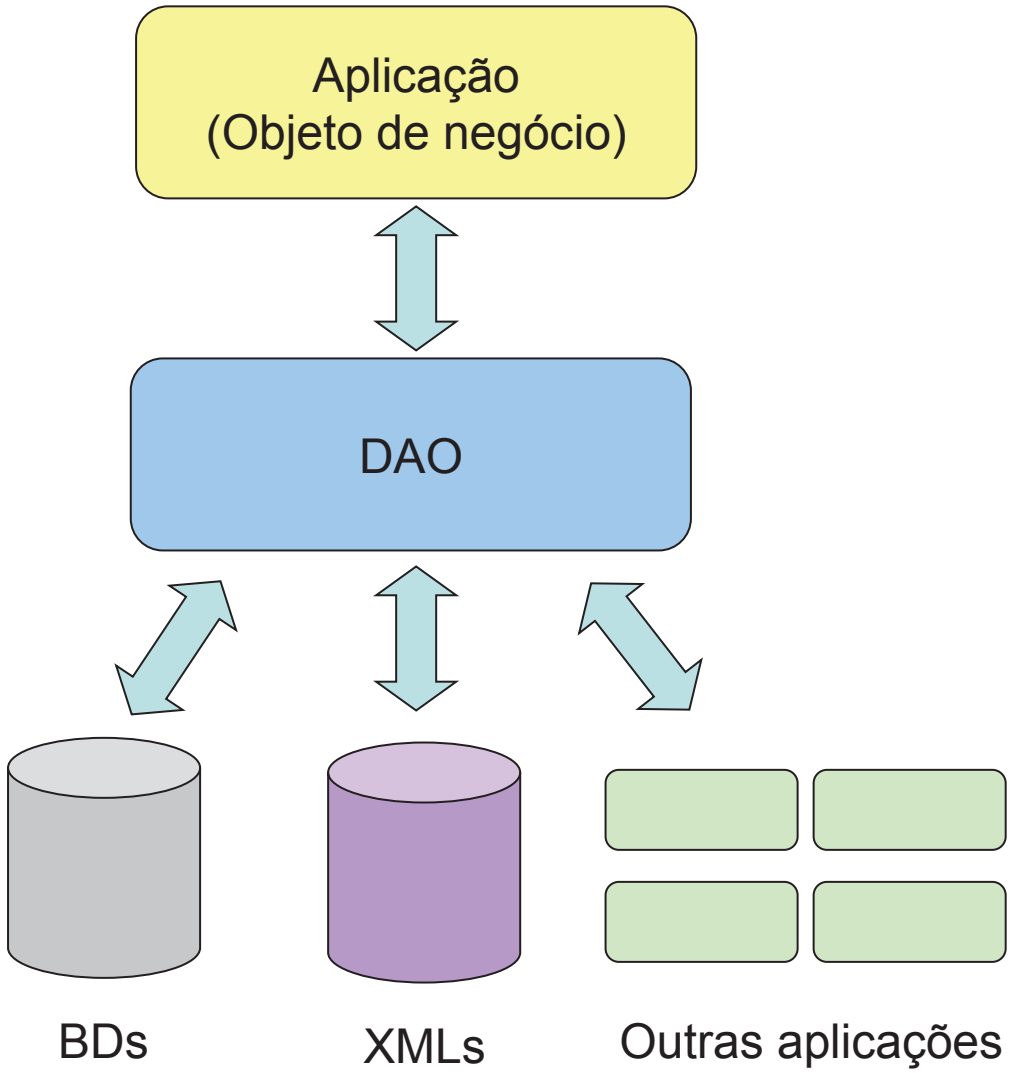
DAO (Data Access Object)

- DAO (Data Access Object)
  - Cria uma abstração responsável pelo acesso a banco de dados.
  - DAOs encapsulam todo o acesso a dados referente às nossas entidades.
  - É um padrão JEE
    - Baseado no padrão de projeto Strategy (Gamma, 1995)

Padrão de projeto DAO

- Data Access Object
- **Encapsula** as operações JDBC, **provendo acesso** de nível **mais alto** aos demais componentes da aplicação
- Pode ser usado em conjunto com outros padrões, como o Singleton

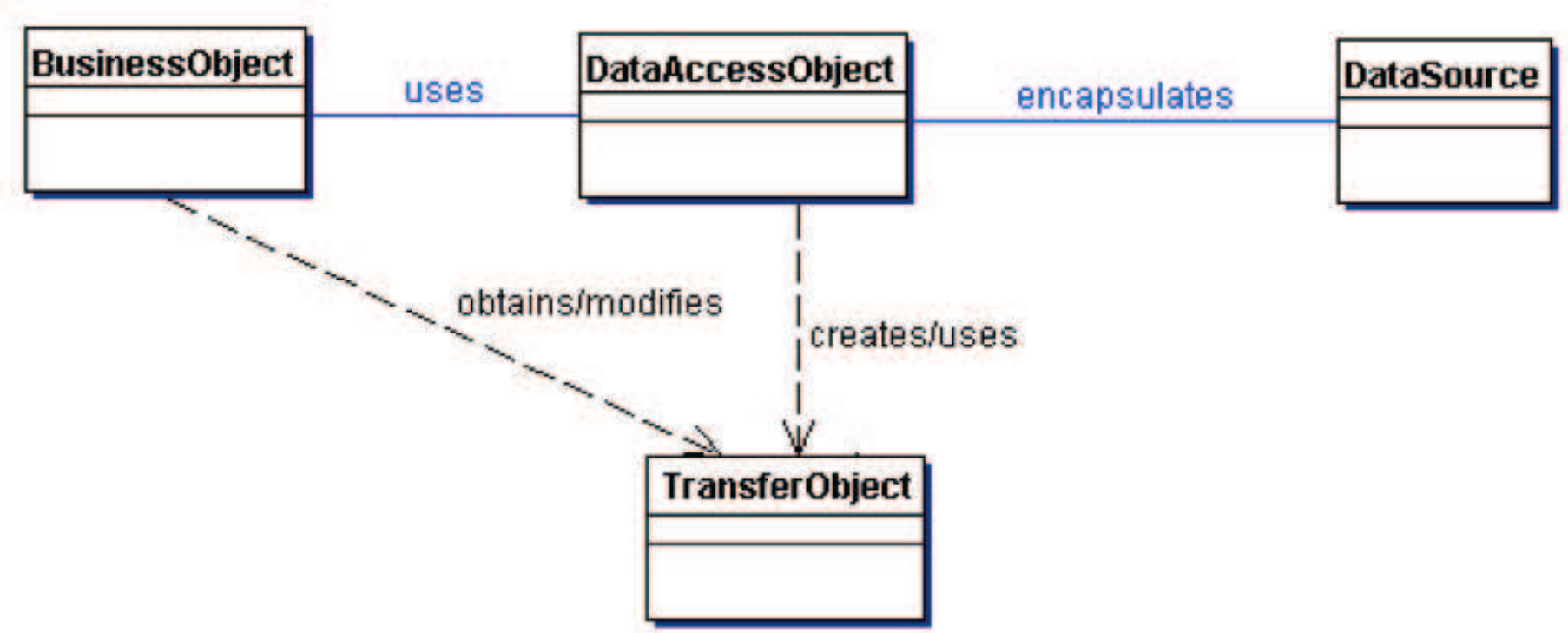
Data Access Object - DAO



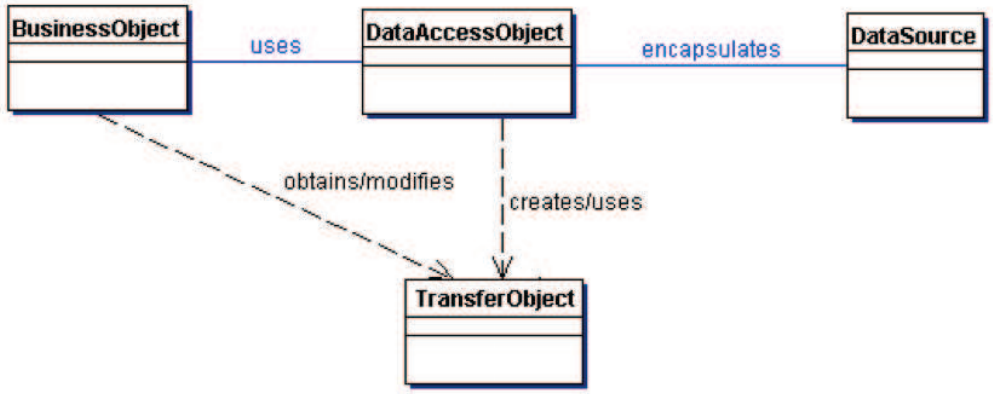
DAO - Data Access Object

- Data Access Object é um padrão da camada de integração para recuperação de dados, independente da fonte, ou seja,
  - podem ser utilizadas vários *data sources* (de uma mesma maneira), deixando o acesso totalmente transparente para a aplicação;
  - possibilita a troca do mecanismo de persistência de forma transparente para a aplicação;
  - Separa as lógicas de negócio e de acesso a dados;
  - Aumenta a portabilidade.

Estrutura do padrão DAO



DAO



- Participantes
  - BusinessObject (BO)
    - Requisita acesso para armazenar ou requisitar algum dado de algum base.
  - DataAccessObject (DAO)
    - Oferece serviços para o BusinessObject de forma transparente.
  - DataSource
    - Representa a fonte de dados (ex: BD, outro sistema, repositório XML, etc). Acessada pelo DAO.
  - TransferObject (TO)
    - Usado para representar os dados obtidos pelo DAO e para serem compreendidos pelo BusinessObject.

aplicação prática usando DAO

```
public class ClienteTO {
        // atributos
        // getters e setters
}


public interface ClienteDAOInterface {
        ClienteTO create () ;
        void insert ( ClienteTO c ) ;
        void update (ClienteTO c ) ;
        void delete (ClienteTO c ) ;
        ClienteTO findByID ( Integer id ) ;
        ClienteTO finfByNumber ( String number ) ;
}
```

Aplicação prática usando DAO

```
public class JDBCClienteDAO implements ClienteDAOInterface {
        public ClienteTO create () {
        …
        }
        public void insert (ClienteTO c ) {
        …
        }
        public void update (ClienteTO c ) {
        …
        }
        public void delete (ClienteTO c ) {
        …
        }
        public ClienteTO findByID ( Integer id ) {
        …
        }
        public ClienteTO finfByNumber ( String number ){
        …
        }
}
```

Características do uso do padrão DAO

- **TO** - Objetos de transporte. Eles são encarregados de manter os dados em memoria na forma de objetos e estruturas de objetos. Exemplo: ClientePOJO

- **Interface DAO** - Interface para o DAO de um certo tipo de TO. Exemplo: ClienteDAOInterface

- **Implementação DAO** - Implementação da interface para o DAO de um certo tipo de TO e um certo tipo de API de persistência. Exemplo: JDBCClienteDAO

DAO Genérico

- Usando tipos genéricos ajuda a diminuir o numero de métodos por interface DAO utilizando interfaces comuns.
  - Portanto, usar *Generics* não nos ajuda a diminuir o número de classes ou simplificar a implementação!
- Com “Genérico”, diminui-se a quantidade de métodos na interface DAO específica de cada objeto de transporte, mas não ganhamos muito.
  - Criamos mais uma classe e a implementação ainda continua específica.

Exemplo de DAO Genérico

```
public interface GenericDAO<T> {

        T create () ;
        void insert ( <T> obj ) ;
        void update ( <T> obj ) ;
        void delete ( <T> obj ) ;
        T findByID ( Integer id ) ;

}

public interface CustomerDAO extends GenericDAO<Customer> {

    Customer finfByCustomerNumber ( String customerNumber ) ;
}
```