

Lenguajes de Programación

Paradigmas

Ma. Laura Cobo

Universidad Nacional del Sur
Departamento de Ciencias e Ingeniería de la Computación
2018

Lenguajes y Paradigmas

Metodología de diseño

es un conjunto de métodos y pautas que guían el proceso de desarrollo

Paradigma de Programación

- Conjunto coherente de métodos para resolver un problema
- Tiene principio básico
- Guía el proceso de desarrollo del software

colección de patrones conceptuales integrados que orientan el proceso de desarrollo de software y determinan la estructura de un programa válido

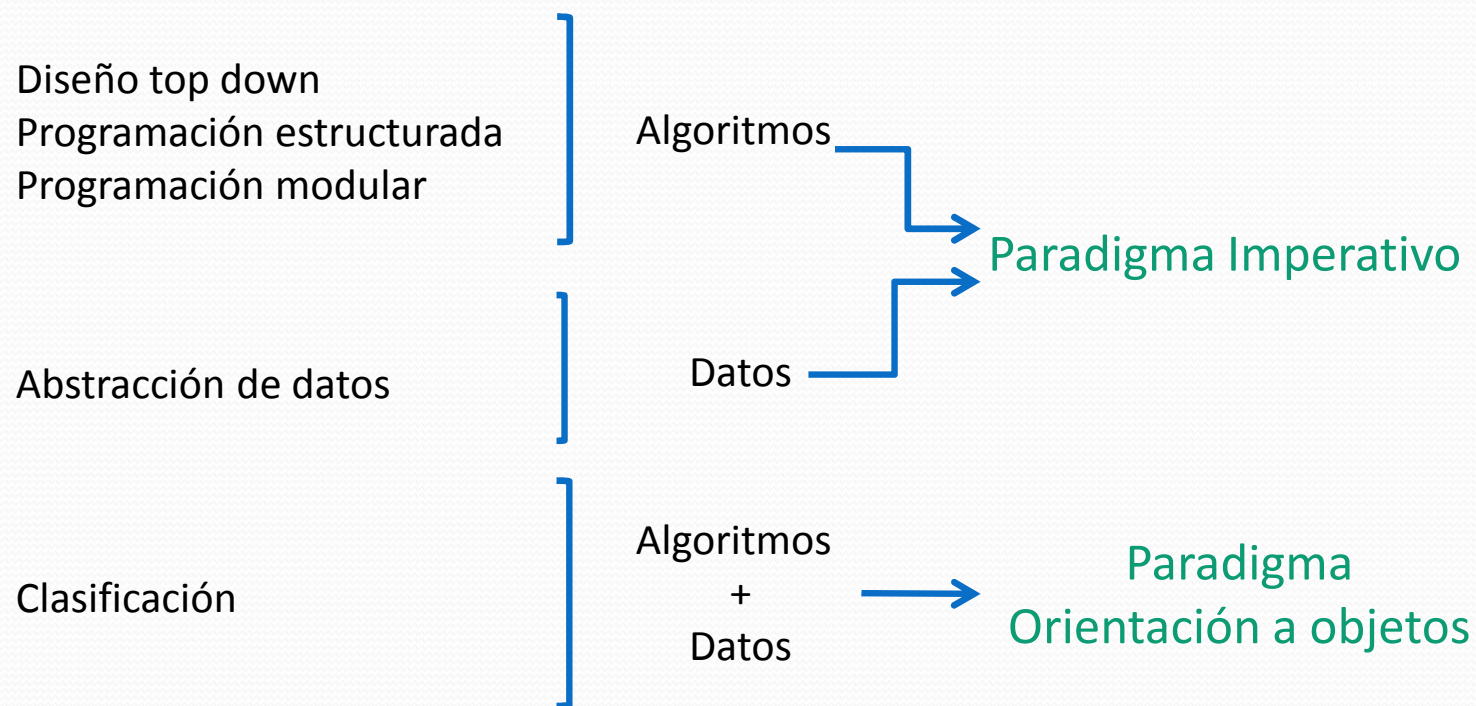
Lenguajes y Paradigmas: “soportar” o “admitir”

Un lenguaje “soporta” un paradigma si provee mecanismos que facilitan su implementación eficiente. Es decir, implementa mecanismos que permiten la sencilla aplicación del paradigma (impone restricciones para respetarlo)

Un lenguaje “admite” un paradigma si es posible escribir programas siguiendo los lineamientos del paradigma, pero hacerlo demanda un esfuerzo notable

El lenguaje no exige la aplicación del paradigma, permite programar de acuerdo al paradigma pero sin proveer facilidades.

Lenguajes y Paradigmas: ¿focalizando en acciones o datos?



Paradigma Imperativo

Un programa es una secuencia de **instrucciones** que indican el flujo de la ejecución.

Señal dada para que se realice
el cambio de estado del
autómata

Las principales características son:

1. ejecución secuencial de instrucciones
2. uso de variables representando valores de locaciones de memoria
3. uso de sentencias de asignación para cambiar los valores de las variables, permitiendo así al programa operar sobre las locaciones de memoria

Paradigma Imperativo

Se busca estructurar el control realizando una programación estructurada y modular con abstracción de datos para fomentar la reusabilidad y extendibilidad

Los programas se construyen siguiendo una aproximación:

- Top-down
 - Modular
- Solo sub-programas
 - Dividir para conquistar
 - Existen abstracciones algorítmicas

Abstracción a nivel instrucción
Agrupar instrucciones en unidades –
Procedure de Pascal

Abstracción de expresiones
Function de Pascal

Paradigma Imperativo

Los programas se construyen siguiendo una aproximación: **Modular**

○ Programación Estructurada

- estructuras de control a nivel instrucción
- inhibición del uso de estructuras de control

○ Programación Modular

Descomponer el problema priorizando
recombinación y reutilización en otros problemas.
El **como** descomponerlo es ingerencia de la ing. De
software



Modulo

con

Independencia de significado
Independencia de implementación

○ Abstracción de datos

- Reconocer entidades abstractas
(hallar representación para los datos)
- Reconocer operaciones lógicas
(transformar en operaciones concretas)

Paradigma Imperativo - Limitaciones

- Dificultad para razonar
 - Alta dependencia de la arquitectura
 - una variable es el de abstracción de celda de memoria
- Transparencia referencial
 - Asignaciones
- Efectos colaterales y aliasing
- Accesibilidad vulnerable
- Estructuras de control : desdibujan la lógica del programa creando problemas de mantenimiento y seguridad

Paradigma Orientado a Objetos


- Se caracteriza por reconocer las entidades del problema (similar a la abstracción de datos)
- Entidad = **Objetos**
- Comunicación por **mensajes**, diferente a la semántica de llamadas a procedimiento

Caracterizado por atributos y comportamiento
(de acuerdo a su propósito y habilidades)

Objetivos:

- Mejorar la **reusabilidad** del software
- Mejorar la **extendibilidad** del software

Para algunos



- Quiebre total con lo anterior
- Nuevo eslabón de la línea estructurada

Paradigma Orientado a Objetos

Los lenguajes orientados a objetos deben proveer:

- Comunicación por mensajes
- Concepto de objeto con estado interno y servicios
- Soportar clasificación

El mínimo para los diferentes autores es:

- Abstracción de datos + herencia
+ encapsulamiento
+ polimorfismo

Para nosotros la abstracción de datos y el encapsulamiento son conceptos que deben ir juntos.

Para algunos autores polimorfismo es caso particular de herencia y para otros polimorfismo, herencia y sobrecarga son conceptos diferentes

Otros autores requieren además ligadura dinámica y otros soporte de concurrencia (aunque esto último no es inherente al paradigma)

Paradigma no convencionales

Las formalizaciones abstractas, como las máquinas de Turing, el cálculo lambda, etc., brindan formas de expresión y razonamiento que permiten obtener construcciones útiles y adecuadas para descubrir problemas

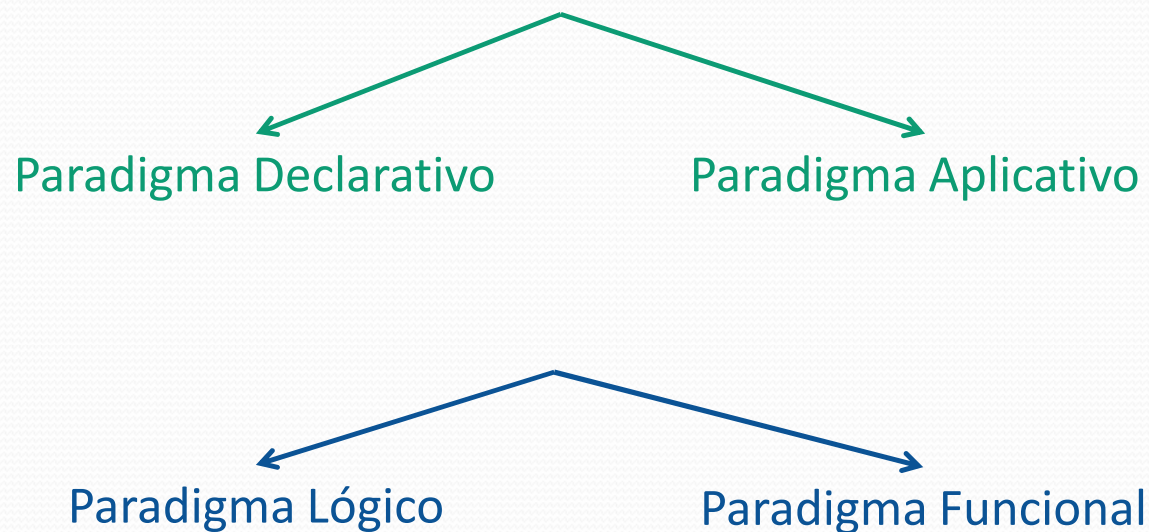
Redescubrir los formalismos como una forma resolver problemas, utilizándolos en el diseño y la implementación

Reciben el nombre de no convencionales porque su forma de especificar y ejecutar están “alejadas” de la arquitectura

Paradigma no convencionales

Alejados totalmente del control del paradigma imperativo.

La lógica y la matemática nos brindan formas de expresión como:



Paradigma Declarativo

Un programa establece un conjunto de propiedades que describen como alcanzar la solución. En ningún caso se indican cómo se computa

Un lenguaje declarativo, brinda las facilidades para especificar estas propiedades y una forma de computar que sea transparente

El principal problema es la eficiencia

La ventaja es que los programas que se obtienen son elegantes y concisos

Paradigma Lógico

Basado en lógica de primer orden (lenguaje preciso para expresar conocimiento)

Lenguaje representativo: ProLog

Los fundamentos del paradigma son:

- a) Deducir consecuencias a partir de premisas
- b) Estudiar o decidir el valor de verdad de una sentencia a partir del valor de verdad de otras
- c) Establecer la consistencia entre hechos y verificar la validez de argumentos

Paradigma Lógico

Características de los lenguajes lógicos

- a) Eliminación del control
- b) El concepto de variable es más matemático, son nombres que retienen valores
- c) Establecen “que” es lo que se debe hacer sin dar ninguna especificación sobre el “como” hacerlo

Características de los programas lógicos

- a) Conjuntos de axiomas que establecen relaciones
- b) Definen un conjunto de consecuencias que determinan el significado
- c) Son teoremas y la ejecución es una prueba automática

Paradigma Aplicativo

Basado en el uso de funciones, de ahí que habitualmente se hable de él como paradigma funcional. Muy popular en la resolución de problemas de inteligencia artificial, matemática, lógica, procesamiento paralelo

Ventajas:

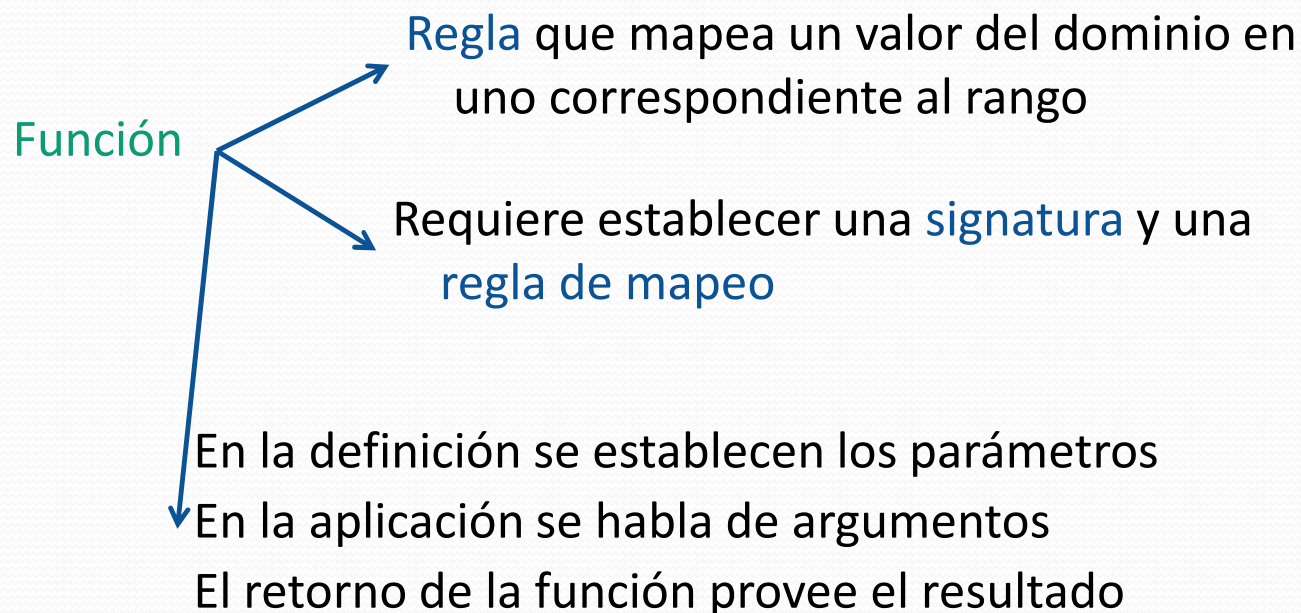
- Vista uniforme de programa y función
- Tratamiento de funciones como datos
- Liberación de efectos colaterales
- Manejo automático de memoria

Desventaja:

- Ineficiencia de ejecución

Paradigma Funcional

La esencia de esta metodología esta en componer funciones para definir otras más complejas.



Paradigma Funcional

Características de los lenguajes funcionales

- a) Define un conjunto de datos
- b) Provee un conjunto de funciones primitivas
- c) Provee un conjunto de formas funcionales
- d) Requiere de un operador de aplicación

Características de los programas funcionales

- a) Semántica basada en valores
- b) Transparencia referencial
- c) Regla de mapeo basada en combinación o composición
- d) Las funciones son “ciudadanos” de primer orden

Integración de paradigmas

Aparecen los lenguajes híbridos, principalmente debido a necesidades heterogéneas y la interoperabilidad disponible.

Lenguajes Script

Motivación

La programación tradicional construye aplicaciones auto-contenidas. Actualmente se espera que las computadoras coordinen múltiples programas.

Un ejemplo se da en la creación de páginas web dinámicas, se requiere autenticación, autorización, acceso a base de datos, manipulación de imágenes, leer y escribir texto HTML.

Es posible escribir código de coordinación en lenguajes convencionales (Java, C, etc.) pero no siempre es fácil. Persigue eficiencia, mantenabilidad, portabilidad y detección estática de errores

Lenguajes Script

Motivación

Los lenguajes convencionales tienden a mejorar eficiencia, mantenibilidad, portabilidad y detección estática de errores.

Los tipos se construyen alrededor de conceptos a nivel hardware como enteros de tamaño fijo, punto flotante, caracteres y arreglos.

Los lenguajes script tienden a mejorar flexibilidad, desarrollo rápido y chequeo dinámico.

Su sistema de tipos se construye sobre conceptos de mas alto nivel como tablas, patrones, listas y archivos.

Lenguajes Script

Los lenguajes script de propósito general (Perl, Python) suelen conocerse como *glue-languages*.

Se diseñaron para “pegar” programas existentes a fin de construir un sistema mas grande.

Se utilizan como lenguajes de extensión, ya que permiten al usuario adaptar o extender las funcionalidad de las herramientas script.

¿qué es un lenguajes Script?

Es difícil definirlos con precisión, aunque hay varias características que tienden a tener en común

Estos lenguajes tienen dos tipos de ancestros:

- interpretes de líneas de comando o “shells”
- herramientas para procesamiento de texto y generación de reportes.

“Los lenguajes script asumen la existencia de componentes útiles en otros lenguajes. Su intención no es escribir aplicaciones desde el comienzo sino por combinación de componentes” (John Ousterhout – creador de TCL)

Características comunes

Uso “batch” e interactivo: pocos lenguajes script usan compilación *just-in-time*. En general compilan o interpretan línea por línea

Economía de expresión: soportan desarrollo rápido y uso interactivo. Tratan de evitar declaraciones extensas y las estructuras propias de los lenguajes convencionales.

Falta de declaraciones; reglas de alcance simple: en muchos lenguajes los nombres son globales o locales por defecto

Tipado dinámico flexible: se chequea el tipo antes de su uso

Características comunes

Acceso simple a las facilidades del sistema: proveen una manera de pedir al sistema operativo la ejecución de otro programa o realizar una operación directamente.

Pattern-matching sofisticado y manipulación de strings: uso directo de expresiones regulares extendidas

Tipos de datos de alto nivel: permiten la funciones de tipos de alto nivel con la sintaxis y semántica del mismo lenguaje (por ejemplo contar con un arreglo indexado por caracteres, con la implementación subyacente de la tabla hash)