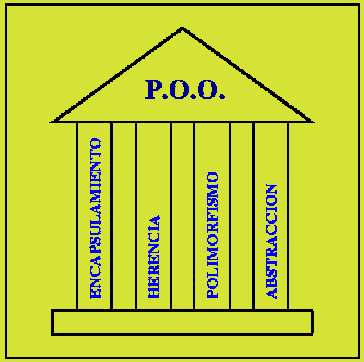


INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS

Eduard Lara

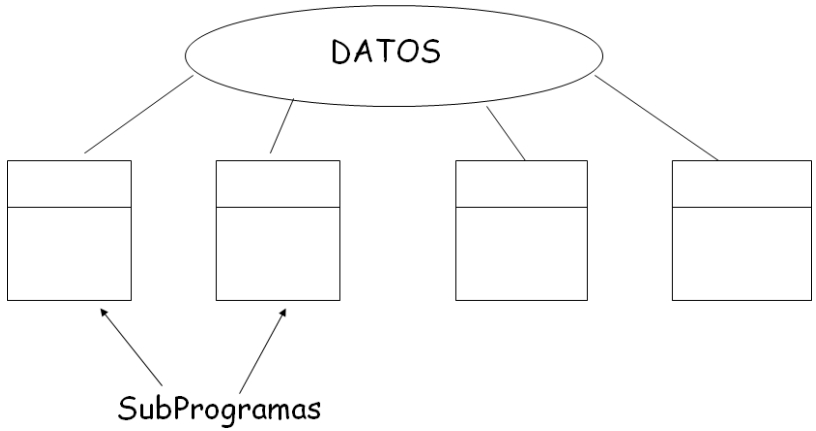
ÍNDICE

1. Evolución histórica
    1. Evolución de las topologías de los lenguajes
    2. Inicios de la orientación a objetos
    3. Lenguajes orientado a objetos
2. Introducción a la orientación a objetos
    1. Programación convencional vs programación orientada a objetos
    2. Beneficios de la OO
3. Ingeniería del software
    1. Nuevo Enfoque OO
    2. Metodologías OO

1. EVOLUCIÓN HISTÓRICA
EVOLUCIÓN DE LAS TOPOLOGÍAS DE LENGUAJES

1º Generación (1954-1958)

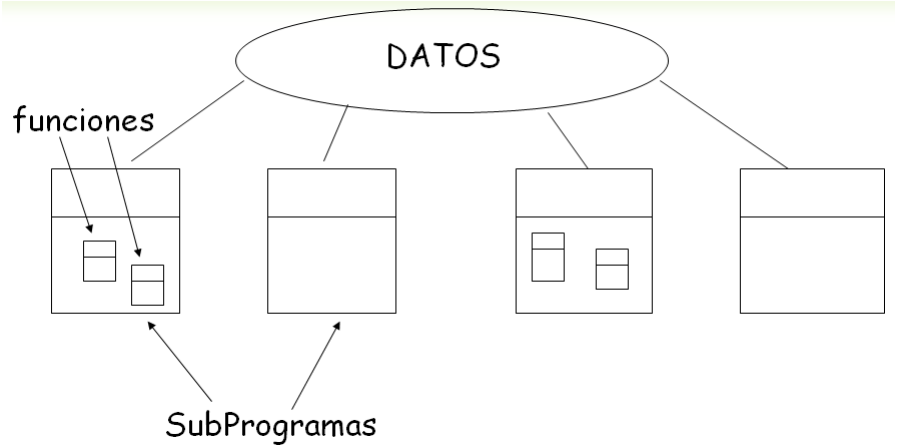
- Los contenedores físicos son SubProgramas.
- Datos globales.
- Los datos eran vistos por todos.
- Un error era devastador.
- FORTRAN, ALGOL 58, COBOL, FLOWMATIC, IPL V.



1. EVOLUCIÓN HISTÓRICA
EVOLUCIÓN DE LAS TOPOLOGÍAS DE LENGUAJES

2º Generación (1958-1962)

- El código finalmente fue reconocido como un punto intermedio entre el problema y la computadora. Y como consecuencia nace la abstracción procedural (funciones).
  - 1. Se desarrollaron mecanismos de paso de parámetros.
  - 2. Se crearon los fundamentos de la programación estructurada.
  - 3. Métodos de diseño para la construcción de grandes sistemas.



1. EVOLUCIÓN HISTÓRICA
EVOLUCIÓN DE LAS TOPOLOGÍAS DE LENGUAJES

2º Generación (1958-1962)

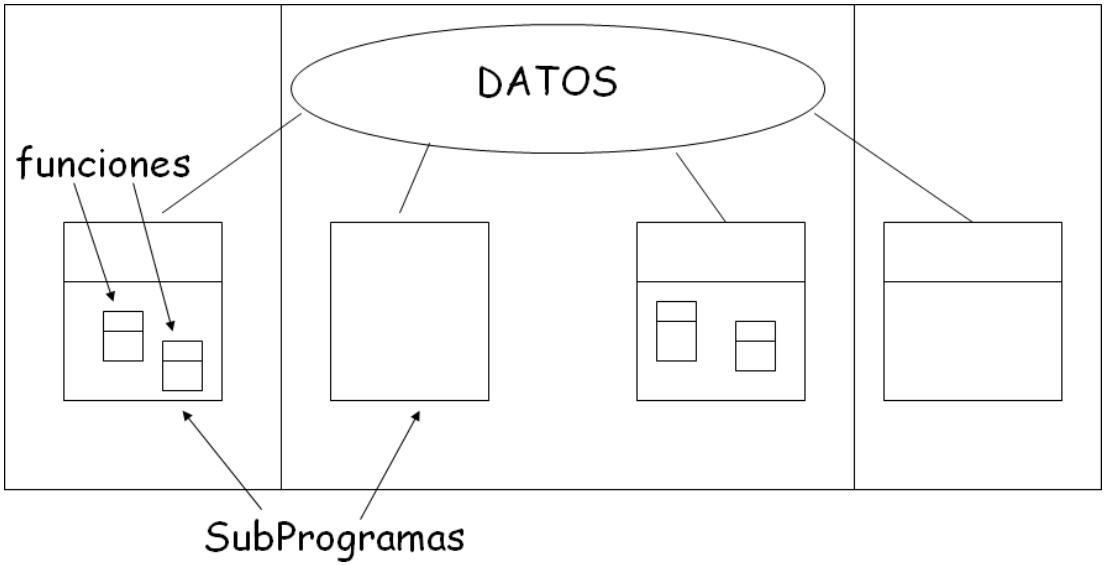
Se desarrollaron lenguajes que:

- Soportan el anidamiento de SubProgramas.
- Estructuras de Control.
- Variables locales y globales.
- FORTRAN II, ALGOL 60, COBOL Y LISP.

1. EVOLUCIÓN HISTÓRICA
EVOLUCIÓN DE LAS TOPOLOGÍAS DE LENGUAJES

3º Generación (1962-1970)

- ❑ Aparecieron los grandes proyectos.
- ❑ Necesidad de desarrollar partes de forma independiente.
- ❑ Aparece la compilación separada lógica.



1. EVOLUCIÓN HISTÓRICA
EVOLUCIÓN DE LAS TOPOLOGÍAS DE LENGUAJES

3º Generación (1962-1970)

Características de los lenguajes

- Soportaban estructura modular.
- Tenían pocas reglas.
- Consistencia semántica, referente a las Interfaces del módulo.
- Pl/1, ALGOL 68, PASCAL, SIMULA

1. EVOLUCIÓN HISTÓRICA
INICIOS DE LA ORIENTACIÓN A OBJETOS

- Finales de los 60's € Se crea Simula 67 (Noruega), el primero de los lenguajes orientado a objetos. Popularizó términos como *clases*, *objetos*, *instancias*, *herencia*, etc.
- Principios de los 70´s € Se da a conocer SmallTalk (Xerox Palo Alto). La gente consideró que era un sistema de ventanas y no apreció el paradigma. Se le considera el lenguaje mas puro: en SmallTalk todo son objetos.
  - Con Simula y SmallTalk quedó demostrada el ahorro de programación si las propiedades comunes a los objetos se programan una sola vez.
  - Eran lenguajes conocidos sólo en las Universidades.

1. EVOLUCIÓN HISTÓRICA
INICIOS DE LA ORIENTACIÓN A OBJETOS

- Mediados de los 70's € C se convierte en un lenguaje popular de desarrollo.
- Mediados de los 80's € Los Laboratorios Bell, ampliaron el lenguaje para que pudiera soportar el paradigma OO. Le llamó C++.
- Principios de los 90's:
  - El paradigma OO, empezó a llamar la atención.
  - La gente lo podía aprender la POO en un léxico ya conocido.
  - No se invertía esfuerzo en aprender nuevos entornos ni lenguajes.

1. EVOLUCIÓN HISTÓRICA LENGUAJES ORIENTADOS A OBJETOS

- Ada
- Modula
- Smalltalk (Alan Kay)
- Java (Sun Microsystems)
- Eiffel
- C++, C#
- Object Pascal
- ActionScript, ActionScript 3
- Clipper, Gambas, Harbour, Fortran 90/95, JavaScript
- Objective-C, Perl, PHP (a partir de su versión 5)
- PowerBuilder, Python, Ruby, Smalltalk
- VB.NET, Visual FoxPro (en su versión 6)

2. INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS

- La orientación a objetos es una nueva forma de pensar acerca del software, basado en el modelado del mundo real a través de la identificación de objetos.
- Surgió inicialmente como un enfoque para la programación pero se ha extendido a todo el ciclo de desarrollo de sistemas: análisis y diseño.
- El *análisis y diseño orientado a objetos* modela el mundo en términos de objetos de datos y operaciones de procesamiento, de forma tal que encapsula la información y el procesamiento.
- Este encapsulamiento es el paradigma fundamental de la orientación por objetos.

2. INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS

- El término "orientación a objeto" significa que organizaremos el software como una colección de objetos discretos que incorporan tanto estructuras de datos como procedimientos.

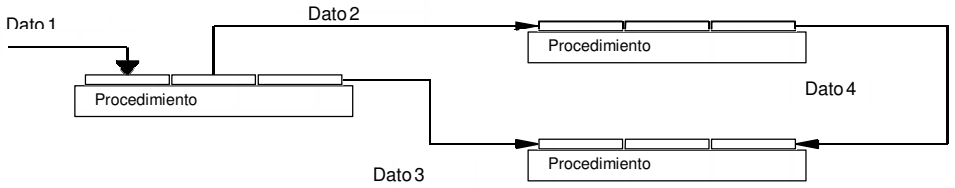


Fig. 2 Sistema convencional

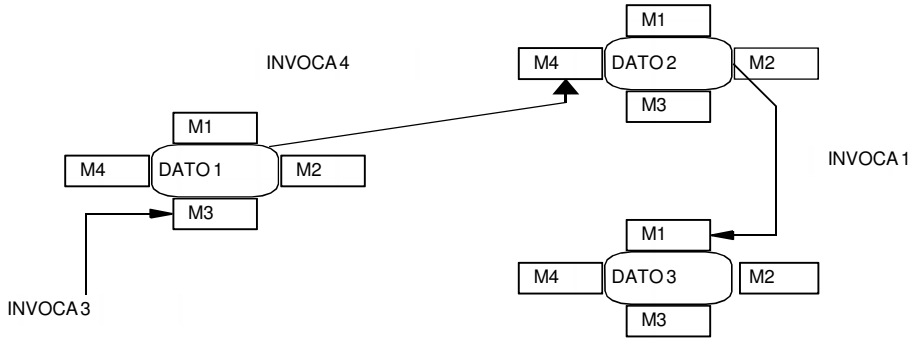
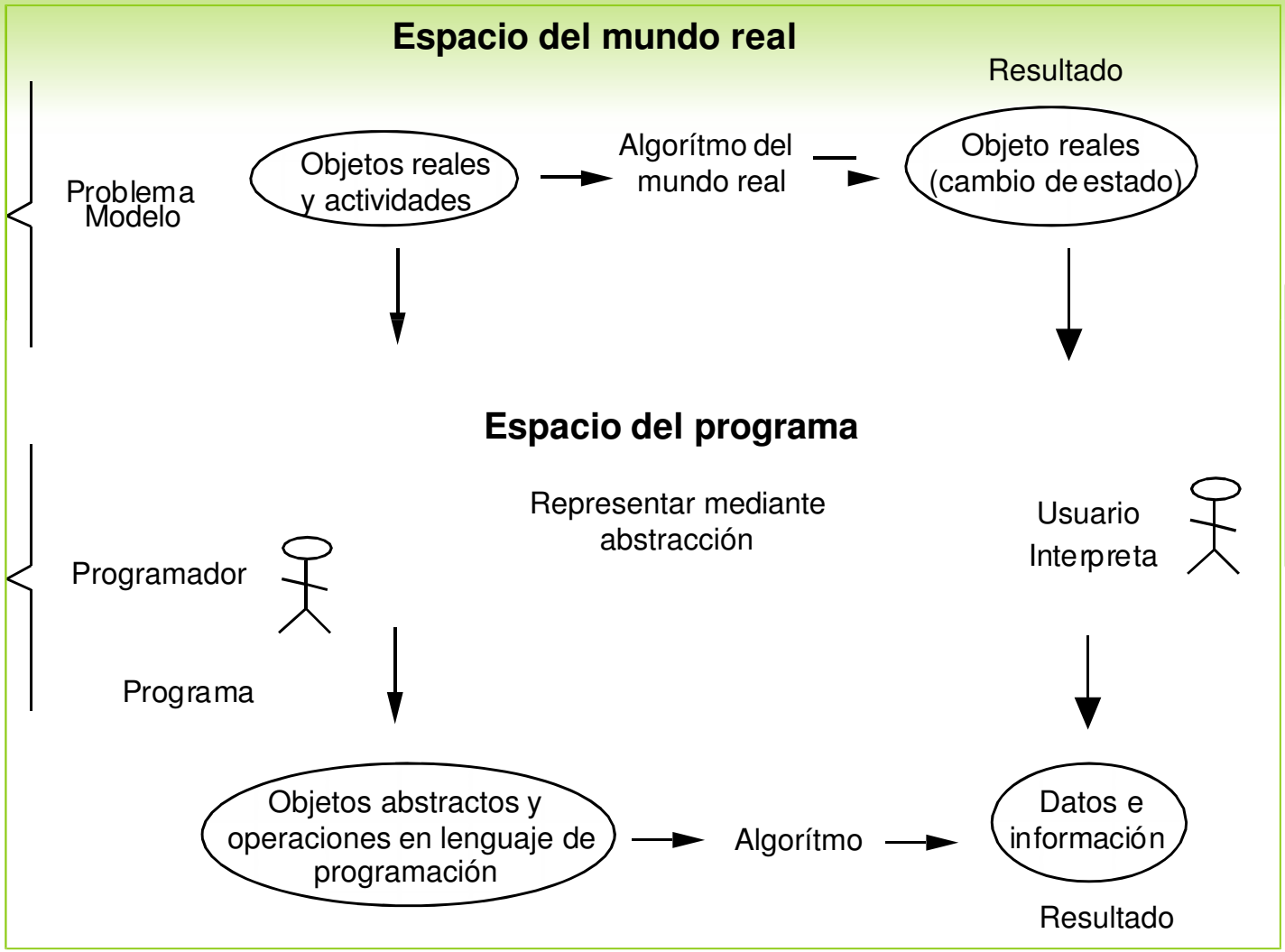


Fig.3 Sistema Orientado por objeto

Esto contrasta con la programación convencional, en la cual las estructura de datos y el comportamiento están solo aproximadamente conectados.

2. INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS
MODELADO MUNDO REAL



Modelo de una tarea típica de programación orientada a objetos

2. INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS
BENEFICIOS DE LA OO

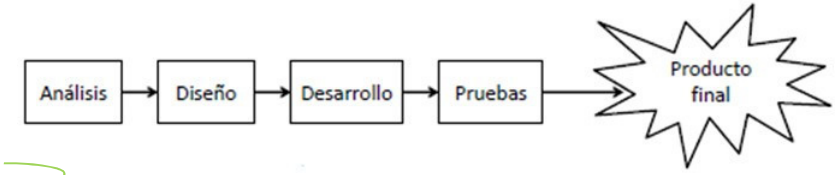
- Reutilización. Permite la reusabilidad de código y la herencia ahorrando dinero y empleando menos tiempo de desarrollo.
- Integridad. Los mecanismos de encapsulación protegen sus propios componentes contra los procesos que no tengan derecho a acceder a ellos.
- La forma de pensar en objetos es más natural. El diseñador piensa en términos de objetos y no en detalles de bajo nivel.
- Programación más sencilla. Los programas se crean a partir de piezas pequeñas.

2. INTRODUCCIÓN A LA ORIENTACIÓN A OBJETOS
BENEFICIOS DE LA OO

- Los métodos de los objetos pueden ser polimórficos, es decir, tienen la habilidad de enviar un mismo mensaje a objetos de clases diferentes, se “comportan” de distintas maneras.
- Es más sencillo modificar código existente, cada clase efectúa sus funciones independientemente de las demás.
- Se construyen clases cada vez más complejas a partir de otras más sencillas ya existentes.
- Confiabilidad. Generalmente las clases ya están probadas.
- Estabilidad de los modelos respecto a entidades del mundo real.

3. INGENIERÍA DEL SOFTWARE

La Ingeniería del software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software. Representa una metodología de desarrollo creciente hacia un fin.



- Construcción.
  - Análisis.
  - Especificación.
  - Diseño.
  - Desarrollo
- Mantenimiento.
  - Corrección de errores.
  - Cambios debido a revisiones.

El coste del Software

3. INGENIERÍA DEL SOFTWARE NUEVO ENFOQUE OO

- El uso de lenguajes como Simula, SmallTalk o C++, requirió un nuevo enfoque de análisis y de diseño.
  - Desarrollo de nuevas aplicaciones
  - Énfasis principal en la estructura de datos

> Aparición de los primeros métodos de diseño y de análisis orientados a objetos

- La ingeniería del software extendió la orientación a objetos a todo el ciclo de desarrollo de sistemas: análisis y diseño.
  - Análisis Orientado a Objetos (basada en TAD'S)
  - Diseño Orientado a Objetos (UML, Rose)
  - Programación Orientada a Objetos (Booch, Cood-Jourdan)

3. INGENIERÍA DEL SOFTWARE METODOLOGIAS OO

Procesos y metodologías asociadas a la orientación a objetos

- Rumbaugh, Blaha, Premeriani (OMT)· 1991
- Coad. Yourdon - 1991
- Shlaer, Mellor - 1992
- Booch -1992
- Odell, Martin - 1992
- Jacobson(OOSE)-1993
- Fusion - 1994
- Booch, Rumbaugh, Jaoobson (UML)· 1997