Estructuras de Datos y Algoritmos

Tipos Abstractos de Datos

Lección 1

Universidad Zaragoza

1474

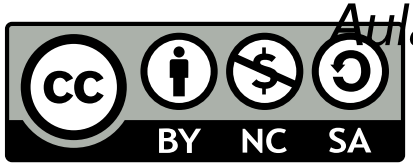
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, España

Curso 2025/2026

Grado en Ingeniería Informática

Universidad de Zaragoza

Aulas A.02 (421) y A.04 (423), Edificio Ada Byron



Adaptadas de diapositivas de Javier Campos

Índice

Lectura recomendada:

Secciones 1.1 y 1.2 del libro de Z.J. Hernández et al. de la bibliografía

Universidad Zaragoza

Índice

Universidad Zaragoza

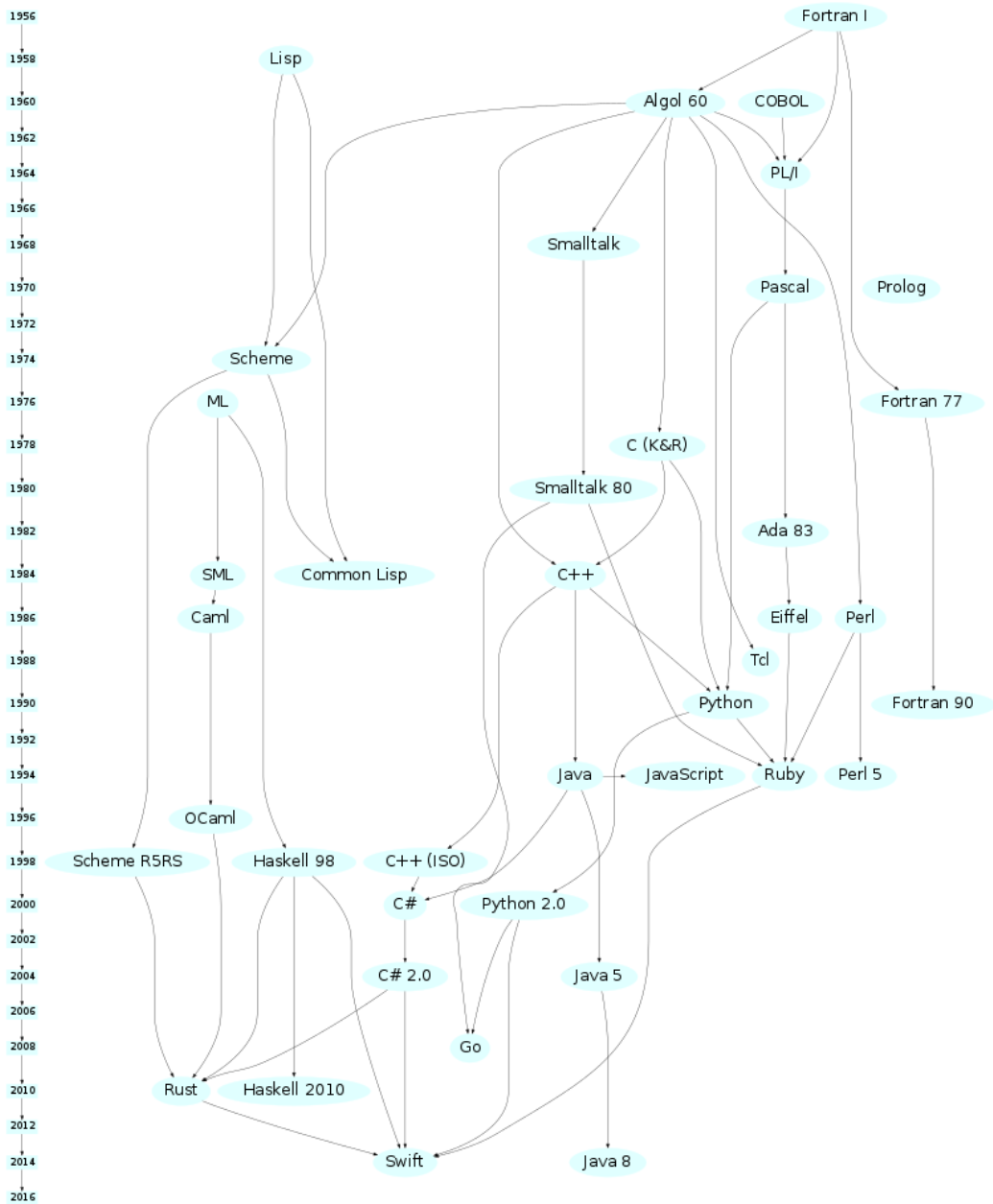
Paradigmas de programación

- *Conjunto de teorías, estándares y métodos que en conjunto representan un medio de organización del conocimiento*
- Múltiples paradigmas y clasificaciones de paradigmas
- Evolución histórica:

- Programación Imperativa (época “caótica” hasta los 60s)
- Programación Estructurada (auge 70s)
- Programación Modular (80s)
- Programación Orientada a Objetos (90s)
- Paradigmas Declarativos
  - Programación Lógica
  - Programación Funcional
- Con sus evoluciones modulares, orientadas a objetos, ...

Universidad Zaragoza

Paradigmas de programación



Créditos: https://ingenieriadesoftware.es/historia-visual-lenguajes-programacion/

Paradigmas de programación
Diseño descendente

- **Diseño mediante abstracción y refinamientos sucesivos**
  - Basado en la abstracción de acciones
  - Ineficiente para programación a gran escala

Paradigmas de programación
Diseño descendente

- **Diseño mediante abstracción y refinamientos sucesivos**
  - Basado en la abstracción de acciones
  - Ineficiente para programación a gran escala
- **Inconvenientes**
  - Acciones abstractas (procedimientos y funciones) **de alto nivel, que manipulan datos concretos de bajo nivel**
  - Fuerza a seleccionar la representación de los datos desde el principio
    - *Esta es una decisión que conviene retrasar hasta identificar qué operaciones serán necesarias...*

Universidad Zaragoza

Paradigmas de programación
Diseño modular

- Ampliación/evolución del diseño descendente
- **División en módulos**:
  - Cada uno con una misión bien determinada, interacción mínima y clara con el resto, que puedan ser desarrollados de forma independiente, reutilizables, etc.
  - Todos juntos solucionan el problema inicial
- **Requisitos de la descomposición en módulos**:
  - Cada módulo debe tener una conexión mínima con el resto: **interfaz**
    - Garantizar la independencia en el desarrollo
    - Reutilización
  - Cambios y mejoras deben afectar sólo a un número pequeño de módulos
  - Tamaño adecuado (ni muy grande, ni muy pequeño)

Universidad Zaragoza

Índice

Universidad Zaragoza

Definición de TAD

- En los 60s, junto con los lenguajes de programación estructurados, surge el concepto de *tipo de datos*
  - Conjunto de valores que sirve de dominio de ciertas operaciones
- Concepto ***insuficiente*** para software a gran escala: sólo el compilador restringe el uso de los datos

Definición de TAD

- En los 60s, junto con los lenguajes de programación estructurados, surge el concepto de *tipo de datos*
  - Conjunto de valores que sirve de dominio de ciertas operaciones
- Concepto ***insuficiente*** para software a gran escala: sólo el compilador restringe el uso de los datos
- En los 70s aparece el concepto de *Tipos Abstractos de Datos*
  - El tipo de datos no sólo es el conjunto de valores, sino también sus operaciones (con sus propiedades) → determinan inequívocamente su comportamiento

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

Definición de TAD

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

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

- La utilización de un TAD sólo depende de su especificación (conjunto de valores, operaciones y sus propiedades), **nunca de su implementación**

Construiremos nuevos tipos de datos, robustos y eficientes, para ser usados de forma similar a los predefinidos en un lenguaje de programación, pero de naturaleza mucho más compleja que los tipos predefinidos

Universidad Zaragoza

Definición de TAD

- Concepto ya existente en lenguajes de programación estructurados: los *tipos predefinidos*
  - No es necesario saber nada sobre implementación para usarlos
- **Ejemplo:** Definición del tipo de datos de los enteros en C++
  - **Valores**: los que vienen dados por el número de bytes utilizados para su representación (i.e., dados por `sizeof(tipo)`)
  - **Operaciones**: +, -, *, /, módulo (resto de la división), etc.
  - **Propiedades de las operaciones**: $a + b = b + a$, $a + 0 = a$, $a * 0 = 0$, etc.

Definición de TAD

Ejemplos de TAD

Las fechas tal como las usamos habitualmente...

Definición de TAD
Ejemplos de TAD

Las fechas tal como las usamos habitualmente...

¿Cómo las especificamos?

Definición de TAD

- Separar estrictamente la representación de un tipo de dato, del uso del tipo de datos → ***abstracción y ocultación***
  - Un tipo consta del conjunto de valores y del conjunto de operaciones para la creación, modificación y manipulación de los valores del tipo → **especificación**
  - La única forma de usar los datos del tipo abstracto será invocando las operaciones de la especificación
- No se podrá acceder a la representación interna del tipo de dato
- Para usar un tipo no hace falta saber el detalle de su implementación, sino sólo su especificación

Universidad Zaragoza
1474

Índice

Universidad Zaragoza

Programación con TAD

Definición e implementación de TAD

- En programación modular, el TAD se encapsula en un módulo
- En programación O.O., el TAD se encapsula en una clase

Pasos en la programación con TAD

1. Especificación del tipo en la interfaz (del módulo o de la clase) → resultante de lo identificado en la fase de diseño
   - Las operaciones necesarias para el TAD se conocerán tras haber diseñado el programa/s usuarios del TAD
2. Implementación del tipo
   - Al implementar el TAD se buscará la máxima eficiencia (se conocen las operaciones a ofrecer)
3. Uso del tipo

Universidad Zaragoza

Programación con TAD

Especificación e implementación de TAD

- Dada una especificación de un TAD, podremos tener diferentes implementaciones del mismo TAD
  - Siempre que respeten la misma especificación, las implementaciones serán intercambiables de forma transparente para los usuarios (programas) del TAD
- Dada una especificación de un TAD, se podrán implementar de forma independiente, simultáneamente y por separado:
  a) Los algoritmos que implementan las operaciones del TAD
  b) Los algoritmos que usan el TAD

Universidad Zaragoza
1474

Programación con TAD

Pasos en la programación con TAD

1. Especificación del tipo
2. Implementación del tipo
3. Uso del tipo

Programación con TAD

Pasos en la programación con TAD

1. Especificación del tipo
2. Implementación del tipo
3. Uso del tipo

Especificación del TAD

- **Establecer la interfaz con el usuario del tipo** ("*lo que necesita saber el usuario-programador*")
  Decir qué es sin decir nada sobre cómo se hace → contrato público
- **Decidir la lista de operaciones necesarias y especificarlas**
  - Para cada operación hay que describir: información de entrada, información de salida, y el comportamiento o efecto de la operación
- Debe ser **precisa** (no ambigua), **concisa** (breve) y **legible**
- Especificación de TAD: formales versus informales

Universidad Zaragoza 1474

Programación con TAD

Criterios de calidad del software

- **Corrección**
  - Debe realizar exactamente las tareas definidas por su especificación
- **Legibilidad**
  - Debe ser fácil de leer y lo más sencillo posible. Contribuyen la abstracción y la codificación con comentarios, sangrados, etc.
- **Extensibilidad**
  - Facilidad para adaptarse a cambios en su especificación
- **Robustez**
  - Capacidad de funcionar correctamente incluso en situaciones anormales
- **Eficiencia**
  - Hace un buen uso de los recursos, tales como el tiempo, espacio (memoria y disco), etc.
- **Facilidad de uso**
  - La utilidad de un sistema está relacionado con su facilidad de uso

Universidad Zaragoza

Programación con TAD

Criterios de calidad del software

- **Portabilidad**
  - Facilidad con la que puede ser transportado a diferentes sistemas físicos o lógicos
- **Verificabilidad**
  - Facilidad de verificación de un software, su capacidad para soportar los procedimientos de validación, juegos de test, ensayo o pruebas
- **Reutilización**
  - Capacidad de ser reutilizados en nuevas aplicaciones o desarrollos
- **Integridad**
  - Capacidad de proteger sus propios componentes frente a accesos o usos indebidos
- **Compatibilidad**
  - Facilidad para ser combinados con otros y usados en diferentes plataformas hardware o software

Universidad Zaragoza
1474

Índice

Universidad Zaragoza

Programación con TAD: ventajas

- **Abstracción**:
  - La complejidad del problema se diluye. Los módulos en los que se descompone el problema serán de menor complejidad
  - Se pueden implementar los TAD sólo a partir de la especificación, sin saber para qué se van a usar → *reusabilidad*
- **Corrección**:
  - Los TAD pueden ser desarrollados y probados de forma independiente
  - Se pueden utilizar los TAD sólo conociendo la especificación (facilita la integración de módulos)
- **Eficiencia**:
  - La implementación puede retrasarse hasta conocer las restricciones de eficiencia sobre sus operaciones
  - Para un TAD podemos contar con diferentes implementaciones validas y optar por la más eficiente y adecuada a las restricciones a cumplir
- **Legibilidad**:
  - La especificación de un TAD es suficiente para entender su significado y comportamiento
  - Un TAD tiene un tamaño y complejidad acotados que facilita su legibilidad

Universidad Zaragoza 1474

Programación con TAD: ventajas

- **Modificabilidad y mantenimiento**:
  - Tanto durante el periodo de desarrollo y pruebas, como durante en el periodo de mantenimiento y de vida del software
  - Cambios localizados y acotados
  - Cambios que no afecten a la especificación no afectan a los programas que usen el TAD
- **Organización**:
  - Facilita el reparto de tareas y la comunicación en un grupo de programadores
  - El equipo desarrolla en paralelo las múltiples partes o módulos del sistema
- **Reusabilidad**:
  - TAD reutilizables en otros contextos con pocos o ningún cambio
    (¡escoger bien el conjunto de operaciones!)
- **Seguridad**:
  - Imposibilidad de manipular directamente la representación interna de los datos u objetos del tipo
  - Impide el mal uso y la generación de valores incorrectos

Universidad Zaragoza
1474

Trabajo encargado

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

Estructuras de Datos y Algoritmos

Tipos Abstractos de Datos

Lección 1

Universidad Zaragoza

1474

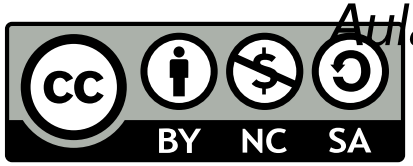
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, España

Curso 2025/2026

Grado en Ingeniería Informática

Universidad de Zaragoza

Aulas A.02 (421) y A.04 (423), Edificio Ada Byron



Adaptadas de diapositivas de Javier Campos