


DISEÑO DE SISTEMAS

Juan Carlos Molina Lozano
Docente

CONTENIDO


- Introducción
 - Objetivos de la Clase
 - Propósito del Diagrama de Colaboración
 - Características del Diagrama de Colaboración
 - Ventajas y Desventajas
 - Elementos del Diagrama de Colaboración
 - Comparación con el Diagrama de Secuencia
 - Ejemplos Prácticos
 - Actividades en Clase
- 

OBJETIVOS DE LA CLASE

- Comprender el concepto y utilidad del Diagrama de Colaboración en el diseño de software.
- Identificar los elementos que componen un diagrama de colaboración y su representación gráfica.
- Diferenciar el diagrama de colaboración de otros diagramas de interacción, como el Diagrama de Secuencia.
- Aplicar los conocimientos adquiridos mediante la realización de ejemplos prácticos.
- Desarrollar la capacidad de modelar la interacción de objetos en un sistema real.

INTRODUCCIÓN

El Diagrama de Colaboración es una herramienta utilizada en UML (Unified Modeling Language) para representar la interacción entre objetos en un sistema. Este diagrama se enfoca en mostrar las relaciones estructurales entre los objetos y cómo intercambian mensajes para cumplir una tarea específica.

A decorative graphic consisting of several parallel white lines of varying lengths, slanted diagonally from the bottom right towards the top right, set against a blue gradient background.

PROPÓSITO DEL DIAGRAMA DE COLABORACIÓN

El diagrama de colaboración tiene como objetivo principal visualizar cómo los objetos de un sistema trabajan en conjunto para lograr una funcionalidad específica. Su propósito incluye:

- Modelar la interacción de objetos en el contexto de un escenario específico.
- Identificar la comunicación entre objetos mediante el intercambio de mensajes.
- Servir como una guía para la implementación del sistema en programación orientada a objetos.
- Facilitar la comprensión de la estructura del sistema y las dependencias entre sus componentes.

CARACTERÍSTICAS DEL DIAGRAMA DE COLABORACIÓN

- Enfatiza la estructura: Muestra cómo los objetos están relacionados entre sí.
- Uso de mensajes numerados: Indica el orden en que ocurren las interacciones entre objetos.
- Relaciones entre objetos: Representadas por líneas de asociación.
- Utilizado para modelar interacciones específicas dentro de un proceso o sistema.
- Complementario al diagrama de secuencia: Aunque ambos representan interacciones, el diagrama de colaboración se enfoca en las relaciones estructurales.

VENTAJAS Y DESVENTAJAS

Ventajas

- ✓ Facilita la identificación de relaciones entre objetos.
- ✓ Ayuda a visualizar cómo los objetos colaboran en un proceso.
- ✓ Útil en el diseño de sistemas orientados a objetos.
- ✓ Muestra claramente el flujo de mensajes y su secuencia.

Desventajas


- ✗ Puede volverse complejo si hay demasiados objetos y mensajes.
- ✗ Menos intuitivo que el diagrama de secuencia para representar la dinámica de la interacción.
- ✗ Difícil de interpretar cuando hay muchas conexiones entre objetos.

ELEMENTOS DEL DIAGRAMA DE COLABORACIÓN

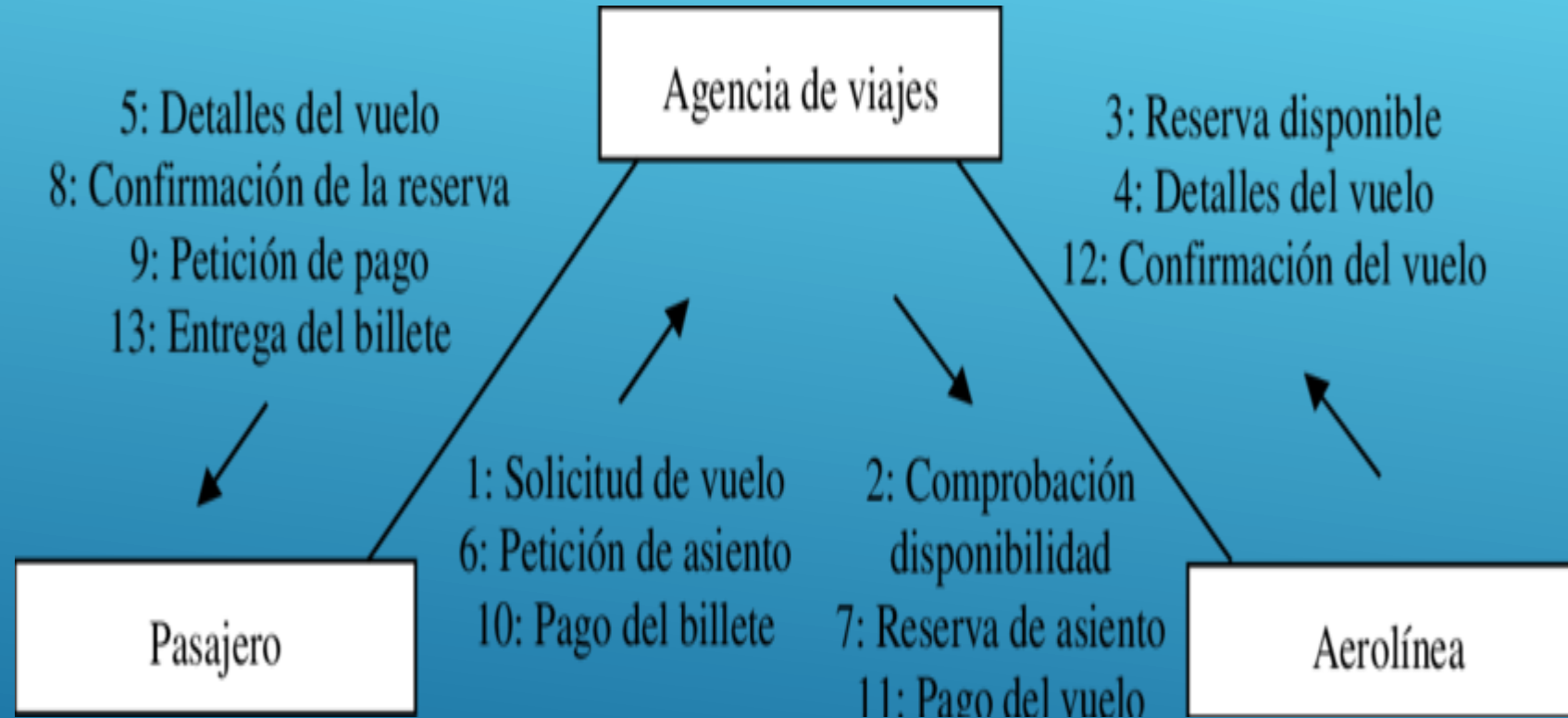
- **Objetos:** Representados como rectángulos con el nombre del objeto subrayado.
Ejemplo: usuario:Cliente
- **Líneas de asociación:** Representan las relaciones entre los objetos.
- **Mensajes:** Indicados con una flecha y numerados para mostrar el orden de ejecución.
- **Ejemplo:** 1: autenticarUsuario()
- **Actores externos:** Representan entidades externas que interactúan con el sistema.

COMPARACIÓN CON EL DIAGRAMA DE SECUENCIA

CARACTERÍSTICA	DIAGRAMA DE COLABORACIÓN	DIAGRAMA DE SECUENCIA
Enfoque	Estructura de la interacción	Orden de los eventos
Representación de objetos	Relación entre objetos	Tiempo y eventos
Facilidad de lectura	Puede ser complejo	Más intuitivo



EJEMPLO



PRACTICA: INICIAR SESIÓN EN UN SISTEMA

- Objetos involucrados:
 1. Usuario
 2. PantallaLogin
 3. SistemaAutenticación
 4. BaseDeDatos
- Genere el diagrama de colaboración para este caso.
- Mensajes:
 1. Usuario → PantallaLogin: ingresarDatos()
 2. PantallaLogin → SistemaAutenticación: validarUsuario()
 3. SistemaAutenticación → BaseDeDatos: consultarCredenciales()
 4. BaseDeDatos → SistemaAutenticación: devolverResultado()
 5. SistemaAutenticación → PantallaLogin: mostrarResultado()

PRACTICA: REGISTRAR USUARIO EN UN SISTEMA

- Objetos involucrados:
 1. Usuario
 2. FormularioRegistro
 3. ControladorRegistro
 4. BaseDeDatos
- Mensajes:
 1. Usuario → FormularioRegistro: ingresarDatos()
 2. FormularioRegistro → ControladorRegistro: validarDatos()
 3. ControladorRegistro → BaseDeDatos: guardarUsuario()
 4. BaseDeDatos → ControladorRegistro: confirmación()
 5. ControladorRegistro → FormularioRegistro: notificarÉxito()
- Genere el diagrama de colaboración para este caso.

PRACTICA: MODIFICAR DATOS DE UN USUARIO

- Objetos involucrados:
 1. :Usuario
 2. FormularioEdición
 3. ControladorUsuario
 4. BaseDeDatos
- Mensajes:
 1. Usuario → FormularioEdición: editarDatos()
 2. FormularioEdición → ControladorUsuario: validarCambios()
 3. ControladorUsuario → BaseDeDatos: actualizarDatos()
 4. BaseDeDatos → ControladorUsuario: confirmación()
 5. ControladorUsuario → FormularioEdición: notificarÉxito()
- Genere el diagrama de colaboración para este caso.

PRACTICA: CONSULTAR DATOS DE UN USUARIO

- Objetos involucrados:
 1. Usuario
 2. InterfazConsulta
 3. ControladorConsulta
 4. BaseDeDatos
- Mensajes:
 1. Usuario → InterfazConsulta: solicitarConsulta()
 2. InterfazConsulta → ControladorConsulta: procesarSolicitud()
 3. ControladorConsulta → BaseDeDatos: obtenerDatos()
 4. BaseDeDatos → ControladorConsulta: devolverDatos()
 5. ControladorConsulta → InterfazConsulta: mostrarDatos()
- Genere el diagrama de colaboración para este caso.

PREGUNTAS

