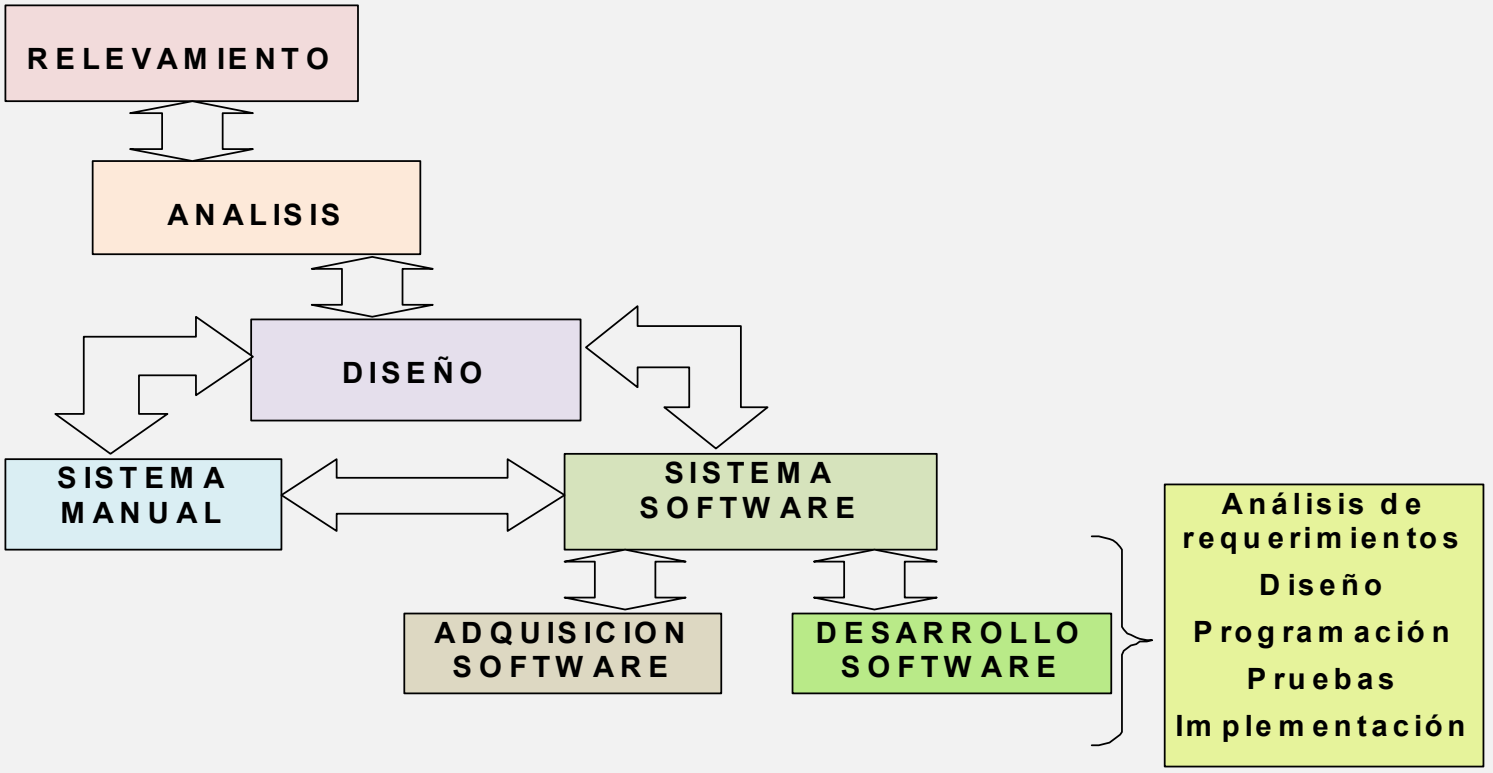
Universidad Tecnológica Nacional
FACULTAD REGIONAL CORDOBA

PARADIGMAS DE PROGRAMACION

Unidad I

Introducción a los Lenguajes y Paradigmas de Programación

Repaso de conceptos: Ciclo Sistémico

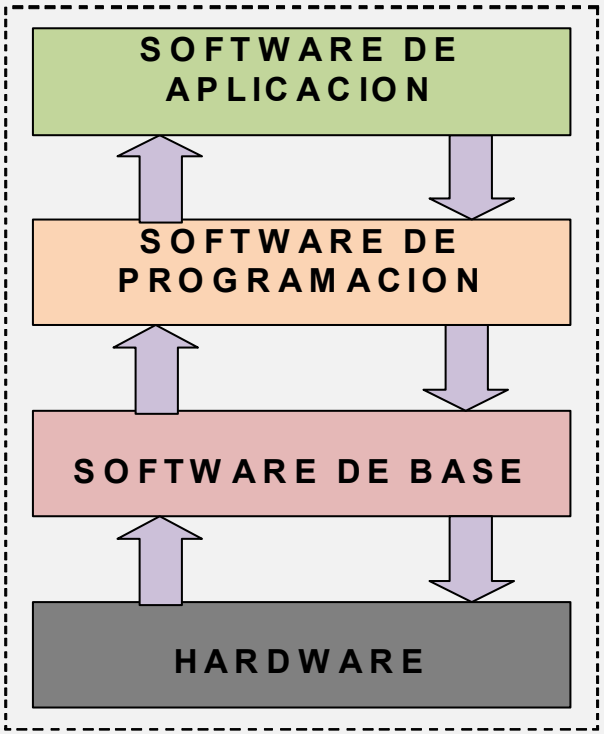


Repaso de conceptos: Software

ALTO NIVEL

MEDIO NIVEL

BAJO NIVEL



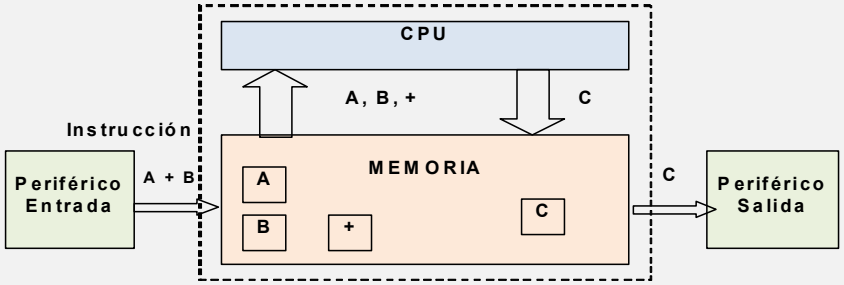
Aplicaciones industriales,
Aplicaciones ofimáticas,
Software educativo,
Software médico,

Editores de texto,
Compiladores,
Intérpretes,
Enlazadores,
Depuradores,

Sistemas operativos,
Controladores de dispositivos,
Servidores.

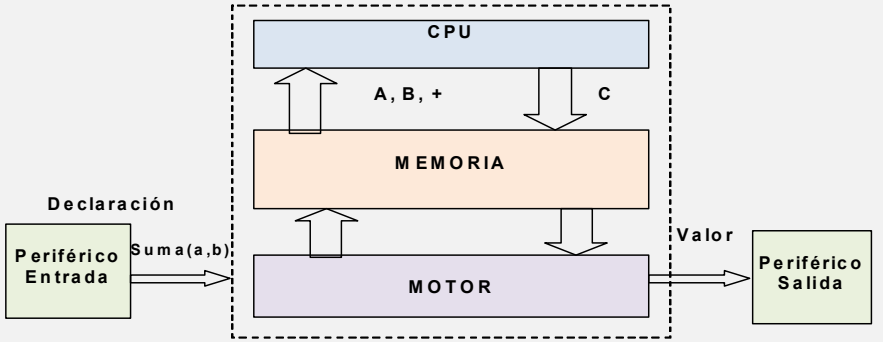
Repaso de conceptos: Modelos

MODELO BASADO EN INSTRUCCIONES



El desarrollador puede administrar el uso de la memoria y la gestión de procesamiento

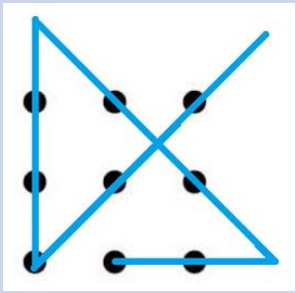
MODELO BASADO EN DECLARACIONES



El desarrollador introduce declaraciones que el motor interpreta.

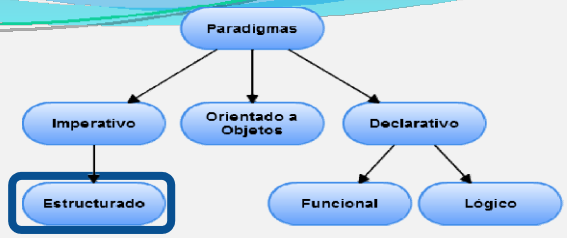
El motor abstrae el uso de memoria y la gestión de procesamiento.

Conceptos de Paradigma

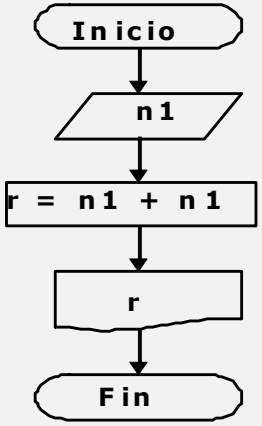
Paradigma	Paradigma de Programación
• Un modelo o ejemplo a seguir por una comunidad científica, de los problemas que tiene que resolver y del modo como se van a dar las soluciones. • Determina una manera especial de entender el mundo, explicarlo y manipularlo. 	• Es un modelo básico de diseño e implementación de programas, que permite desarrollar software conforme a ciertos principios o fundamentos. • Es un marco conceptual que determina los bloques básicos de construcción de software y los criterios para su uso y combinación. • Es una colección de patrones conceptuales (estructuras o reglas). • Determina la forma de pensar y entender un problema y su solución.

Paradigmas fundamentales

Paradigma Estructurado



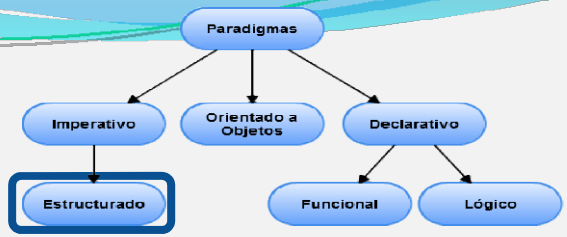
Algoritmo + Estructura de Datos = programa




```
Inicio
 Leer n1
 Procesar r = n1 + n1
 Mostrar r
Fin
```

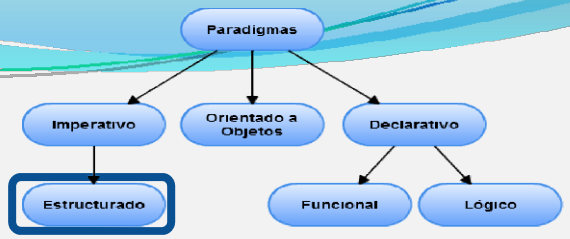
Lenguajes asociados: Fortran, C, Pascal, Python, C++,…

Paradigma Estructurado



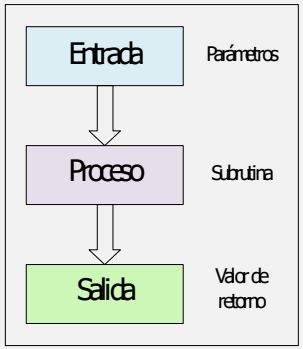
Ventajas	El conjunto de instrucciones del programa es más cercano al conjunto de instrucciones de código de máquina, por consiguiente el código es más directo y es más rápida su ejecución.
Limitaciones	• La complejidad de los sistemas actuales hace que sea más complicado definir la distribución del código solo en módulos. • No se adapta a todos los tipos de problemas.
Aplicaciones	• Sistemas de bajo nivel. • Sistemas de tiempo real. • Programación de micro controladores. • Máquinas de estado. • Desarrollo de video juegos de consola. • Aplicaciones que deben manejar recursos limitados.

Paradigma Estructurado

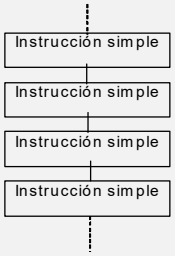


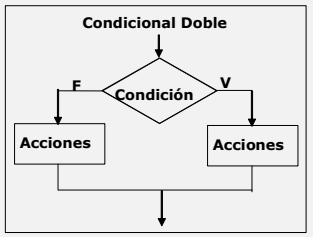
Características

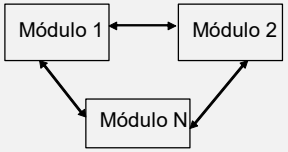
- Tiende a ser orientada a la **acción**, es la unidad fundamental.
- Los programadores se concentran en escribir **procesos**.

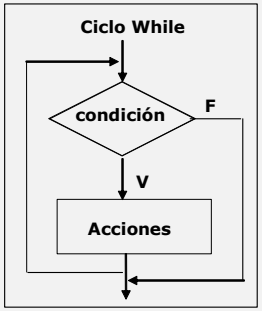


- Variables Locales y Globales.
- Estructuras de control (secuencial, condicional y repetitiva).
- Modularización (funciones y procedimientos).



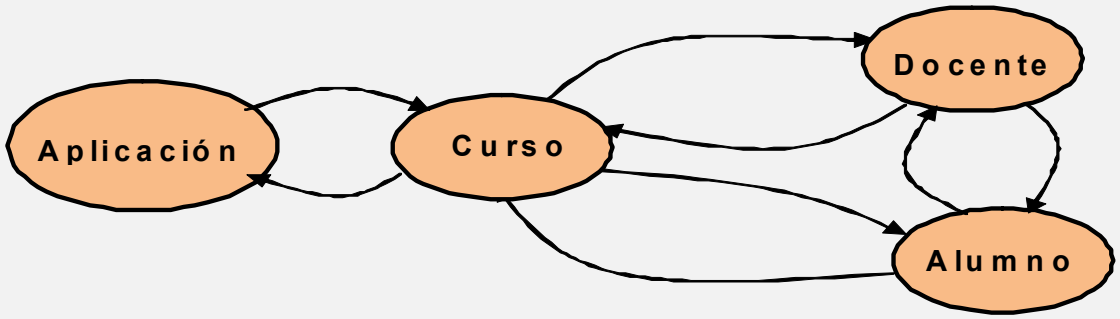






Paradigma Orientado a Objetos

Objetos + Mensajes = Programa



Conjuntos de objetos que colaboran entre si

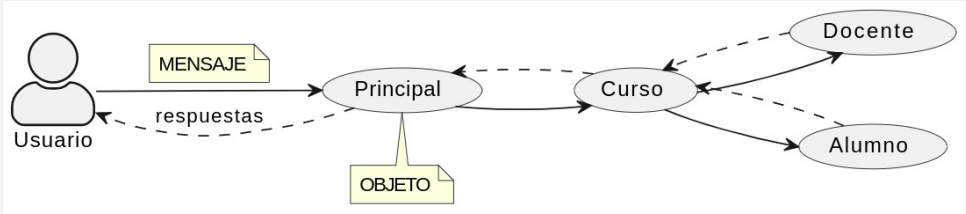
Lenguajes asociados: Smalltalk, Python, Java, C++,…

Paradigma Orientado a Objetos

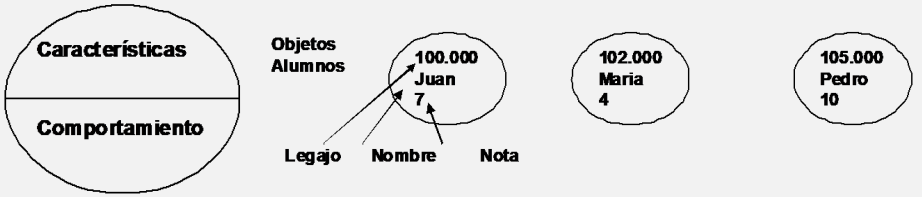
Ventajas	• Facilita el mantenimiento del software. • Permite crear sistemas más complejos. • Agiliza el desarrollo de software. • Facilita el trabajo en equipo.
Limitaciones	• No se adapta a todos los tipos de problemas. • Tamaño de las aplicaciones.
Aplicaciones	• Ampliamente usado en aplicaciones de negocios y tecnología Web. • Modelado y simulación. • Bases de datos orientadas a objetos. • Programación en paralelo. • Sistemas grandes que requieren la administración de estructuras complejas.

Paradigma Orientado a Objetos

- Los programas se organizan como colecciones de objetos que colaboran entre sí enviándose mensajes.
- Cada objeto tiene un rol específico en el programa y todos pueden comunicarse entre sí de formas predefinidas.



- Un OBJETO es una entidad de la vida real que posee características (atributos) y comportamientos (métodos).



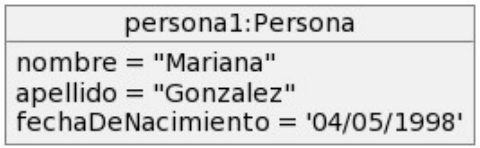
Paradigma Orientado a Objetos

CLASE:

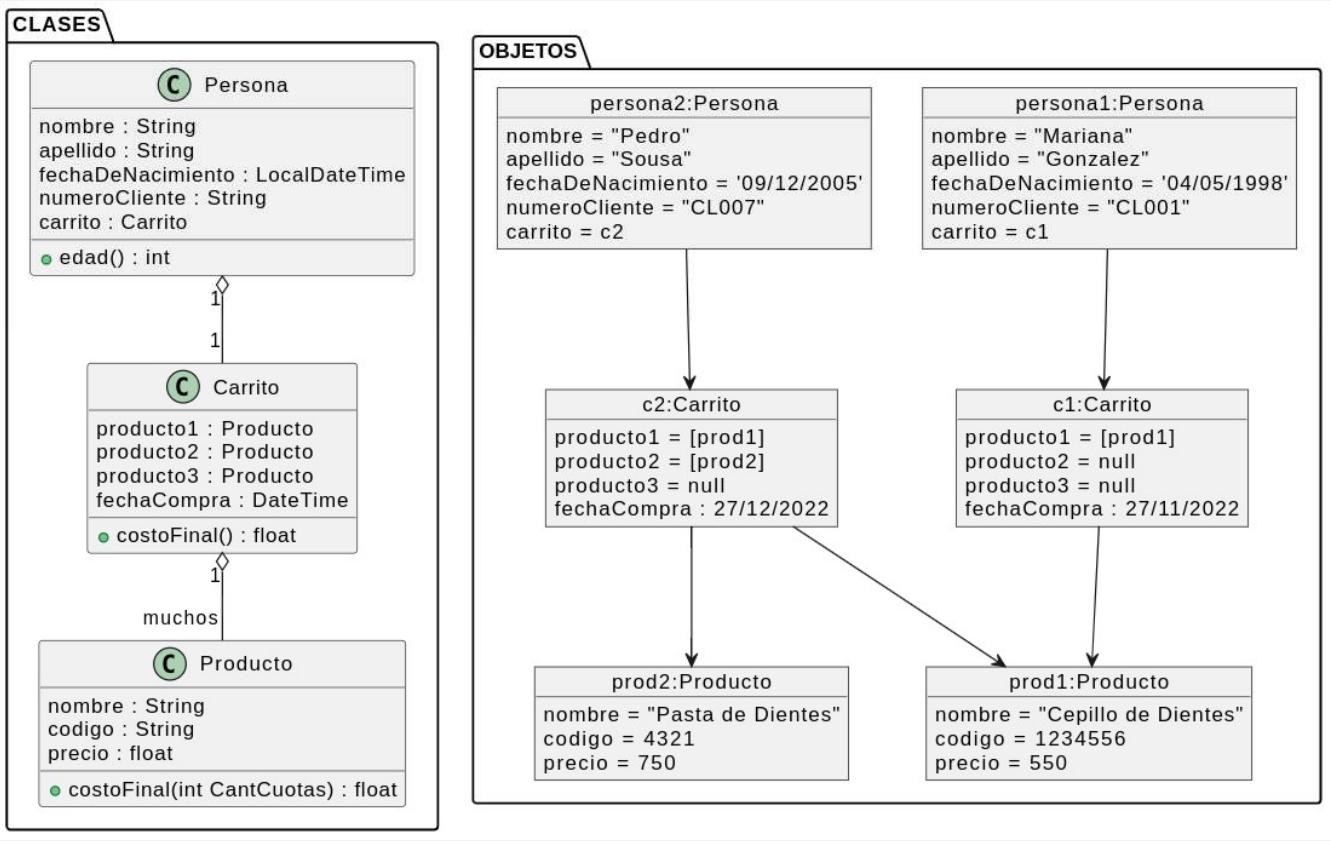
- Es un molde o plantilla que define las características o atributos (variables) y comportamientos o métodos (funciones) comunes de un conjunto de objetos relacionados.
- Describe cómo deben ser los objetos que se crean a partir de ella.

OBJETO:

- Es una instancia o ejemplar de una clase.
- Es una entidad concreta que se crea a partir de una clase y tiene sus propios atributos y comportamientos.
- Posee las características generales de la misma, pero con valores concretos.
- Es una variable compuesta por variables y funciones, definida por el usuario.



Paradigma Orientado a Objetos



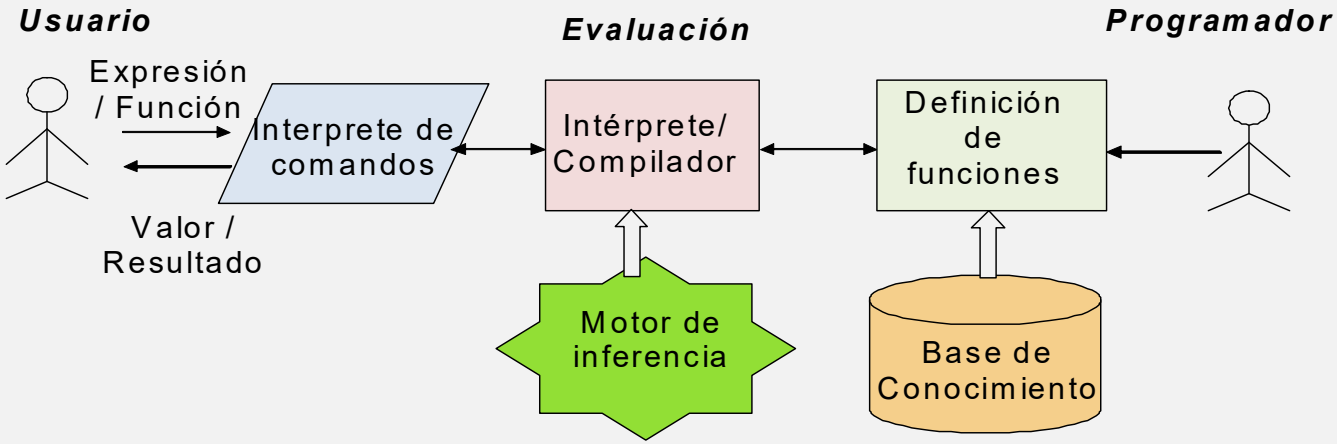
Persona tiene un **Carrito** de compras y el **Carrito** de compras tiene **Productos**

Paradigma Declarativo

- Está basado en el desarrollo de programas especificando o "declarando" un conjunto de condiciones, proposiciones, afirmaciones, restricciones, ecuaciones o transformaciones que describen el problema y detallan su solución.
- La solución es obtenida mediante mecanismos internos de control, sin especificar exactamente cómo encontrarla.

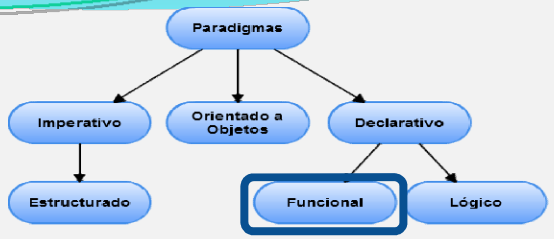
Paradigma Funcional

Funciones + control = programa



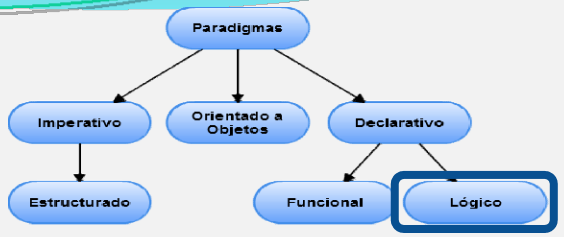
Lenguajes asociados: Lisp, Haskell, ML

Paradigma Funcional

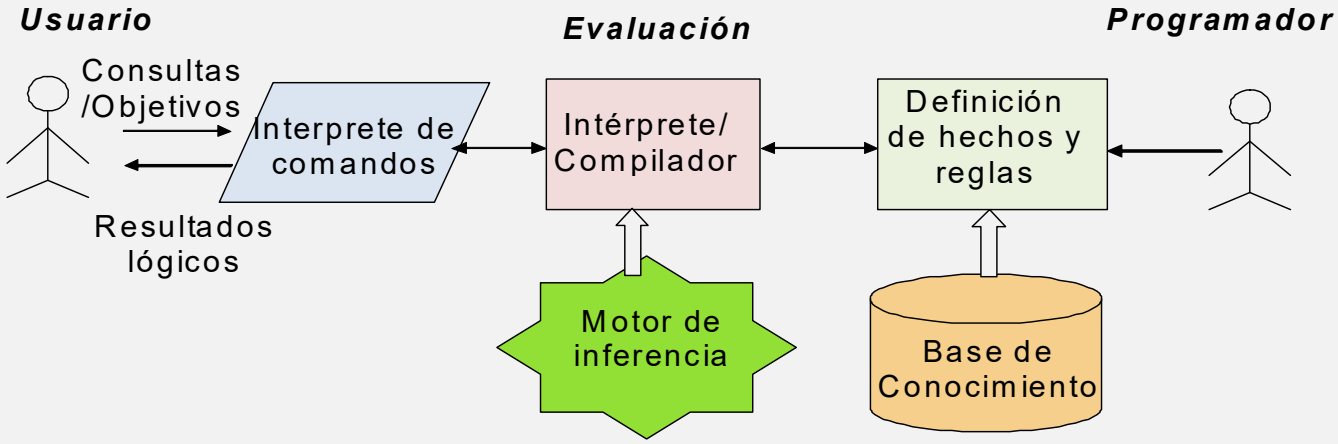


Ventajas	• Fácil de formular matemáticamente. • Administración automática de la memoria. • Simplicidad en el código. • Rapidez en la codificación de los programas.
Limitaciones	• No es fácilmente escalable. • Difícil de integrar con otras aplicaciones. • No es recomendable para modelar lógica de negocios o para realizar tareas de índole transaccionales.
Aplicaciones	• Programas de Inteligencia Artificial. • Sistemas distribuidos de control (NCS) tolerante a fallos de software (por ejemplo, control de tráfico aéreo, mensajería instantánea, servicios basados en Web). • Aplicaciones académicas de gran complejidad matemática.

Paradigma Lógico

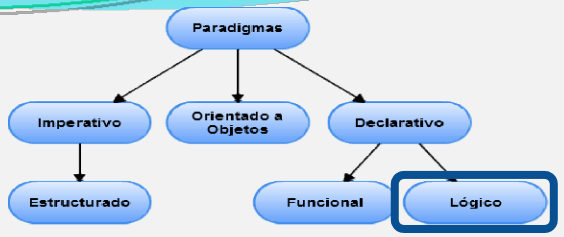


Lógica + control = programa



Lenguajes asociados: Lisp, Prolog, ML

Paradigma Lógico



Ventajas	• Simplicidad en el código. • Cercanía a las especificaciones del problema realizada con lenguajes formales. • Sencillez en la implementación de estructuras complejas.
Limitaciones	• Poco utilizado en aplicaciones de gestión.
Aplicaciones	• Sistemas expertos y robótica. • Sistemas de conocimiento y aprendizaje. • Procesamiento de lenguaje natural. • Sistemas de soporte a decisiones (DSS). • Definiciones de reglas de negocio (Drools)

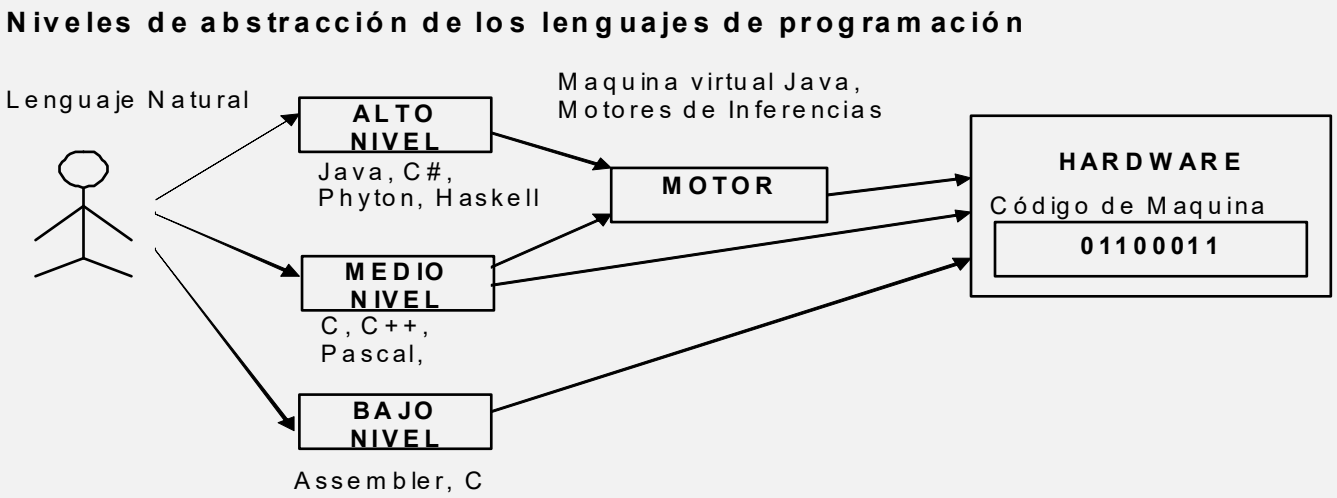
Lenguajes y Paradigmas

- El Paradigma de Programación condiciona la forma en que se expresa la solución a un problema.

- El Lenguaje de Programación (que se encuadra en un determinado paradigma) es la herramienta que permite expresar la solución a un problema.

Lenguaje de programación

- Es una técnica o herramienta estándar de comunicación que permite expresar las instrucciones que han de ser ejecutadas en una computadora.
- Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su gramática y el significado de sus elementos y expresiones.
- Pueden aplicar un paradigma de programación o varios.



Lenguajes de Programación

- Sintaxis, semántica y gramática.
- Criterios de evaluación.
- Reseña histórica y evolución.
- Tipos: Híbridos y puros.