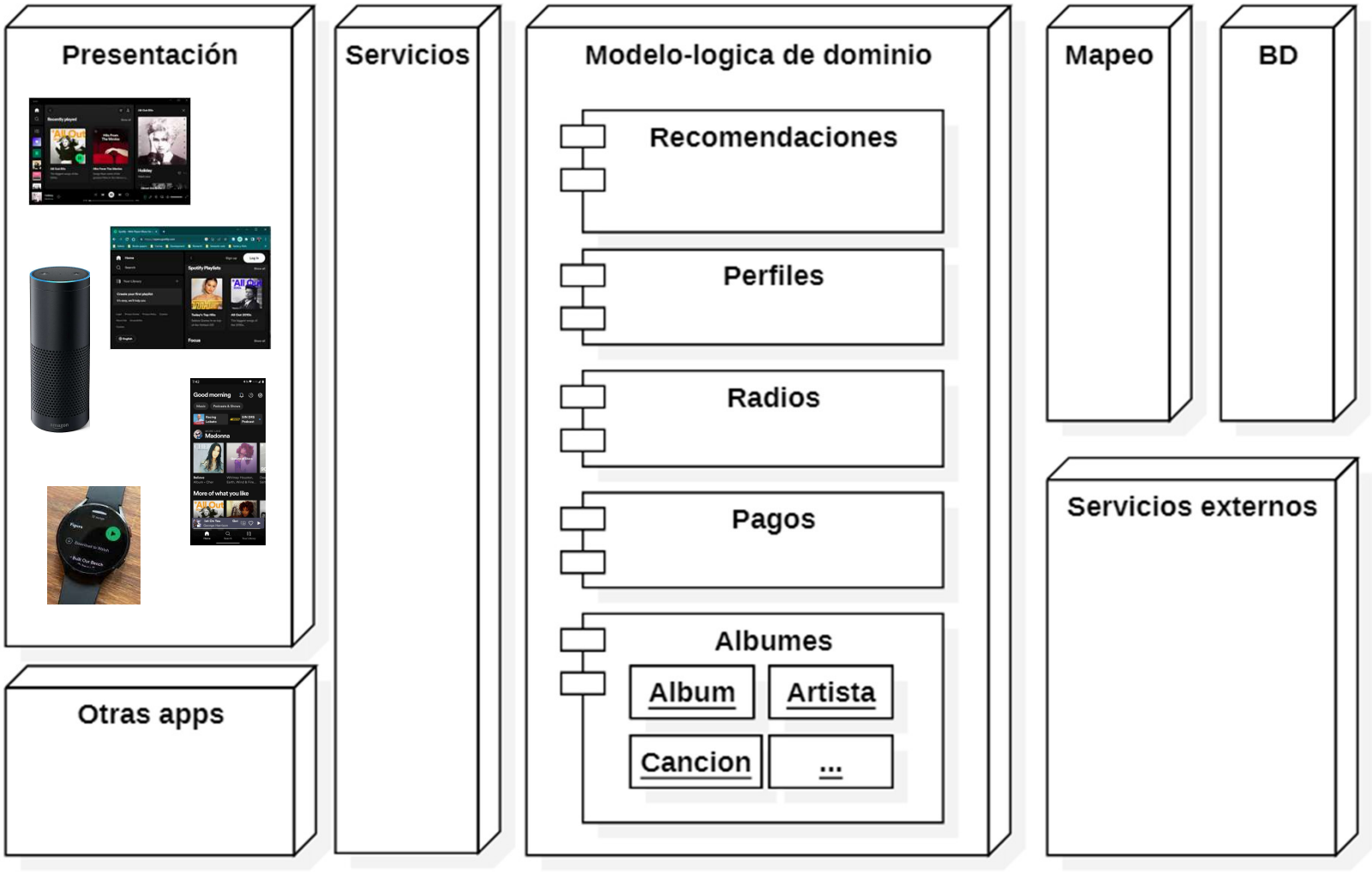
# Arquitectura de referencia de OO1

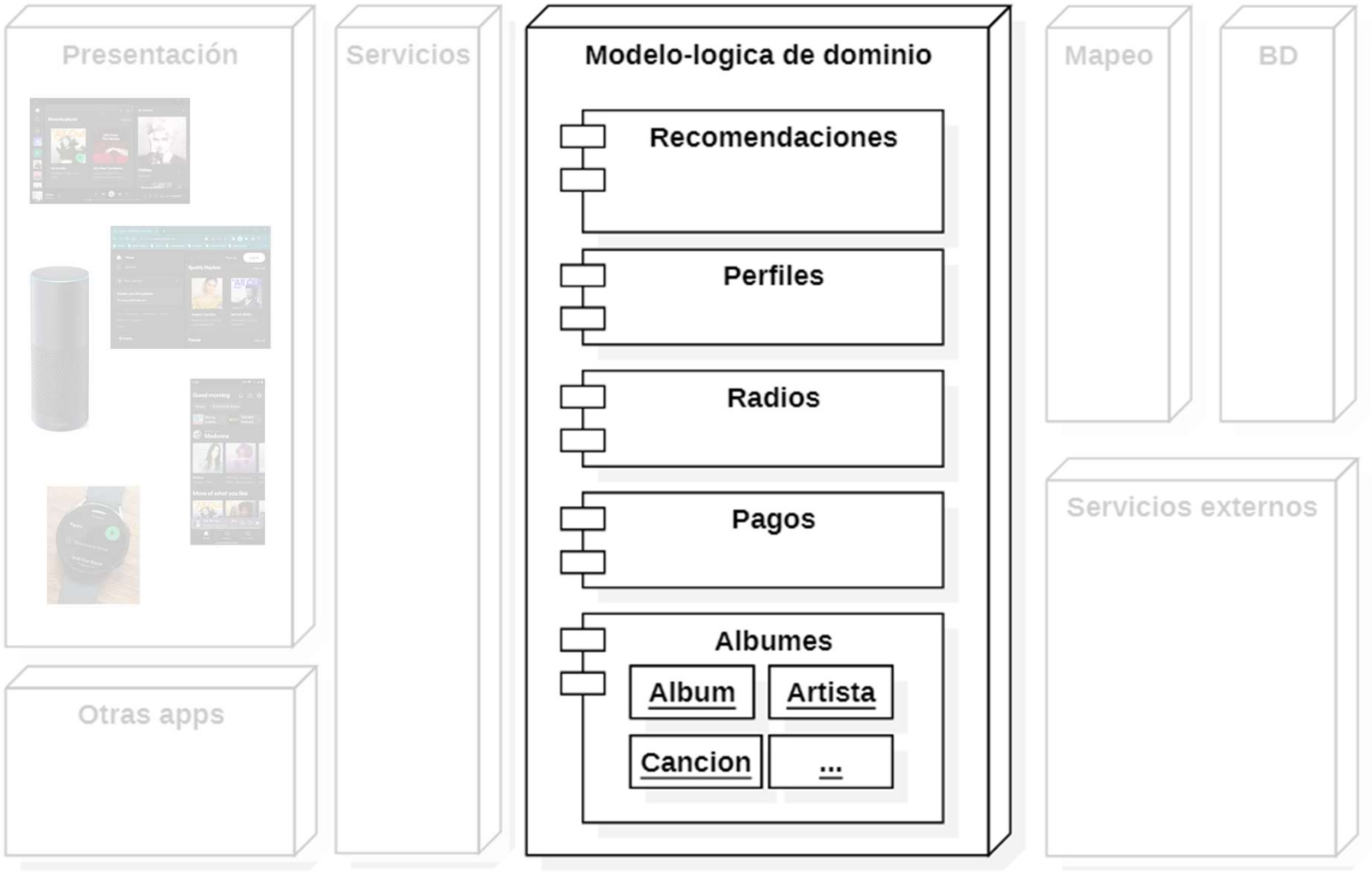
Lógica de interacción, lógica de dominio, tests unitarios y el “monolito pedagógico”

Manolito Pedagógico

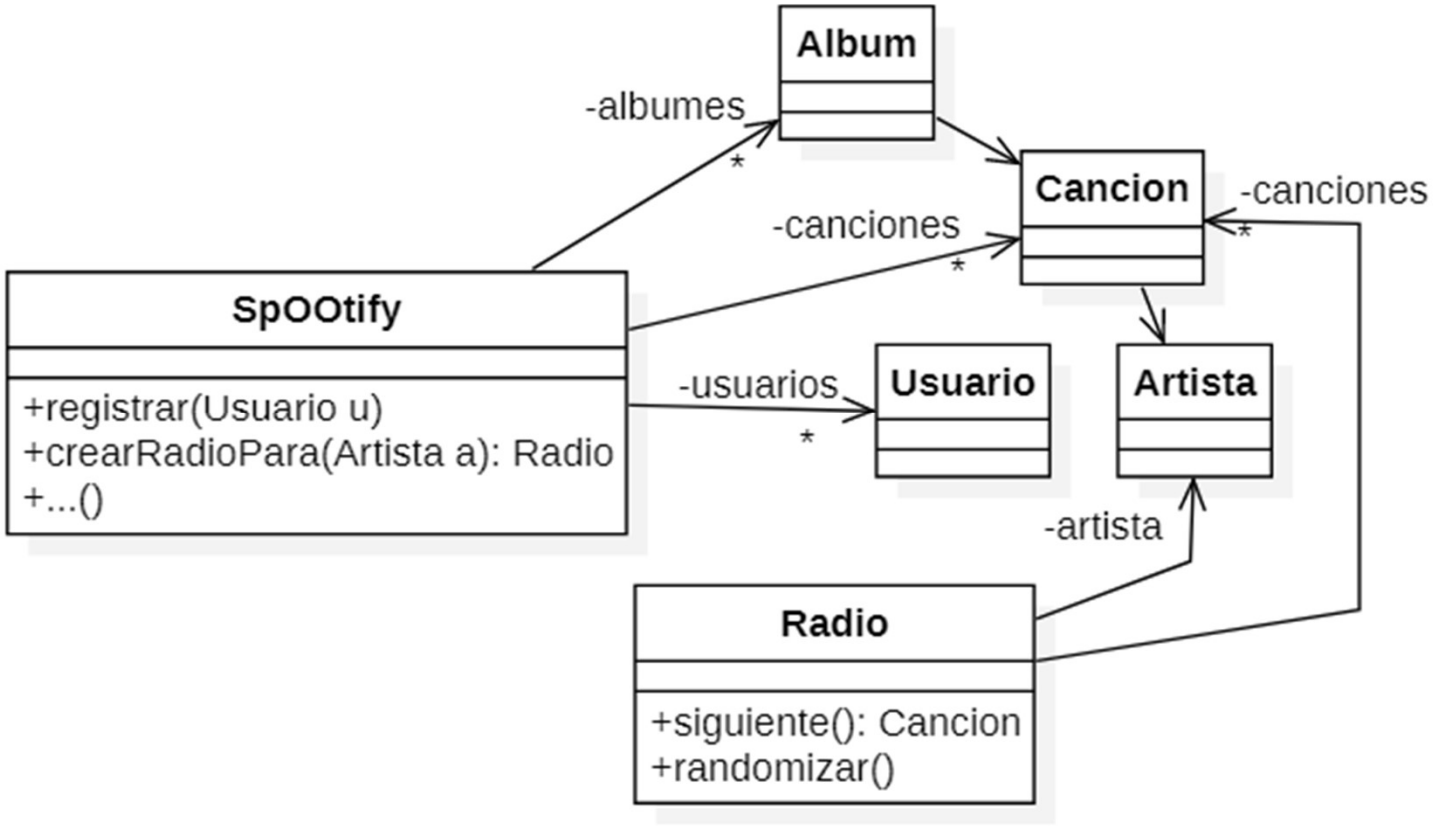
# Un programa/sistema/aplicación hoy ...



# Nuestro foco en OO1

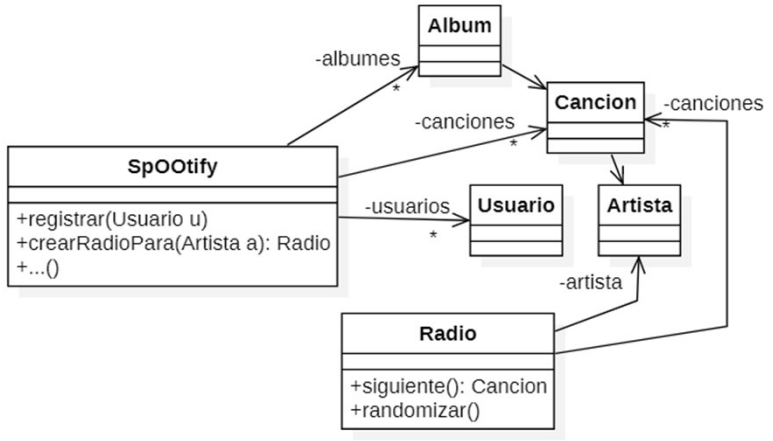


# Nuestra arquitectura (monolítica) ...



# Nuestra arquitectura simplificada (monolítica)

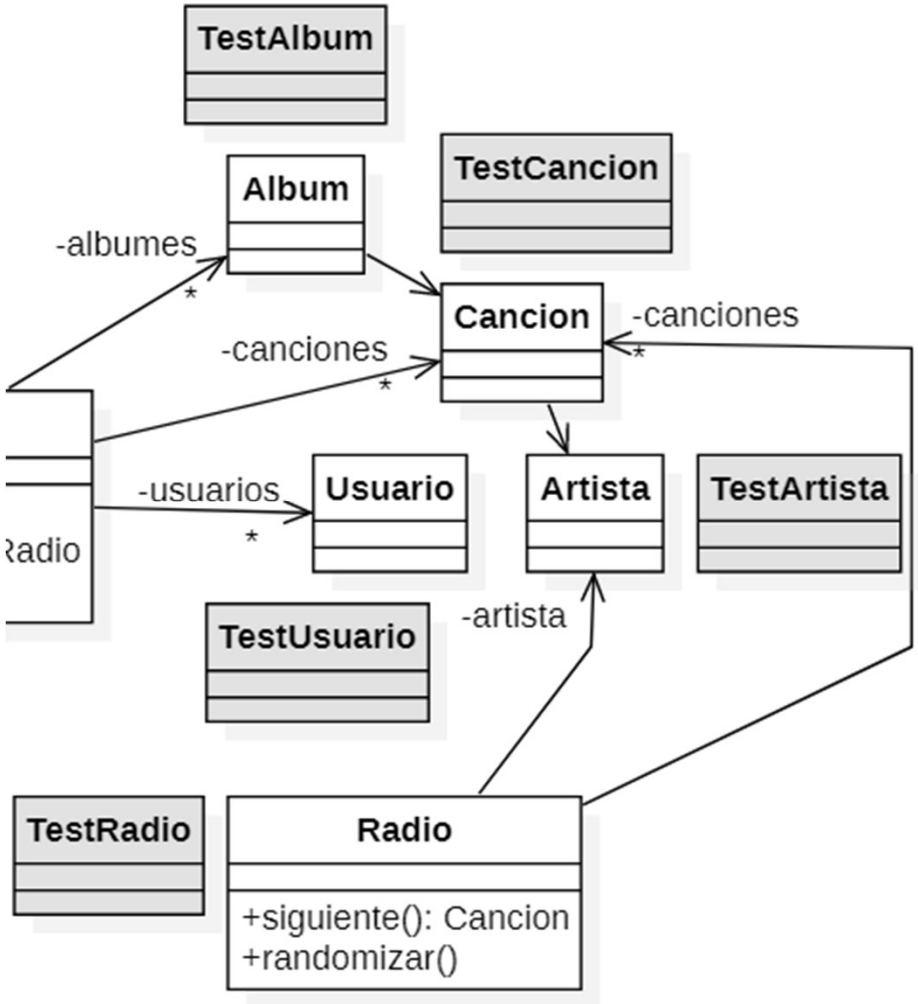
- Un objeto que modela el sistema/servicio
  - Asumimos que se mantiene en el tiempo
  - Persiste objetos en colecciones
  - Es nuestro punto de entrada al grafo de  objetos
  - Hace lo menos posible (no es un “main”)
- Al arrancar, solo tenemos una instancia de SpOOtify




- Lo que aprendamos, es aplicable a arquitecturas más modernas
- En otras materias aprendemos a “romper el monolito”

# Tests unitarios

- Son objetos que prueban a otros
- Son “parte” de “programar”
- Por cada clase de mi modelo, tengo una clase de test
  - Cada método prueba una cosa
- Los objetos test son instanciados por una herramienta
  - Envía mensajes de test
  - Colecta los resultados
  - Reporta resultados al final



# Tests unitarios

- Utilizo @BeforeEach para indicar el método que prepara lo necesario
- Utilizo @Test para indicar métodos que son tests
  - Cada método @Test ejercita una tarea/comportamiento
  - Utiliza assert...() para confirmar comprobar efectos
- Aprenderemos a diseñarlos bien

```
*/
class WallPostTest {

  WallPost wallPost;
  WallPost coolPost;

  @BeforeEach
  void setUp() throws Exception {
    wallPost = new WallPostImpl();
    coolPost = new WallPostImpl();
    coolPost.like();
    coolPost.like();
    coolPost.like();
    coolPost.like();
    coolPost.toggleFeatured();
  }

  @Test
  void testCreation() {
    assertEquals("Undefined post", wallPost.getText());
    assertEquals(0, wallPost.getLikes());
    assertEquals(false, wallPost.isFeatured());
  }
```