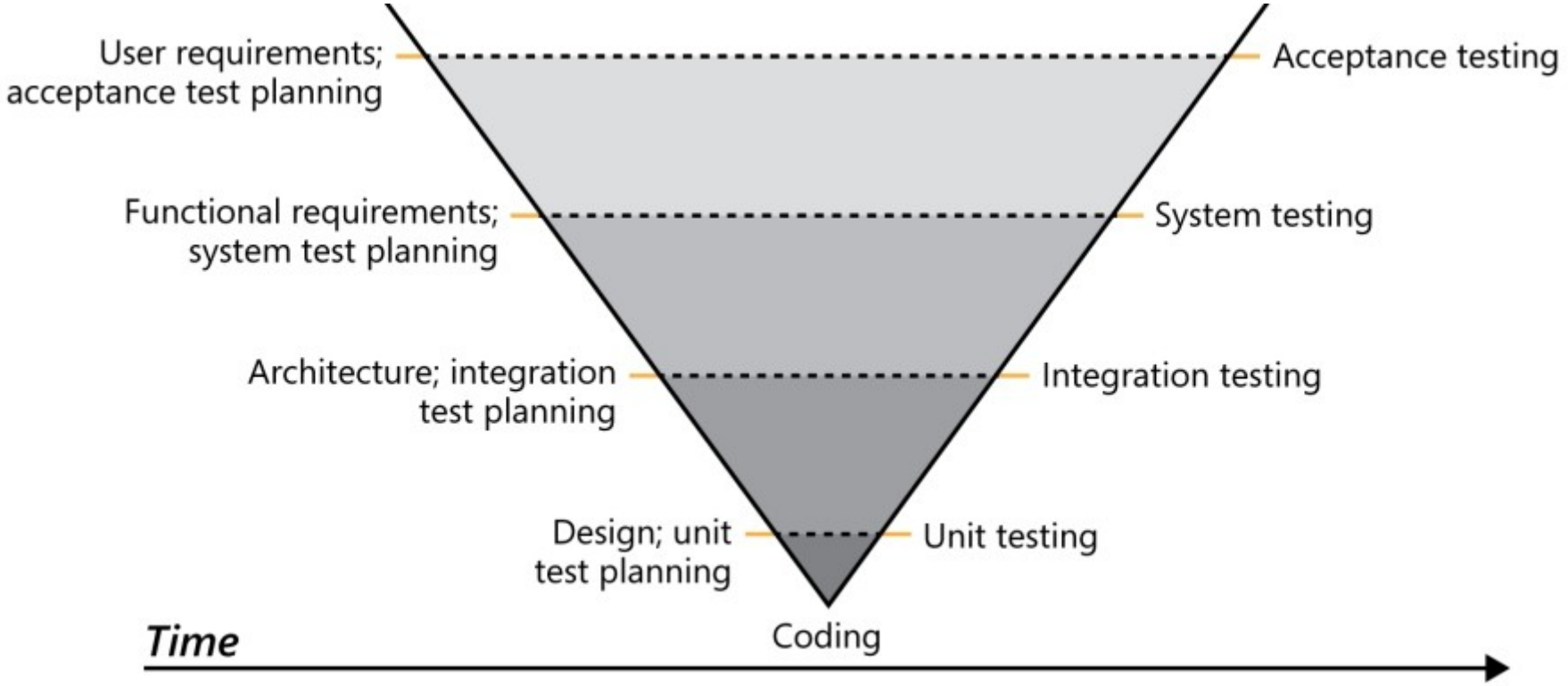
Completeness

- ❑ Do the requirements address all known customer or system needs?
- ❑ Is any needed information missing? If so, is it identified as TBD?
- ❑ Have algorithms intrinsic to the functional requirements been defined?
- ❑ Are all external hardware, software, and communication interfaces defined?
- ❑ Is the expected behavior documented for all anticipated error conditions?
- ❑ Do the requirements provide an adequate basis for design and test?
- ❑ Is the implementation priority of each requirement included?
- ❑ Is each requirement in scope for the project, release, or iteration?

Correctness

- ❑ Do any requirements conflict with or duplicate other requirements?
- ❑ Is each requirement written in clear, concise, unambiguous, grammatically correct language?
- ❑ Is each requirement verifiable by testing, demonstration, review, or analysis?
- ❑ Are any specified error messages clear and meaningful?
- ❑ Are all requirements actually requirements, not solutions or constraints?
- ❑ Are the requirements technically feasible and implementable within known constraints?

Probando los requisitos



Validar utilizando criterios de aceptación

- El cliente es el que tendrá la opinión final sobre un producto. La idea es entonces establecer, de forma complementaria a la definición de los requisitos, los criterios con los cuales se aceptarán.

- Los criterios de aceptación no son pruebas funcionales o unitarias; sino condiciones que deben cumplir el sistema. Las pruebas son más completas y prueba todos los flujos funcionales, excepciones, límites, etc.

Criterios de aceptación - Ejemplos

Requerimientos	Criterios de Aceptación
Registrar Oferta Laboral	Permite que una empresa pueda registrar una oferta laboral basándose en las capacidades y competencias que maneja la carrera de computación.
Toma de tests por alumnos de psicología a personas de otras instituciones	Permite convocar a un alumno que, luego de ser evaluado por la empresa, cumple con todos los requisitos que la oferta laboral establece.

Gestión de requisitos

- Gestionar los cambios de los requisitos durante el proceso de ingeniería de requisitos y el desarrollo del sistema.
- Nuevos requisitos surgen cuando un sistema está siendo desarrollado y después de haber entrado en uso.
- Es necesario mantener un rastreo (*trazabilidad*) de los requisitos y mantener sus referencias para facilitar el análisis de impacto ante posibles cambios.
- Se recomienda establecer un proceso formal de control de cambios.

Cambios en los requisitos

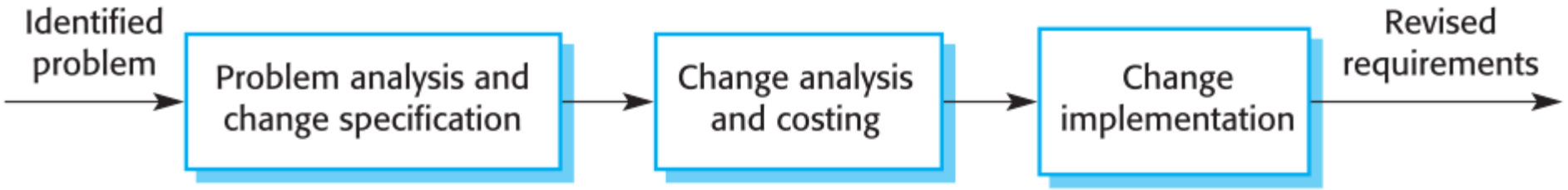
- El entorno empresarial y técnico del sistema siempre cambia después de la instalación.
  - Nuevo hardware, nuevas interfaces con otros sistemas, cambian las prioridades del negocio, nuevas legislaciones, etc.
- Las personas que pagan por un sistema y los usuarios de ese sistema casi nunca son las mismas personas.
- Los grandes sistemas tienen una diversa comunidad de usuarios, algunos de los cuales tienen diferentes requisitos y pueden existir conflictos en las prioridades.

Planificación de la gestión de requisitos

- Establece el detalle del nivel de gestión requisitos que es requerido.
- Decisiones de gestión de requisitos:
  - **Identificación de requisitos—** Cada requisito debe ser identificado de modo que pueda hacerse una referencia cruzada con otros.
  - **Proceso de gestión de cambios—** Es el conjunto de actividades que evalúan el impacto y el costo de los cambios.
  - **Políticas de trazabilidad—** Estas políticas definen cómo registrar las relaciones entre los requisitos y el sistema diseñado.
  - **Soporte de herramientas—** Las herramientas que utilizarán.

Gestión de cambios en los requisitos

- Decidir sí un cambio en los requisitos debe ser aceptado.
- Problemas de análisis y cambios en la especificación
  - Se analiza para chequear si es valido.
- Cambiar el análisis y calculo de costos
  - se evalúa el efecto (info de trazabilidad y el conocimiento general del sistema). se toma la decisión de sí se debe proceder o no.
- Implementación del cambio



Cambios de requisitos en ágiles

- Cuando un usuario propone un cambio de requisitos, este cambio no pasa por un proceso formal de gestión de cambios.
- El usuario debe priorizar ese cambio y, si es de alta prioridad, debe decidir qué características del sistema que se planificaron para la próxima iteración se deben eliminar para que se implemente el cambio.
- Pero…. en los sistemas con múltiples partes interesadas, los cambios beneficiarán a algunas partes interesadas y no a otras.
- Sugerencia: autoridad independiente, que puede equilibrar las necesidades de todos los interesados.