Tema 5.2: Especificación de Requisitos

Ingeniería de Requisitos

Raquel Martínez España

Grado en Ingeniería Informática

Índice

1. Especificación de requisitos
2. El documento de requisitos
3. Especificación formal

Objetivos

- Comprender por qué las técnicas de especificación formal ayudan a descubrir problemas en los requisitos del sistema.
- Conocer el uso de técnicas algebraicas de especificación formal para definir las especificaciones de interfaz.
- Entender cómo las técnicas formales basadas en modelos formales se usan para especificar el comportamiento.

Índice

1. Especificación de requisitos
2. El documento de requisitos
3. Especificación formal
    1. Especificación algebraica
    2. Especificación basada en modelos

Especificación formal

Una especificación escrita y aprobada de acuerdo con algunos estándares establecidos.

Una especificación escrita en una notación formal, a menudo para ser usada en pruebas de corrección.

[IEEE Std. 610.12-1990 (IEEE Standard Glossary of Software Engineering Terminology)]

Especificación formal

- La especificación formal es una especificación expresada en un lenguaje cuyo vocabulario, sintaxis y semántica están formalmente definidas.
- Los lenguajes de especificación formal están basados en conceptos matemáticos (teoría de conjuntos, lógica y álgebra).
- La especificación formal es parte de una colección de técnicas conocidas como “métodos formales”. Los métodos formales incluyen:
  - Especificación formal
  - Análisis y prueba de la especificación
  - Verificación de programas

Especificación formal

El uso de métodos formales:

- Facilita encontrar errores en la especificación, permite producir una especificación no ambigua, facilita las pruebas y reduce su coste
- Introducir rigor e “ingeniería” en el desarrollo de sistemas
- Documentación de requisitos
- Comprensión de propiedades intrínsecas del software
- En general, facilita y ayuda en todo el ciclo de vida:
  - Diseño
  - Implementación (generación automática)
  - Test (verificación formal)
  - Mantenimiento

Especificación formal

Razones para que su uso no sea muy extendido:

- Éxito de la IS: el uso de métodos como métodos estructurados, gestión de configuración, ocultación de la información,... en el diseño y proceso de desarrollo software ha mejorado la calidad del software.
- Cambio del mercado: en los 80 la calidad del software se tomó como la clave del problema de la IS, pero la clave es time-to-market. Las técnicas para desarrollo rápido de software no se llevan bien con la especificación formal.
- Alcance limitado de los métodos formales. No se adaptan para especificar las interfaces de usuario y la interacción con el usuario.
- Escalabilidad limitada de los métodos formales. Se ha usado con éxito en sistema críticos relativamente pequeños, el problema crece al no haber herramientas de soporte.

Especificación formal

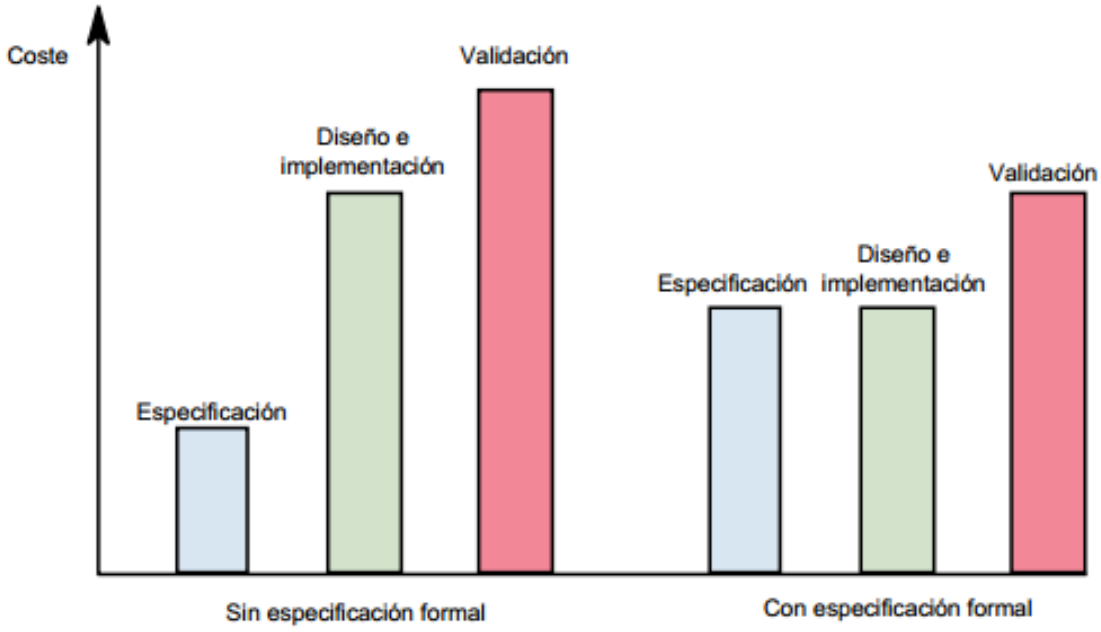
Algunas características:

- La utilización de métodos formales se justifica para seguridad y fiabilidad muy altas. Por ejemplo en sistemas de control de tráfico aéreo, de control médico, sistemas espaciales, ...
- Al estar basados en formalismos matemáticos es posible verificar la corrección, completitud y consistencia de los requisitos (de forma automática).
- Son difíciles de usar y comprender.
- Hace falta mucho esfuerzo para especificar formalmente.
- La intervención del cliente disminuye al avanzar la especificación.
- Exige un conocimiento y análisis muy detallado que revela inconsistencias y errores en la especificación informal de requisitos

Especificación formal

Coste:

- Especificación e implementación son comparables y los de validación se reducen de forma significativa.



Especificación formal

Existen básicamente dos enfoques para realizar especificación formal:

- **Algebraico**.
  - El sistema se especifica en términos de sus operaciones y las relaciones entre ellas.
- **Basado en modelo**.
  - El sistema se especifica en términos de un modelo de estado que se construye usando constructores matemáticos como conjuntos y sucesiones. Las operaciones se definen según cómo modifican el estado del sistema.

	Secuencial	Concurrente
Algebraico	Larch (Gutta, Horning et al. 1985; Gutta, Horning et al. 1993) OBJ (Futatsugi, Boguen et al. 1985)	Lotos (Bolognesi and Brinksma, 1987)
Basado en modelos	Z (Spivey, 1992) VDM (Jones, 1980) B (Woedsworth, 1996)	CSP (Hoare, 1985) Petri Nets (Peterson, 1981)

Índice

1. Especificación de requisitos
2. El documento de requisitos
3. Especificación formal
    1. Especificación algebraica
    2. Especificación basada en modelos

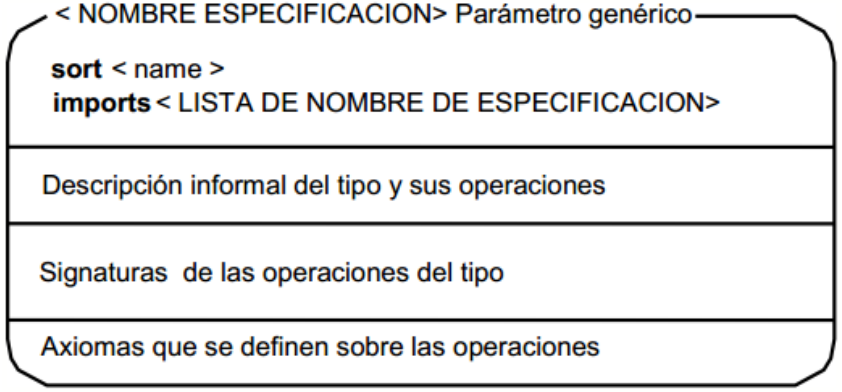
Especificación algebraica

- Se utiliza especialmente para la especificación de **interfaces de subsistemas**.
- La especificación de interfaces entre subsistemas permite el **desarrollo independiente** de los subsistemas.
- La especificación de las interfaces consiste en dar información sobre qué servicios están disponibles para otros subsistemas y cómo pueden se utilizados.
- Las interfaces se pueden definir como **tipos abstractos de datos** o **clases de objetos**.
- El TAD se define especificando las operaciones del tipo más que la representación del tipo.
- La especificación algebraica define el tipo abstracto de datos en términos de las relaciones entre las operaciones del tipo.

Especificación algebraica

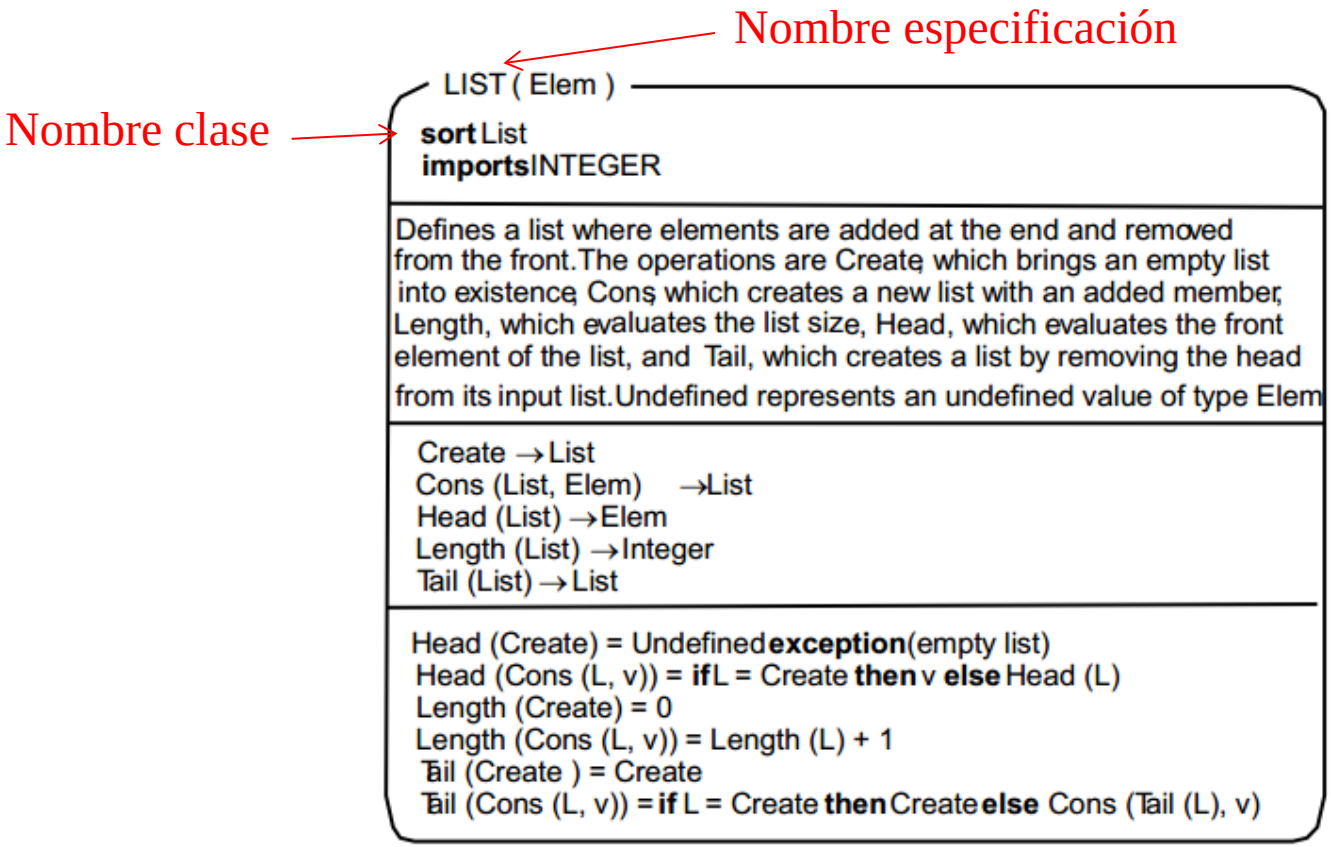
Estructura de una especificación algebraica

- **Introducción**. Define el nombre del tipo y declara otras especificaciones que se usan.
- **Descripción**. Describe informalmente las operaciones del tipo. La notación completa esta descripción con una sintaxis y semántica no ambiguas para las operaciones del tipo.
- **Signatura**. Define la sintaxis de las operaciones, número y clase de sus parámetros, y las clases de los resultados.
- **Axiomas**. Define la semántica de las operaciones mediante la definición de axiomas que caracterizan su comportamiento.



Especificación algebraica

Estructura de una especificación algebraica



Especificación algebraica

Proceso de desarrollo de una especificación algebraica

1. Estructura de la especificación.
   - Conjunto de TDA o clases.
   - Definir informalmente las operaciones de cada clase.
2. Nombrado de la especificación.
   - Nombre, parámetros.
3. Selección de las operaciones.
   - Operaciones.
   - Incluir operaciones para crear instancias y modificar/inspeccionar valores de las instancias.
4. Especificación informal de las operaciones.
   - Cómo las operaciones afectan a la clase definida.
5. Definición de la sintaxis.
   - Se define la sintaxis de las operaciones y sus parámetros.
6. Definición de los axiomas.
   - Semántica de las operaciones describiendo qué condiciones son siempre verdaderas.

Índice

1. Especificación de requisitos
2. El documento de requisitos
3. Especificación formal
    1. Especificación algebraica
    2. Especificación basada en modelos

Especificación basada en modelos

- Especificación basada en modelos:
  - Permite la especificación del comportamiento del sistema.
  - La especificación del sistema se expresa como un modelo de estados del sistema.
  - Las operaciones se especifican definiendo cómo afectan al estado del sistema.
  - Algunas notaciones: VDM (Jones, 1980;1986), B (Wordsworth, 1996) y Z (Hayes, 1987; Spivey,1992).
- Las especificaciones algebraicas → operaciones del objeto son independientes del estado.
- La combinación de ambas define el comportamiento del sistema en su totalidad.

Especificación basada en modelos

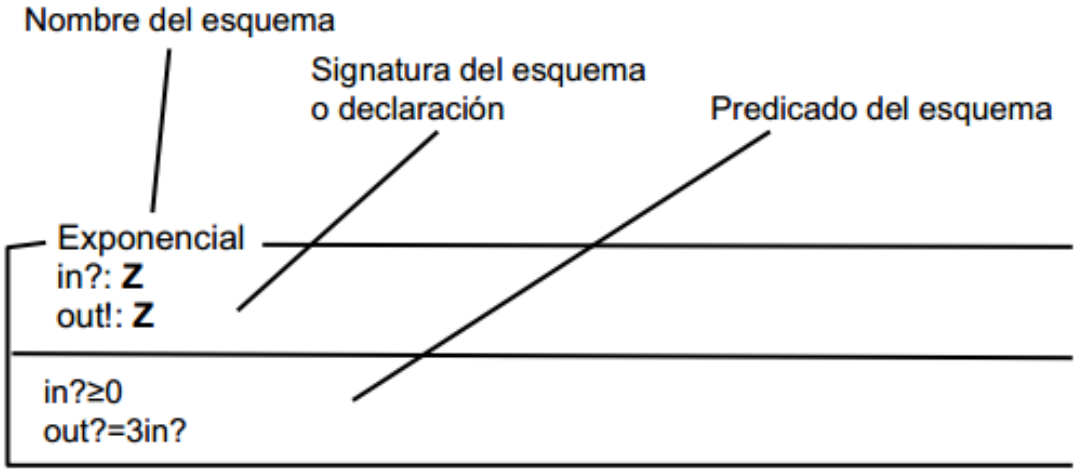
Z:

- Los sistemas se modelan usando conjuntos y relaciones entre conjuntos.
- Las especificaciones se presentan como texto informal complementado con descripciones formales → legibilidad
- Las descripciones formales se incluyen como pequeños trozos fáciles de leer: esquemas.
- Los esquemas introducen variables de estado y definen restricciones y operaciones en el estado.

Especificación basada en modelos

Z: ejemplo de la función exponencial:

- La **signatura** define las entidades que constituyen el estado del sistema.
- El **predicado** establece las condiciones que siempre deben cumplirse para esas entidades.



Especificación basada en modelos

Z

- Los esquemas pueden ser manipulados utilizando operaciones tales como composición de esquemas, renombrado de esquemas y ocultación de esquemas → Estas operaciones permiten manipular los esquemas y definir esquemas más complejos
- La signatura del esquema define las entidades que forman el estado del sistema y el predicado del esquema establece las condiciones que deberían cumplirse siempre para estas entidades.
- Cuando un esquema define una operación, el predicado puede establecer precondiciones y postcondiciones.
  - Estás definen el estado antes y después de la operación. La diferencia entre ambas define la acción especificada en el esquema de la operación.

Especificación basada en modelos

- **Ejemplo “Raíz cuadrada”**

Esquema

Raíz Cuadrada Entera

$x? : \mathbb{N}$

$r! : \mathbb{N}$

$x? \geq 0$

$(r! * r!) \leq x? \land (r! + 1) * (r! + 1) > x?$

Especificación basada en modelos

Ejemplo “Banco”

- Cada cliente se identifica por su número de documento y puede tener sólo una caja de ahorro.
- No se guardará el historial de transacciones de las cajas; sólo se preservará el saldo de cada una.
- Los clientes sólo pueden extraer y depositar dinero en efectivo, y solicitar el saldo de su caja.
- Un cliente puede solicitar el cierre de su caja de ahorro sólo si su saldo es cero.

Especificación basada en modelos

Z Ejemplo “Banco”

- *Tipos elementales:*

[DNI]

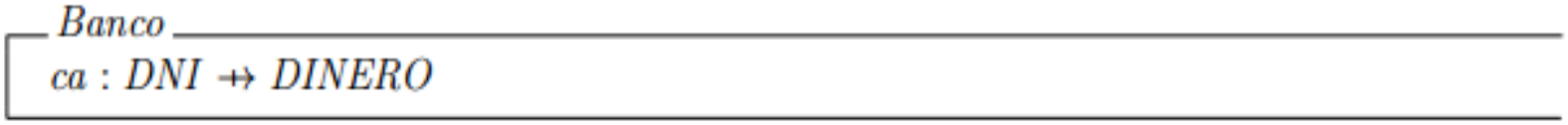
En Z el único tipo básico nativo es Z, es decir los enteros. Los naturales se pueden definir: $N == \{n : Z \mid 0 \leq n\}$

DINERO == N

Especificación basada en modelos

Z Ejemplo “Banco”

- *Estado del banco:*



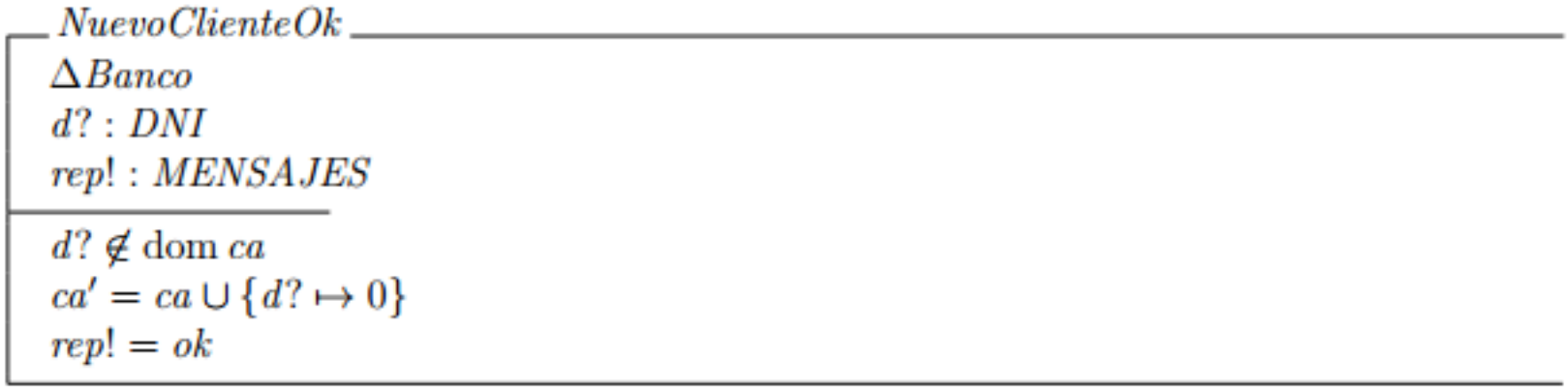
- El banco “es" sus clientes y sus cajas de ahorro
- Hay una relación funcional entre los clientes y sus cajas de ahorro.
- Función parcial: no todos los elementos de DNI son clientes del banco. DNI representa a todos los DNIs de todos los potenciales clientes del banco, y no a todos los clientes actuales.

- *El estado inicial del banco:*

Especificación basada en modelos

Z Ejemplo “Banco”

- *Las operaciones:*
  - Nuevo cliente: (I) Cuando una persona se transforma en cliente del banco se le crea una caja de ahorro. (II) El saldo inicial de la cuenta es cero



- (?): indica que son variables de *entrada*.
- (!): indica que son variables de salida
- Δ: indica que tras esta operación se producirá un *cambio de estado*

Especificación basada en modelos

Z Ejemplo “Banco”

- *Las operaciones:*
  - Nuevo cliente: esquema de error de la operación

$$
\begin{array}{|l}
\hline
NuevoClienteE \\
\Xi Banco \\
d? : DNI \\
rep! : MENSAJES \\
\hline
d? \in \mathrm{dom}\ ca \\
rep! = numeroClienteEnUso \\
\hline
\end{array}
$$

- Θ: indica que tras esta operación no se producirá un *cambio de estado*
- MENSAJES habría que definirlo

La operación total Nuevo cliente:

$$NuevoCliente == NuevoClienteOk \lor NuevoClienteE$$

Especificación basada en modelos

Z Ejemplo “Banco”

- *Las operaciones:*
  - Depositar dinero en una caja de ahorro (I)

DepositarOk

$\Delta Banco$
$d? : DNI$
$m? : DINERO$
$rep! : MENSAJES$

$d? \in \text{dom } ca$
$0 < m?$
$ca' = ca \oplus \{d? \mapsto (ca\ d?) + m?\}$
$rep! = ok$

Especificación basada en modelos

- **Ejemplo “Banco”**
- *Las operaciones:*
  - Depositar dinero en una caja de ahorro (II)

$$
\begin{array}{|l}
\hline
DepositarE1 \\
\Xi Banco \\
d? : DNI \\
rep! : MENSAJES \\
\hline
d? \notin \mathrm{dom}\ ca \\
rep! = clienteInexistente \\
\hline
\end{array}
$$

$$
\begin{array}{|l}
\hline
DepositarE2 \\
\Xi Banco \\
m? : DINERO \\
rep! : MENSAJES \\
\hline
m? \leq 0 \\
rep! = montoNulo \\
\hline
\end{array}
$$

Especificación basada en modelos

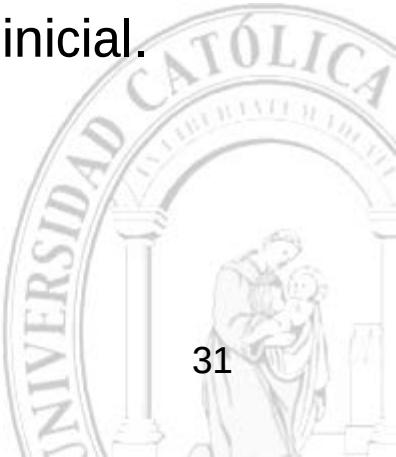
- **Ejemplo “Banco”**
- *Las operaciones:*
  - Depositar dinero en una caja de ahorro (III)

$$DepositarE == DepositarE1 \lor DepositarE2$$

$$Depositar == DepositarOk \lor DepositarE$$

Puntos clave

- Los métodos de especificación formal complementan a las técnicas de especificación informal de requisitos. Pueden utilizarse con la definición de requisitos mediante lenguaje natural para clarificar cualquier área de ambigüedad.
- Las especificaciones formales son precisas y no ambiguas, evitando problemas de mala interpretación del lenguaje. Sin embargo, los no especialistas pueden encontrar estas especificaciones difíciles de entender.
- Un ventaja fundamental de usar métodos formales es que fuerza a un análisis de los requisitos del sistema en una etapa inicial. Corregir errores en esta etapa es menos costoso.



Puntos clave

- Las técnicas algebraicas de especificación formal son especialmente adecuadas para especificar interfaces en donde la interfaz se define como un conjunto de clases de objetos o tipos abstractos de datos. Éstas técnicas ocultan el estado del sistema y especifican el sistema en función del las relaciones entre las operaciones de la interfaz.
- Las técnicas basadas en modelos modelan el sistema utilizando construcciones matemáticas tales como conjuntos y funciones. Éstas pueden mostrar el estado del sistema, lo que simplifica algunos tipos de especificación del comportamiento.