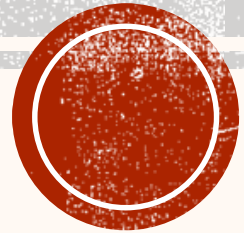
ABSTRACCIÓN DE DATOS

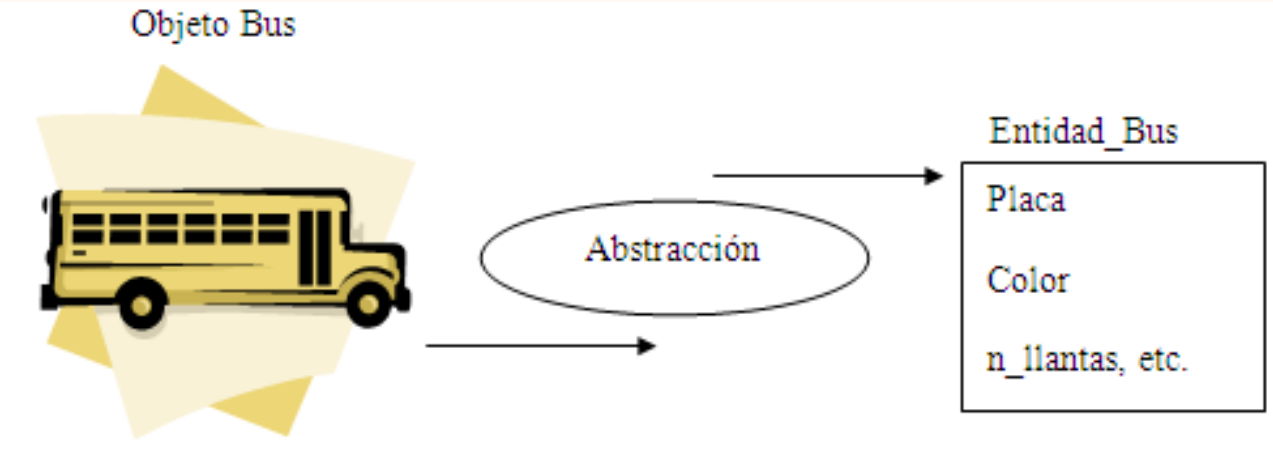
Tipo de Dato Abstracto



Dra. Yolanda Moyao Martínez

DEFINICIÓN DE ABSTRACCIÓN DE DATOS

- Modelo matemático
- Expresa características esenciales de un objeto, las cuales distinguen al objeto de los demás.
- Se enfoca a "¿que hace?" mas que en "¿Cómo lo hace?"



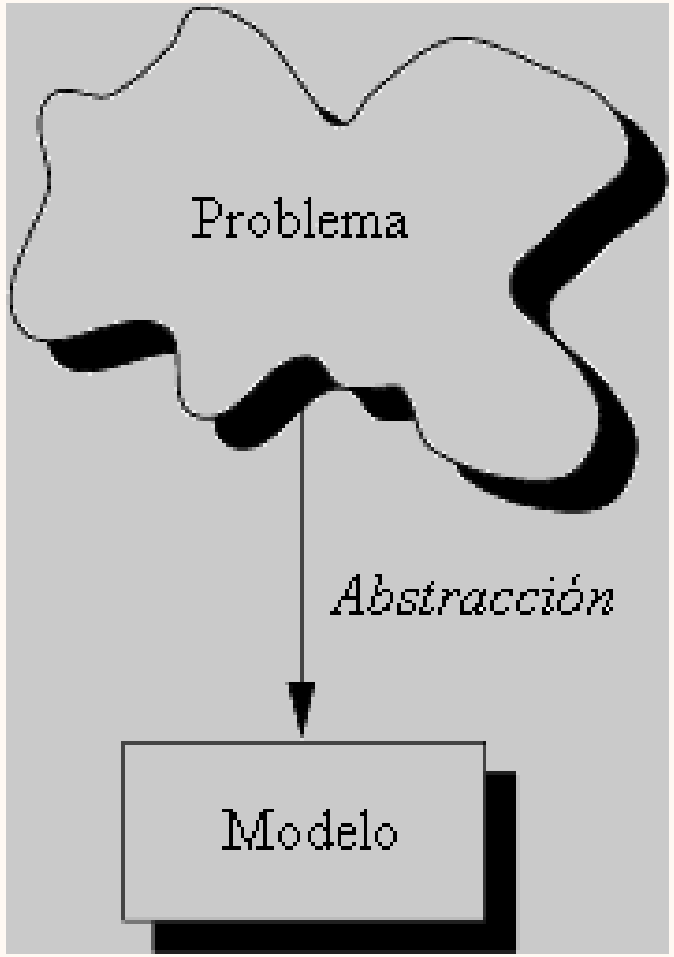
Fuente: https://www.youtube.com/watch?v=WhTJWLlsAYI

TDA

- Elemento básico de la abstracción de datos.
- Su desarrollo es independiente del lenguaje.
- Formado por un conjunto válido de elementos y un número de operaciones primitivas que se pueden aplicar sobre ellos.
- Se declaran variables del tipo TDA y se opera con ellas

TDA

- Al diseñar una **nueva estructura**, ésta pasa a ser un Tipo de Dato Abstracto
- La **especificación lógica** de un TDA es un documento en el que se **plasma**
- La **abstracción** realizada al diseñar una estructura de datos
- Se convierte en el plano mediante el que se implementará la estructura de datos.

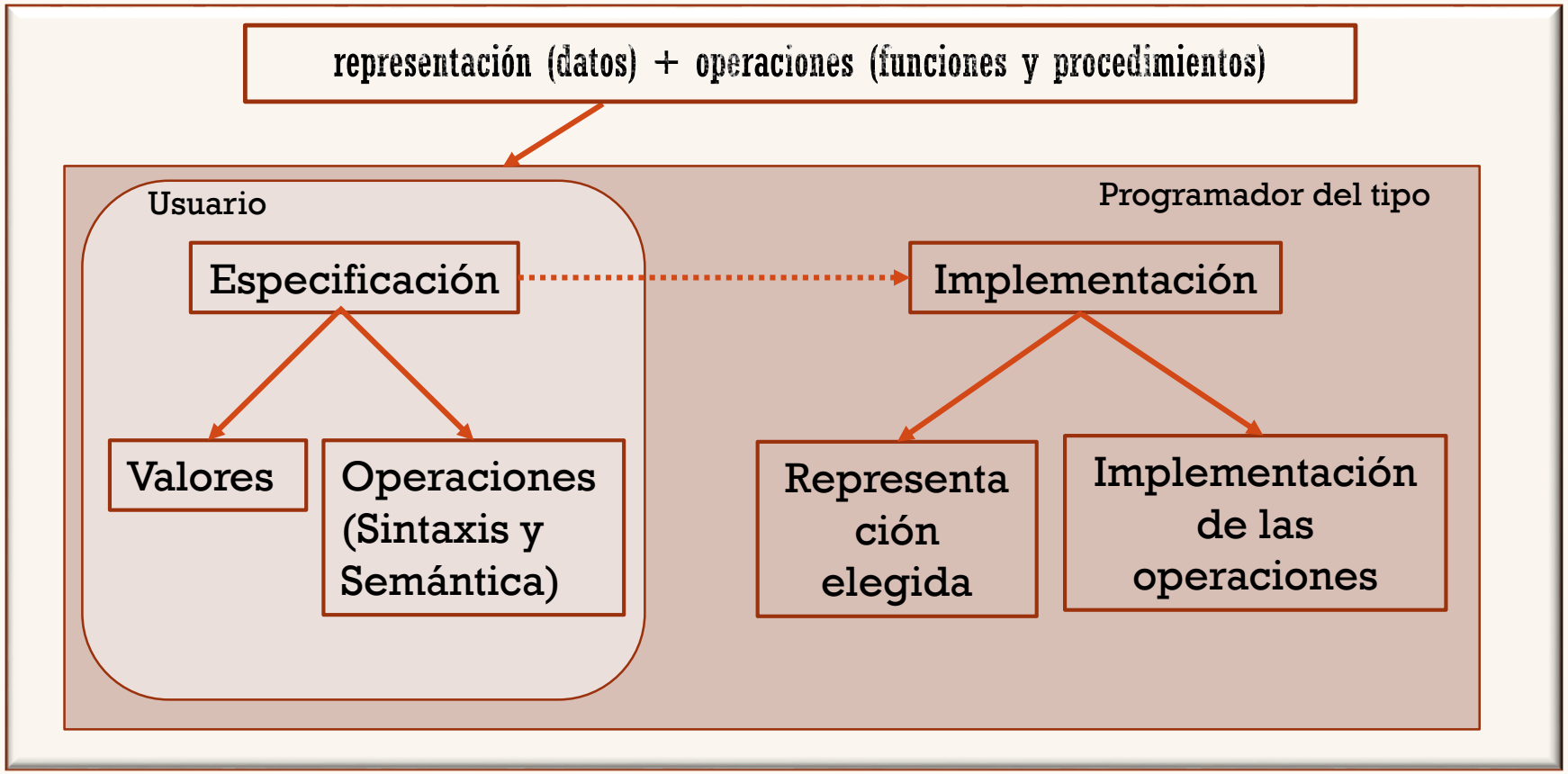


Fuente: https://www.youtube.com/watch?v=WhTJWLlsAYI

CONSTRUCCIÓN DE TDA

1. Definir una perspectiva abstracta del problema.
2. Enfocar el modelo solo a aspectos relacionados con el problema.
3. Tratar de definir propiedades del problema.
4. Las propiedades incluyen:
   - Datos que son afectados
   - Operaciones identificadas

DISEÑO TAD



Fuente: https://slideplayer.es/slide/1612981/

EJEMPLO: TDA PARA REPRESENTAR UN NÚMERO RACIONAL

- DESCRIPCIÓN: Los valores del TDA racional son números racionales.
- OPERACIONES:
  1. crea(a,b:entero) devuelve (Racional), requerimientos: b<>0.

     efecto: Devuelve un número racional cuyo numerador es a y cuyo denominador es b.
  2. num(a:Racional) devuelve (entero)

     efecto: Devuelve el numerador del número racional a.
  3. den(a:Racional) devuelve (entero)

     efecto: Devuelve el denominador del número racional a.
  4. suma(a,b:Racional) devuelve (Racional)

     efecto: Devuelve un número racional que es la suma de los números racionales a y b.

5. resta(a,b:Racional) devuelve (Racional)
   efecto: Devuelve un número racional que es la resta de los números racionales a y b.
6. multiplica(a,b:Racional) devuelve (Racional)
   efecto: Devuelve un número racional que es la multiplicación de los números racionales a y b.
7. divide(a,b:Racional) devuelve (Racional)
   efecto: Devuelve un número racional que es la división de los números racionales a y b.
8. simplifica(a:Racional) devuelve (Racional)
   efecto: Devuelve un número racional que es la simplificación del número racional a.

TDA FORMAL

- Definición del tipo
- racional: Conjunto de pares de elementos (a , b) de tipo entero, con b<>0.
- Operaciones:
  1. CrearRacional: a, b = (a , b) (a ,b:entero) -> racional
  2. Numerador: (a , b) = a
  3. Suma: (a , b) + (c , d) = (a*d+b*c , b*d) → racional
  4. Resta: (a , b) - (c , d) = (a*d-b*c , b*d) → racional
  5. Producto: (a , b) * (c,d) = (a*c , b*d) → racional
  6. División: (a , b) / (c,d) = (a*d , b*c) → racional
  7. Simplifica: (a , b) = (a/mcd(a , b) , b/mcd(a , b)) → racional

USANDO EL TDA

- racional v1,v2, resultado;
- CrearRacional(6,9,&v1);
- CrearRacional(2,5,&v2);
- Suma(v1,v2,&resultado);

EJEMPLO: DEFINE TDA NÚMERO COMPLEJO

- Definición del tipo

NumeroComplejo es un par ordenado tal que si: $z \in \mathbb{C}$ entonces

$z = a + bi$, con a y b $\in \mathbb{R}$

CreaComplejo

- Parte Real $\mathbb{R}(z) = a$
- Parte Imaginaria $I(z) = b$

- Operaciones

1. Crea: Crea un número complejo
   se realiza de la siguiente manera: z = a + bi, donde a es Real y bi es imaginario
3. Conjugación: Se realiza de la siguiente manera: z' = a − bi
4. Suma: la suma de 2 números complejos z = a + bi y w = x + yi
   se realiza de la siguiente manera: z + w = (a + x) + i(b + y)
5. Resta: Se realiza de la siguiente manera: z − w = (a − x) + i(b − y)
6. Multiplicación: Se realiza de la siguiente manera: z · w = (ax − by) + i(ay + bx)
7. División: Se realiza de la siguiente manera:

$$\frac{a+bi}{c+di}=\frac{(a+bi)\cdot(c-di)}{(c+di)\cdot(c-di)}=\frac{(ac+bd)+(bc-ad)i}{c^2+d^2}=\frac{ac+bd}{c^2+d^2}+\frac{bc-ad}{c^2+d^2}i$$

8. Módulo: Se realiza de la siguiente manera:

$$r = |Z| = \sqrt{a^2 + b^2}$$

USANDO EL TDA

NumeroComplejo : tNumeroComplejo
Crea(var unNumero :tNumeroComplejo ; real,imaginario : REAL)
Conjugación(var unNumero :tNumeroComplejo):tNumeroComplejo
Módulo(var unNumero :tNumeroComplejo):Real
Suma numeros(var unNumero, otroNumero
:tNumeroComplejo):tNumeroComplejo
Resta numeros(var unNumero, otroNumero
:tNumeroComplejo):tNumeroComplejo
Multiplicación numeros(var unNumero, otroNumero
:tNumeroComplejo):tNumeroComplejo
División numeros(var unNumero, otroNumero
:tNumeroComplejo):tNumeroComplejo

TDA: CADENA

- **Elementos**: ASCII
- **Estructura**: Relación lineal entre los caracteres
- **Dominio**: Existen entre 0 y 255 caracteres en cada valor de TDA.
- El dominio son todas las secuencias de caracteres que cumplan con las reglas

‘E’	**‘s’**		**‘c’**	**‘a’**	**‘d’**	**‘e’**	**‘n’**	**‘a’**	**‘\0’**

OPERACIONES

- **Borra_Inicio**
  - **Utilidad: Elimina el primer carácter**
  - **Entrada: Cadena S**
  - **Salida: Carácter mas a la izquierda de S**
  - **Precondición: longitud de S mayor que 0**
  - **Postcondición: S tiene todos los caracteres menos el primero**

- **Agrega_Final**
  - **Utilidad: Agrega un carácter al final de la cadena**
  - **Entrada: Cadena S y carácter L**
  - **Salida: Cadena S modificada**
  - **Precondición: longitud de S es menor que 255**
  - **Postcondición: S tiene el carácter L que queda al extremo derecho de S**

- **Vacía**
  - **Utilidad: Verifica si una cadena está vacía o no.**
  - **Entrada: Cadena S**
  - **Salida: Verdadero o Falso**
  - **Precondición: Ninguna**
  - **Postcondición: Ninguna(pues S no se modifica)**

- **Llena**
  - **Utilidad: Verifica si una cadena está llena o no.**
  - **Entrada: Cadena S**
  - **Salida: Verdadero o Falso**
  - **Precondición: Ninguna**
  - **Postcondición: Ninguna(pues S no se modifica)**

- **Invierte**
  - **Utilidad: Invierte el orden de caracteres en S.**
  - **Entrada: Cadena S.**
  - **Salida: Cadena S modificada.**
  - **Precondición: Ninguna.**
  - **Postcondición: Cadena S modifica, es decir el primer carácter toma el lugar del último y así sucesivamente.**

EJEMPLO:

- Tipo Bolsa (Elemento)
- Sintaxis
  1. bolsavacia → Bolsa
  2. poner(Bolsa,Elemento) → Bolsa
  3. esvacia(Bolsa) → booleano
  4. cuantos(Bolsa,Elemento) → natural
- Semántica ∀ b ∈ Bolsa, ∀ e,f ∈ Elemento:
  1. esvacia(bolsavacia) cierto
  2. esvacia(poner(b,e)) ⇒ falso
  3. cuantos((bolsavacia),e) ⇒ cero

EJEMPLO: ADMINISTRACION DE EMPLEADOS

- Los empleados caracterizadas por muchas propiedades
- Solamente algunas propiedades son *específicas del problema*

DATOS

- nombre,
- fecha de nacimiento,
- número social,
- Sueldo

×número de cuarto,

×color de pelo,

×peso

×pasatiempos.

OPERACIONES

- Crear un empleado
- Contratar
- Despedir
- AumentarSueldo

PRÁCTICA 1

1. Diseña un TDA Vector que describa operaciones con vectores
2. Diseña un TDA para matriz que describa operaciones con matrices
3. Diseña un TDA para cuenta de cheques que describa operaciones tales como: alta, depósito, etc.

TAREA 1

- Instalar el lenguaje UML
- Aprender a utilizar UML

Definición de un TDA

TDA se caracteriza por las siguientes propiedades:

1. Exporta un tipo.
2. Exporta un conjunto de operaciones. Este conjunto es llamado interface.
3. Las Operaciones de la interface son el único mecanismo de acceso a la estructura de datos del tipo.
4. Axiomas y precondiciones definen el dominio de la aplicación del tipo.

Especificación Sintáctica

- Qué hace? Especificación de las entidades y sus propiedades.

- Definir el nombre de las entidades abstractas.

- Definir el nombre de las operaciones indicando el dominio (argumentos) y el co-dominio o rango (los valores de retorno).

Especificación Semántica

- Cómo lo hace? Descripción de la representación del objeto (estructuras de los datos) y desarrollo de las operaciones.
- Definir el significado de cada operación usando los símbolos definidos en la especificación sintáctica.
- La especificación puede ser de dos tipos:
  - Informal, a través del lenguaje natural.
  - Formal, rigurosa y fundamentada matemáticamente.

Especificación Semántica

- Además:
  - {P} Pre-condición: condiciones que deben cumplirse antes de realizar la operación.
  - {Q} Post-condición: condiciones que se cumplen una vez realizada la operación.

- La notación usual es {P} S {Q}, donde S es la función o procedimiento.

Especificación de un TDA

- La especificación de un TDA consiste en establecer las propiedades que lo definen.
- Para describir un TDA es necesario describir:
  - Los valores que pueden tomar los datos de ese tipo.
  - Todas las operaciones realizables sobre de ellos.
- Una especificación debe poseer 4 propiedades:
  - Ser precisa: Solo dice lo imprescindible.
  - Ser general: Es adaptable a diferentes contextos.
  - Ser legible: Transmite a los usuarios del tipo y al implementador el comportamiento del tipo.
  - No ambigua: Evita dobles interpretaciones.

Ejemplo: TDA Bolsa

- **Definición:** Es una colección no ordenada de elementos con repetición.
- **Tipo:** Bolsa.
- **Sintaxis:**
  CrearBolsa () → Bolsa
  BolsaVacia → Bolsa
  BolsaLlena → Bolsa
  Poner (Bolsa, Objeto) → Bolsa
  EsVacia (Bolsa) → Boolean
  Retirar (Bolsa, Objeto) → Objeto
- **Semántica:** b es Bolsa, e, f son elementos
  CrearBolsa () = BolsaVacia
  EsVacia (CrearBolsa()) = Verdadero
  EsVacia (Poner(CrearBolsa(), e)) = Falso
  Retirar (BolsaVacia, e) = Error
  Retirar (Poner(CrearBolsa(), f), e) = f si f=e
  Poner (BolsaLlena, e) = Error

Ejemplo: TDA Bolsa

- **Definición:** Es una colección no ordenada de elementos con repetición.
- **Tipo:** bolsa.
- **Operaciones:**

  Función **Construir_Bolsa** () → bolsa

  {postcondición: Devuelve una bolsa vacía}

  Función **Poner** (B: bolsa; e: elemento) → bolsa

  {precondición: La bolsa no esta llena}

  {postcondición: Añade el elemento e a la bolsa}

  Función **EsVacia** (B: bolsa) → boolean

  {postcondición: Devuelve verdadero si la bolsa no tiene elementos, falso en otro caso}

  Función **Retirar** (B: bolsa; e: elemento) → elemento

  {precondición: La bolsa no esta vacía}

  {postcondición: Elimina el elemento e de la bolsa B}