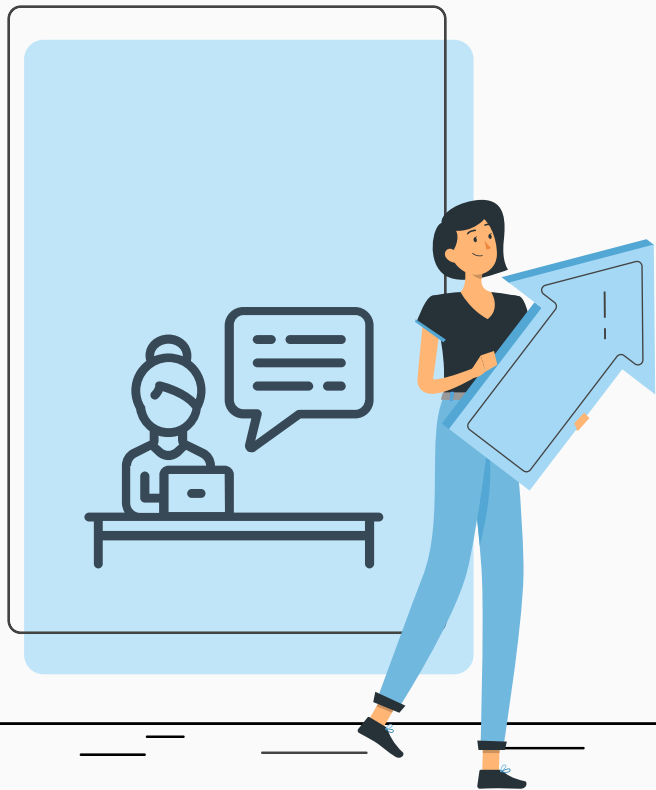


ORIENTACIÓN A OBJETOS

P00 paradigma de programación





01 PRINCIPIOS

Descripción del paradigma

02 HERENCIA

Subclases y super clases

03 CLASES Y OBJETOS

Atributos, métodos,
variables e instancias

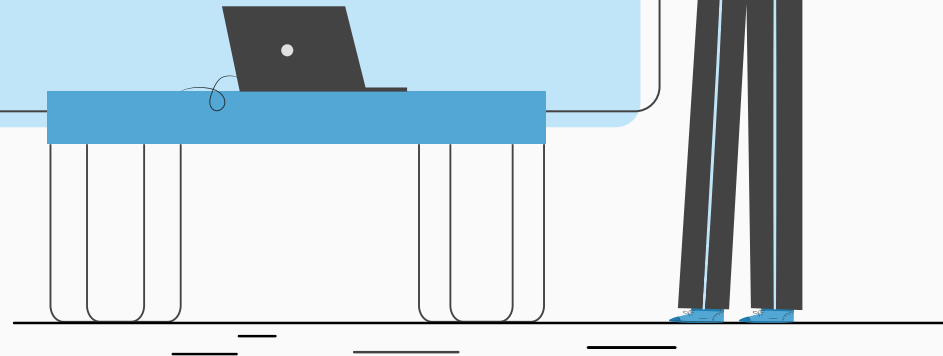
04 DESARROLLO

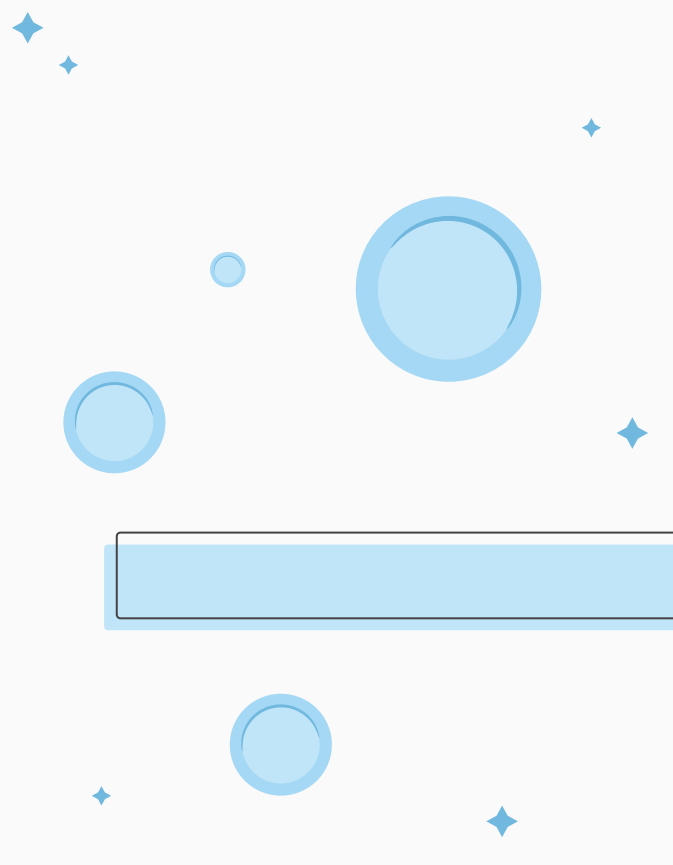
Librerías de clase,
herramientas y UML

INTRODUCCIÓN

Los objetos se utilizan como metáfora para emular las entidades reales del negocio a modelar.

En el paradigma orientado a objetos, el programa se organiza como un conjunto finito de objetos que contienen datos y operaciones.



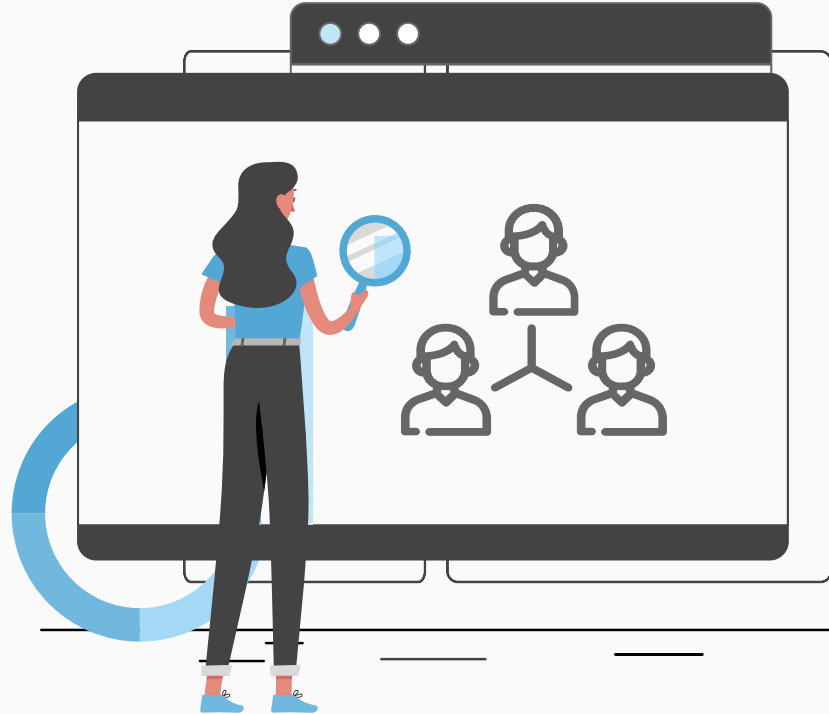
The slide features several decorative elements: a large blue circle with a double outline in the upper left, a medium blue circle with a double outline below it, and a small blue circle with a double outline to the right of the medium one. There are also several small blue four-pointed stars scattered around these circles. At the bottom, there is another medium blue circle with a double outline and two more small blue stars.

“Escojo a una persona perezosa para hacer
un trabajo duro. Porque una persona
perezosa encontrará una manera sencilla
de hacerlo.”

—BILL GATES

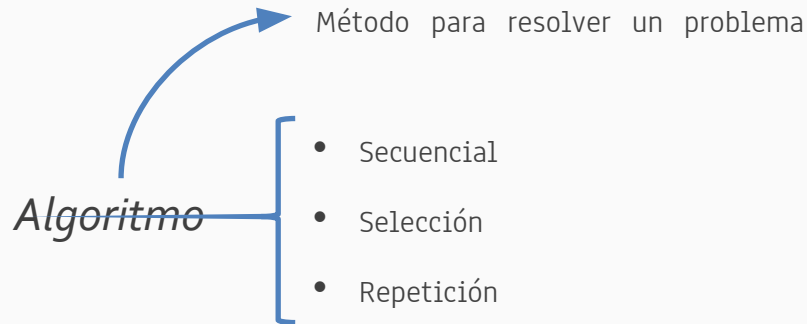
01. PRINCIPIOS

Surge como una evolución de la programación estructurada, basada en subrutinas y tipos abstractos de datos



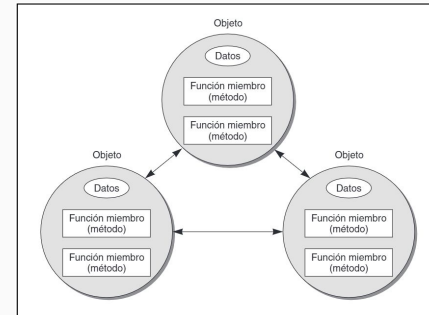


PROGRAMACIÓN ESTRUCTURADA



ORIENTADA A OBJETOS

Colección de objetos que tienen un determinado estado, comportamiento y propiedades.



CLASES

Una clase es un tipo abstracto de datos, una plantilla para crear objetos que recibe el nombre de instancia de clase

NOMBRE

PROPIEDADES

MÉTODOS

```
<?php
class NombreClase
{
    // Declaración de una propiedad
    public $var = 'un valor predeterminado';

    // Declaración de un método
    public function mostrarVar() {
        echo $this->var;
    }
}
?>
```

DEFINICIÓN DE UNA CLASE EN PHP

La definición básica de una clase comienza con la palabra reservada `class`, seguida de un nombre de clase, y continuando con un par de llaves que encierran las definiciones de las propiedades y métodos pertenecientes a dicha clase.



ACTIVIDAD

CONCEPTOS BÁSICOS

01 ATRIBUTOS

Son variables que almacenan datos propios y particulares de un objeto instanciado. Se declaran dentro de una clase con un nombre y un tipo de datos

03 EXCEPCIONES

Los lenguajes orientados a objetos cuentan con un conjunto de instrucciones para controlar los errores o las excepciones en las que incurra el programa

02 MÉTODO

Son las funciones o procedimientos declarados dentro de la clase, que constituyen las acciones realiza el objeto

04 AGRAGACIÓN DE CLASES

Un programa puede componerse de varias clases y a su vez una clase puede componerse de varias clases, esto se llama agregación de clases (hijas)

DEFINICIÓN DE UNA PROPIEDAD

Usando una de las palabras reservadas **public**, **protected**, o **private**, seguido de una declaración normal de variable. Esta declaración puede incluir una inicialización, pero esta inicialización debe ser un valor constante, es decir, debe poder ser evaluada durante la compilación y no depender de información generada durante la ejecución.

```
<?php
/** Definición de MyClass */
class MyClass
{
    public $public = 'Public';
    protected $protected = 'Protected';
    private $private = 'Private';

    function printHello()
    {
        echo $this->public;
        echo $this->protected;
        echo $this->private;
    }
}

$obj = new MyClass();
echo $obj->public;      // Funciona bien
echo $obj->protected;  // Error Fatal
echo $obj->private;    // Error Fatal
$obj->printHello();    // Muestra Public,
                      Protected y Private
?>
```

CREAR UNA INSTANCIA DE UN OBJETO

INSTANCIAR UN OBJETO

```
$objeto = new MiClase();
```

LLAMAR A UNA PROPIEDAD

```
$Objeto->propiedad;
```

CAMBIAR EL VALOR DE LA PROPIEDAD

```
$Objeto->propiedad = 'valor a guardar';
```

LLAMAR A UN MÉTODO

```
$Objeto->metodo();
```

VISIBILIDAD DE ATRIBUTOS Y MÉTODOS

PUBLIC

Modificador de acceso que hace visible al atributo o método desde cualquier parte del código

PRIVATE

Modificador de acceso que hace visible al atributo o método sólo desde la clase que lo define (no se hereda)

PROTECTED


Modificador de acceso que hace visible al atributo o método desde la clase que lo declara y en sus clases hijas



PHP posee un concepto de **destructor** similar al de otros lenguajes, como C++. El método destructor será llamado cuando no hayan otras referencias al objeto o en cualquier otra circunstancia de finalización.

```
class MyDestructableClass
{
    function __destruct() {
        print "Destruyendo " . __CLASS__ . "\n";
    }
}
$obj = new MyDestructableClass();
```

Son dos guiones
bajos seguidos _ _
sin espacio



CONSTRUCTORES Y DESTRUCTORES



ACTIVIDAD

HERENCIA

PRINCIPIO DE PROGRAMACIÓN

Este principio afectará la manera en que las clases y objetos se relacionan entre sí.

REUTILIZAR

La herencia permite la implementación de funcionalidad adicional en objetos similares sin tener que reimplementar nada

Una clase heredada se define mediante la **extends** palabra clave.

La palabra clave **final** se puede utilizar para evitar la herencia de clases o para evitar la invalidación del método.

EXTENDS

Cuando se extiende una clase, la subclase hereda todos los métodos públicos y protegidos de la clase padre

FUNCIONALIDAD

Si la clase no sobrescribe esos métodos, mantendrán su funcionalidad original





ACTIVIDAD

Lenguaje unificado de modelado: Es un lenguaje gráfico para **visualizar**, especificar, construir y documentar un sistema.

UML, estándar para describir un modelo del sistema, incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

ESTÁNDAR UML

TIPOS DE DIAGRAMAS EN UML

ESTRUCTURALES

Muestran la estructura estática de los objetos en un sistema

DE COMPORTAMIENTO

Muestran el comportamiento dinámico de los objetos en el sistema

DE INTERACCIÓN

Muestran las interacciones de los objetos en un sistema

ESTRUCTURALES



DIAGRAMA DE CLASES

Son los más utilizados en DOO. Clases, atributos, operaciones y relaciones



DIAGRAMA DE COMPONENTES

Relación estructural de los componentes de un sistema de software

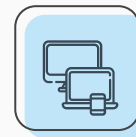


DIAGRAMA DE DESPLIEGUE

Hardware del sistema y el software de ese hardware.



DIAGRAMA DE OBJETOS

Diagramas de instancia, son muy similares a los diagramas de clases



DIAGRAMA DE PAQUETES

Dependencias entre diferentes paquetes de un sistema



DIAGRAMA DE PERFILES

Se utiliza muy raramente en cualquier especificación.

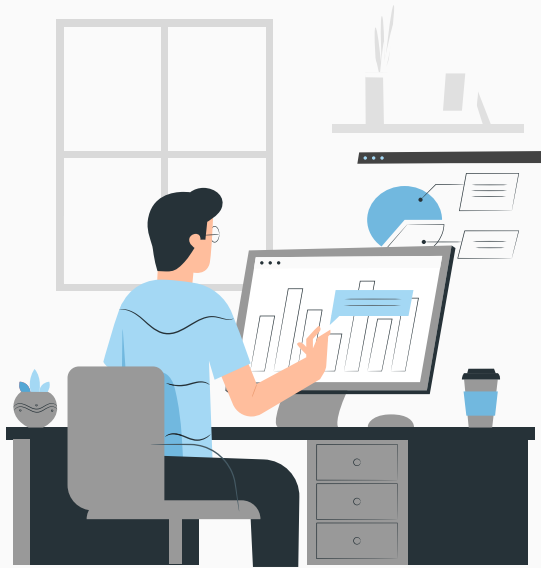
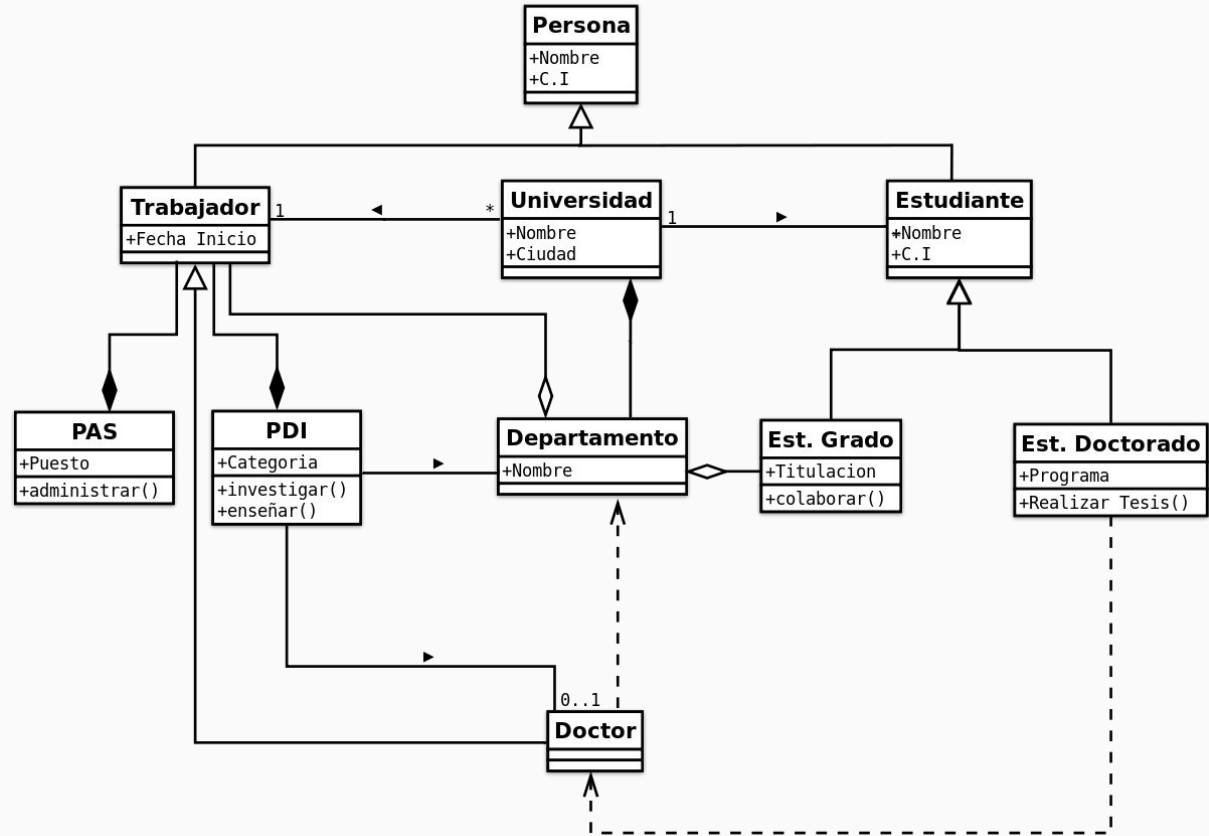


DIAGRAMA DE CLASES



DE COMPORTAMIENTO

DIAGRAMA DE ACTIVIDADES

Representan los flujos de trabajo de forma gráfica

DIAGRAMA DE CASOS DE USO

Ofrecen una visión general de los actores involucrados en un sistema

DIAGRAMA DE MÁQUINA DE ESTADOS

Útiles para describir el comportamiento de los objetos que actúan de manera diferente de acuerdo con el estado en que se encuentran en un momento dado

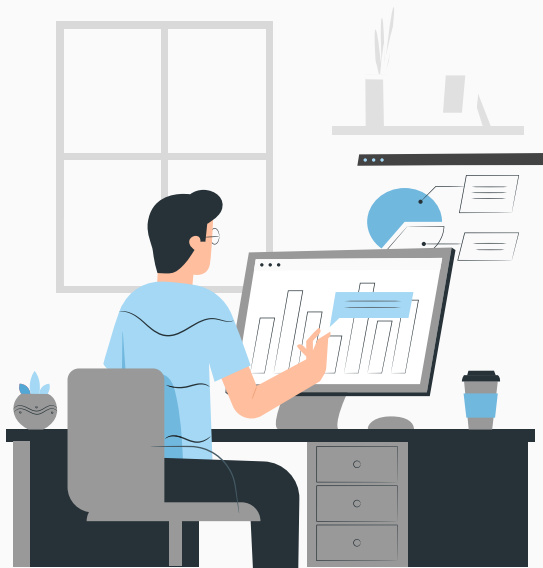
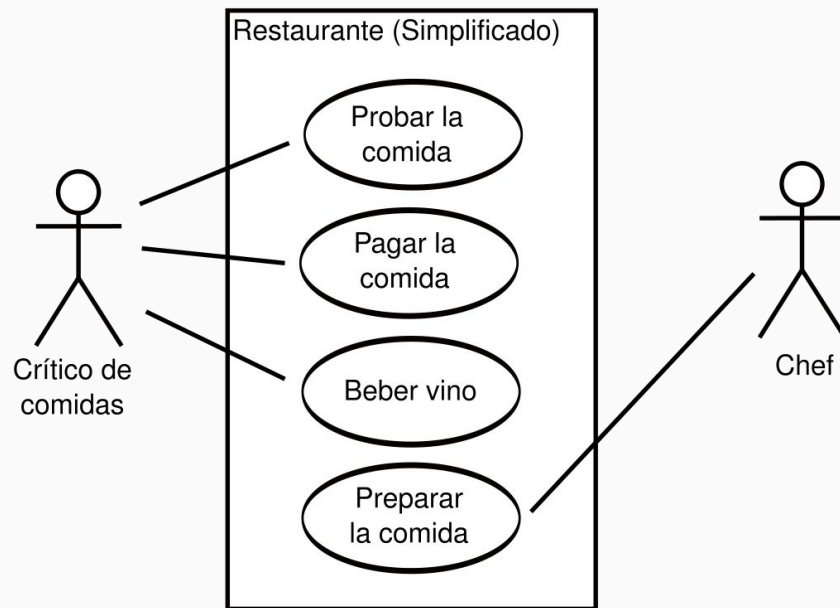


DIAGRAMA DE CASOS DE USO



DE INTERACCIÓN

DIAGRAMA GLOBAL DE INTERACCIONES

Muestran una secuencia de diagramas de interacción

DIAGRAMA DE COMUNICACIÓN

Es similar a los de secuencia, pero el foco está en los mensajes pasados entre objetos

DIAGRAMA DE SECUENCIA

Muestran cómo los objetos interactúan entre sí y el orden en que se producen esas interacciones

DIAGRAMA DE TIEMPOS

Representan el comportamiento de los objetos en un marco de tiempo dado

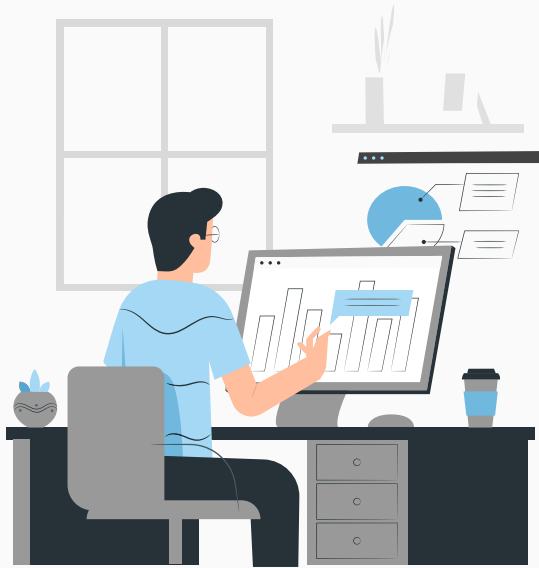
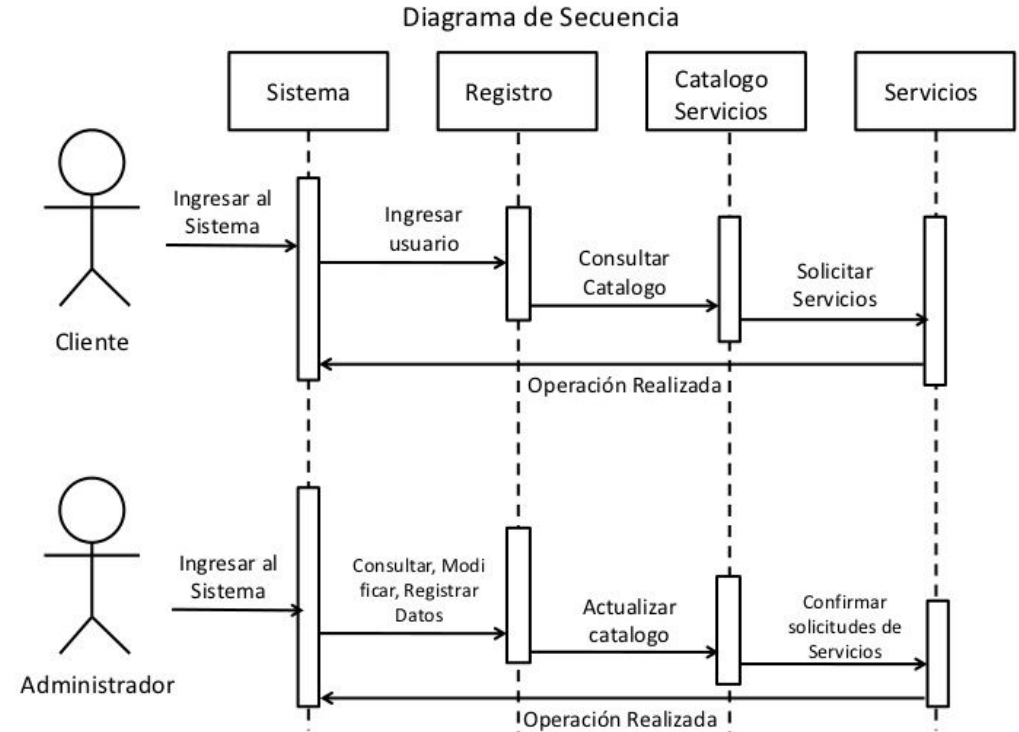


DIAGRAMA DE SECUENCIA



**“ LEY DE ALZHEIMER DE LA PROGRAMACIÓN:
SI LEES UN CÓDIGO QUE ESCRIBISTE HACE MÁS DE DOS
SEMANAS ES COMO SI LO VIERAS POR PRIMERA VEZ”**

– Dan Hurvitz.

GRACIAS

¿Tienes alguna pregunta?

bitonobit@gmail.com



CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik** and illustrations by **Storyset**

Please keep this slide for attribution.

