Estructuras de Datos y Algoritmos

Especificación de TAD

Lección 2

Universidad Zaragoza

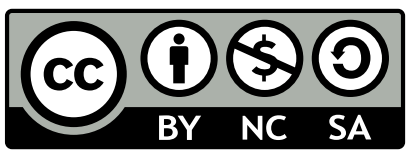
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, España

Curso 2024/2025

Grado en Ingeniería Informática

Universidad de Zaragoza

Aula 0.04, Edificio Agustín de Betancourt



Adaptadas de diapositivas de Javier Campos

Índice

Lectura recomendada:

Sección 1.3 del libro de Z.J. Hernández et al. de la bibliografía

Universidad Zaragoza

Índice

Universidad Zaragoza

Características generales

Definición de *Tipo Abstracto de Datos* (TAD)

- Conjunto de valores y operaciones definidos mediante una **especificación independiente de cualquier representación e implementación**

$$\text{TAD} = \textsc{valores} + \textsc{operaciones}$$

Universidad Zaragoza

Características generales
Programación con TADs

Pasos en la programación con TAD:

1. **Especificación**
2. **Implementación**
3. **Uso**

Universidad Zaragoza

Características generales
Programación con TADs

Pasos en la programación con TAD:

1. **Especificación**
   - **Establecer la interfaz** con el usuario del tipo (*lo que necesita saber el usuario-programador*)
     - **Decir qué es el TAD, sin decir cómo se implementa**
   - Decidir la **lista de operaciones necesarias y especificarlas**
     - Información de entrada, salida, y comportamiento/efecto de cada operación
2. **Implementación**
3. **Uso**

Universidad Zaragoza

Características generales
Especificación de un TAD

- Consiste en **establecer las propiedades que lo definen**
- Características de una **especificación útil**
  - *Concisa* (breve, debe decir lo imprescindible)
  - *General* (adaptable a diferentes contextos)
  - *Legible* (para todos)
  - *Precisa* (exenta de ambigüedades para evitar diferentes interpretaciones)

La especificación define un único tipo, es decir, define totalmente su comportamiento

Universidad Zaragoza

Características generales
Especificación de un TAD

Alternativas

- **Especificaciones formales**: *especificación algebraica*
  - Muy precisa y concisa
  - Sintaxis específica
  - Verificación formal de TADs
- **Especificaciones no formales**
  - Expresadas en lenguaje natural (generalmente)
  - Debe ser precisa, concisa, general, legible y definir totalmente su comportamiento

Universidad Zaragoza
1542

Índice

Universidad Zaragoza

Especificación algebraica

- **Técnica formal para especificar un TAD**
- Objetivo:
  - **Definir un tipo de datos sin ambigüedades**
    (conjunto de valores y operaciones permitidas)
- **Ventajas**:
  - Unanimidad en la interpretación
  - Definición de tipos independientemente de la representación
  - Deducción de propiedades satisfechas por implementaciones correctas (verificación formal)
  - Generación automática de código

Universidad Zaragoza
1542

Especificación algebraica

- **Técnica formal para especificar un TAD**
- Objetivo:
  - **Definir un tipo de datos sin ambigüedades**
    (conjunto de valores y operaciones permitidas)
- **Ventajas**:
  - Unanimidad en la interpretación
  - Definición de tipos independientemente de la representación
  - Deducción de propiedades satisfechas por implementaciones correctas (verificación formal)
  - Generación automática de código
- **Una especificación tiene dos partes**:
  - *Parte sintáctica*: nombre del tipo y operaciones
  - *Parte semántica*: propiedades de las operaciones

Universidad Zaragoza

Especificación algebraica

Ejemplo: el TAD booleano

```
espec booleanos
    género booleano
    operaciones
        verdadero: -> booleano
        falso: -> booleano
        ¬_: booleano -> booleano
        _∧_, _∨_,: booleano booleano -> booleano
    ecuaciones b: booleano
        ¬verdad = falso
        ¬falso = verdad
        b ∧ verdad = b
        b ∧ falso = falso
        b ∨ verdad = verdad
        b ∨ falso = b
fespec
```

Recordatorio: AND es $\wedge$*, OR es* $\vee$*, NOT es* $\neg$

Especificación algebraica

Componentes

- **SIGNATURA** → parte sintáctica, donde se define
  - **GÉNEROS**: nombres de los nuevos tipos
  - Nombre de las **OPERACIONES**
  - **PERFILES** de las operaciones
    - Dominio (o aridad)
    - Rango (o coaridad)
- Conjunto de **ECUACIONES** → parte semántica

Universidad Zaragoza

Especificación algebraica

Ejemplo: signatura del TAD booleano y natural

```
espec boolnat
    géneros booleano, natural
    operaciones
        verdadero: -> booleano
        falso: -> booleano
        ¬_: booleano -> booleano
        _∧_, _∨_,: booleano booleano -> booleano
        0: -> natural
        suc: natural -> natural
        _+_, _*_,: natural natural -> natural
        _≤_, _<_,: natural natural -> booleano
fespec
```

- _ indica la posición de los argumentos respecto al nombre de la operación
  - Operaciones prefijas sin paréntesis (e.g., ¬_)
  - Operaciones infijas (e.g., _+_)

Universidad Zaragoza

Especificación algebraica

Sobre las OPERACIONES

- **Tipos de operaciones**: generadoras, modificadoras, u observadoras
- **Notación funcional**
  - Toma como parámetros 0 o N valores (dominio)
  - Produce un único valor resultado (rango)
- Las operaciones 0-arias **se denominan CONSTANTES de su tipo de dato**
  - Por ejemplo, las operaciones `verdadero`, `falso` y `0` en el TAD booleano y natural

Universidad Zaragoza

Especificación algebraica

```
espec naturales
    género natural
    operaciones
        0: -> natural
        suc: natural -> natural
        _+_,: natural natural -> natural
    ecuaciones x, y: natural
        x + 0 = x
        x + suc(y) = suc(x + y)
fespec
```

Sobre las ECUACIONES

- ∀ término bien formado de los géneros de las variables

Especificación algebraica

Ejemplo: el TAD fecha

```
espec fechas
    usa enteros, booleanos
    género fecha
    operaciones
        parcial crear: entero entero entero -> fecha

        dia: fecha -> entero
        mes: fecha -> entero
        año: fecha -> entero

        iguales: fecha fecha -> booleano
        anterior: fecha fecha -> booleano
        posterior: fecha fecha -> booleano
    dominios de definición d, m, a: entero
        crear(d,m,a) está definida si y solo si 1 ≤ d ≤ 31 y 1 ≤ m ≤ 12, y
        además d, m y a componen una fecha válida (es decir, el valor de
        día d es adecuado para el mes m y el año a, según las reglas del
        calendario gregoriano)
...
```

Especificación algebraica

Ejemplo: el TAD fecha (continuación)

```
...
    ecuaciones d, d1, d2, m, m1, m2, a, a1, a2: entero; f1, f2: fecha
        dia(crear(d, m, a)) = d
        mes(crear(d, m, a)) = m
        año(crear(d, m, a)) = a

        iguales(crear(d1, m1, a1), crear(d2, m2, a2)) =
                        = (d1 = d2) ∧  (m1 = m2) ∧ (a1 = a2)
        anterior(crear(d1, m1, a1), crear(d2, m2, a2)) =
                        = (a1  < a2) ∨  ((a1 = a2) ∧ (m1 < m2))
                        ∨ ((a1 = a2) ∧ (m1 = m2) ∧ (d1 < d2))
        posterior(f1, f2) = ¬(iguales(f1,f2) ∨ anterior(f1, f2))
fespec
```

Especificación algebraica
Especificación formal de TAD

Ventajas

- **Demostrar que la especificación es correcta**
- **Demostrar la corrección de una implementación (a partir de su especificación)**
  - La implementación es fiel a la especificación
- **Corrección de los programas** que utilicen TADs
- **Generar código de implementación del TAD (*prototipo*), a partir de su especificación**

Universidad Zaragoza
1542

Índice

Universidad Zaragoza

Especificación no formal

- Expresadas **en lenguaje natural** (generalmente)
- También debe ser **precisa, concisa, general, legible y definir totalmente el comportamiento del TAD y sus operaciones**

Similar a un "contrato público"

Especificación no formal

Características

- Describir **conjunto de valores y operaciones del TAD**, y establecer todas las **propiedades** que lo definen, de forma **independiente de cualquier posible representación o implementación**

Especificación no formal

Características

- Describir **conjunto de valores y operaciones del TAD**, y establecer todas las **propiedades** que lo definen, de forma **independiente de cualquier posible representación o implementación**
- **Sintaxis parecida** a la vista para especificaciones algebraicas
  - **Notación funcional** (para las operaciones)
    - Toma como parámetros 0 o N valores (dominio)
    - Produce un único valor resultado (rango)
  - **Diferencias**
    - **Cada uno de los parámetros del dominio tendrá un nombre**
    - **NO hay sección de ecuaciones**: la parte semántica se describe textualmente

Universidad Zaragoza

Especificación no formal

Componentes

- **SIGNATURA** → parte sintáctica, que define
  - **GÉNEROS**: nombres de los nuevos tipos
  - Nombre de las **OPERACIONES**
  - **PERFILES** de las operaciones
    - Dominio (o aridad)
    - Rango (o coaridad)
- **Descripciones textuales** → parte semántica
  - Escritas en lenguaje natural, pero de forma precisa, general, legible y no ambigua

Universidad Zaragoza
1542

Especificación no formal

Parte sintáctica

Disponible en el material de clase:

Resumen (chuleta) de la notación no formal que se utilizará en la asignatura para especificar TADs

```
espec nombreEspecificacion
    [usa especificación1, especificación2, ...]
    ... {aquí una parte de la especificación que se verá más adelante}
    género nombreGénero1, ...
    operaciones
        [parcial] nombreOperacion: [dominio] -> rango
        [parcial] nombreOperacion: genArg nomArg -> rango
        [parcial] _ nombreOp _ : genArg1 nomArg1, genArg2 nomArg2 -> rango
fespec
```

- [ ] indica opcionalidad (es decir, lo que está entre [ ] puede aparecer o no)
- **Dominio**: lista de elementos separados por `','`
  - Cada elemento describe un argumento de la operación, indicando su **género** y **nombre**
- **Rango**: es un género
- El símbolo `'_'` indica la posición de los argumentos respecto a la operación
  - Se utiliza para indicar operaciones con notación prefija sin paréntesis o infija

Universidad Zaragoza

Especificación no formal

Parte semántica

- **Sin detalles de posibles implementaciones**

La especificación del TAD es independiente de cualquier representación o implementación del TAD

- **Descripciones textuales**, escritas en lenguaje natural
- De forma precisa, general, legible y no ambigua

Universidad Zaragoza

Especificación no formal

Contenido de las descripciones textuales

- **DESCRIPCIÓN del dominio de valores del TAD**
  - Describiendo conjunto de valores que representa el TAD y las operaciones que lo definen
- **PARA CADA OPERACIÓN**: junto a su perfil, hay que describir completamente:
  1. Información de entrada y prerrequisitos para usar la operación (*precondición*)
  2. Comportamiento o efecto de la operación al aplicarse a las entradas, indicando qué resultado se genera (*postcondición*)
  3. Situaciones indeseadas o de *error*
     - Aquellos casos para los que no existe un resultado válido de la operación
     - Se consideran **operaciones parciales**, indicándose en la especificación

Universidad Zaragoza

Especificación no formal

Parte sintáctica y semántica

Disponible en el material de clase:

Resumen (chuleta) de la notación no formal que se utilizará en la asignatura para especificar TADs

```
espec nombreEspecificacion
    [usa especificación1, especificación2, ...]
    ... {aquí una parte de la especificación que se verá más adelante}
    género nombreGénero1, ... {descripción del TAD}
    operaciones
        [parcial] nombreOperacion: [dominio] -> rango
        {Descripción del dominio y del rango de la operación
        Parcial: descripción de las situaciones que hacen la operación parcial}
        ...
        [parcial] nombreOperacion: genArg nomArg -> rango
        {Descripción del dominio y del rango de la operación
        Parcial: descripción de las situaciones que hacen la operación parcial}
        ...
        [parcial] _ nombreOp _ : genArg1 nomArg1, genArg2 nomArg2 -> rango
        {Descripción del dominio y del rango de la operación
        Parcial: descripción de las situaciones que hacen la operación parcial}
        ...
fespec
```

Universidad Zaragoza

Especificación no formal

Ejemplo: el TAD fechas

```
espec fechas
    usa enteros, booleanos
    género fecha
    {(Descripción del TAD:) Los valores del TAD fechas representan
        fechas válidas según las reglas del calendario gregoriano}
    operaciones
        parcial crear: entero d, entero m, entero a -> fecha
        {Dados los tres valores enteros, se obtiene una fecha compuesta
             con los tres valores dados usados como día, mes y año
            respectivamente.
        Parcial: 1 ≤ d ≤ 31, 1 ≤ m ≤ 12, 1582 ≤ a y además deben formar
             una fecha válida según el calendario gregoriano.}

        día: fecha f -> entero
        {Dada una fecha f, se obtiene el entero que corresponde al día
            en la fecha f}

        mes: fecha f -> entero
        {Dada una fecha f, se obtiene el entero que corresponde al mes
            en la fecha f}

        año: fecha f -> entero
        {Dada una fecha f, se obtiene el entero que corresponde al año
            en la fecha f}
...
```

Especificación no formal

Ejemplo: el TAD fechas

```
...
        iguales: fecha f1 , fecha f2 -> booleano
        {Dadas dos fechas f1 y f2, se obtiene un booleano con valor
            verdad si y solo si la fecha f1 es igual que la fecha f2,
            es decir, corresponden al mismo día, mes y año.}

        anterior: fecha f1 , fecha f2 -> booleano
        {Dadas dos fechas f1 y f2, se obtiene un booleano con valor
            verdad si y solo si la fecha f1 es cronológicamente
            anterior a la fecha f2.}

        posterior: fecha f1 , fecha f2 -> booleano
        {Dadas dos fechas f1 y f2, se obtiene un booleano con valor
            verdad si y solo si la fecha f1 es cronológicamente
            posterior a la fecha f2.}
fespec
```

Especificación no formal
Algunos comentarios finales

- **Sobre las operaciones parciales**:
  - **Los usuarios del TAD han de asegurarse de que se cumplen los prerrequisitos** (es decir, los valores de los argumentos están dentro del dominio de la definición) antes de utilizar la operación
  - Aún así, **implementación del TAD robusta** (*programación defensiva*): protegerse frente a valores inconsistentes o que no cumplan los prerrequisitos (tratamiento de errores)
- **Mecanismos de protección frente a errores**:
  - **Implementación dependiente del lenguaje de programación a utilizar**
    - Manejo de excepciones
    - Parámetros de salida de error en cada operación
  - **Indicar en la especificación las situaciones de error como información adicional** para el implementador del TAD y para el usuario del TAD

Universidad Zaragoza

Especificación no formal

Ejercicio

Diseña un programa que lea una secuencia de enteros de un fichero, mayores que 0, y al acabar escriba en pantalla cada entero leído del fichero junto con su frecuencia de aparición (es decir, número total de apariciones del número en el fichero), siguiendo el orden de frecuencias decrecientes.

Especificación no formal

Ejercicio

Diseña un programa que lea una secuencia de enteros de un fichero, mayores que 0, y al acabar escriba en pantalla cada entero leído del fichero junto con su frecuencia de aparición (es decir, número total de apariciones del número en el fichero), siguiendo el orden de frecuencias decrecientes.

Posible solución dividida en dos módulos:

- *Módulo principal*, que interactúa con el usuario y se encarga de:
  - Preguntar y obtener el nombre del fichero a leer
  - Leer el contenido del fichero y generar los resultados
  - Mostrar los resultados (números y su frecuencia ordenados) al usuario
- *Módulo con operaciones que permitan almacenar los enteros leídos, junto con el número de veces que aparecen, y preguntar sobre los resultados necesarios* → **TAD tabla**

Universidad Zaragoza

Especificación no formal

Ejercicio – el TAD tabla

```
espec tablas
    usa naturales, enteros {supondremos que el 0 está en los naturales}
    género tabla
    {((DESCRIPCIÓN:)
    Los valores del TAD tablas de frecuencia representan colecciones de
         números enteros tales que:
        - no se almacenan enteros repetidos, pero sí se registra
            cuántas veces se ha introducido cada entero (su frecuencia)
        - las operaciones permiten obtener la información de un entero
            o su frecuencia
según su puesto en el orden decreciente por valores de frecuencia}

    operaciones
        inicializar: -> tabla
        {Devuelve una tabla vacía, es decir, que no contiene datos para
             ningún número entero}

        añadir: tabla t, entero e -> tabla
        {Si e no está t, devuelve la tabla resultante de añadir e a t
            con número de apariciones igual a 1; si e está en t,
            devuelve la tabla resultante de incrementar en 1 el número
            de apariciones de e (su frecuencia) en t}
...
```

Especificación no formal

Ejercicio – el TAD tabla

```
...
        total: tabla t -> natural
        {Devuelve el número total de enteros para los que t contiene
            información}

        parcial infoEnt: tabla t , natural n -> entero
        {Devuelve el entero que corresponde al n-ésimo entero en t
            según el orden en número de apariciones decreciente.
        Parcial: la operación no está definida si n=0 OR n>total(t)}

        parcial infoFrec: tabla t , natural n -> natural
        {Devuelve el natural que corresponde al número de apariciones
            del n-ésimo entero en la tabla t según el orden en número
            de apariciones decreciente.
        Parcial: la operación no está definida si n=0 OR n>total(t)}

fespec
```

Trabajo encargado

Leer lecciones 1 y 2 de los apuntes y material adicional de Moodle

Estructuras de Datos y Algoritmos

Especificación de TAD

Lección 2

Universidad Zaragoza

1542

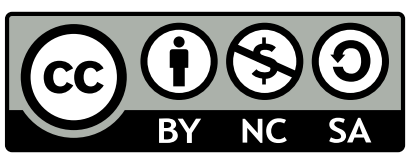
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, España

Curso 2024/2025

Grado en Ingeniería Informática

Universidad de Zaragoza

Aula 0.04, Edificio Agustín de Betancourt



Adaptadas de diapositivas de Javier Campos