Le design pattern DAO
Data Access Object

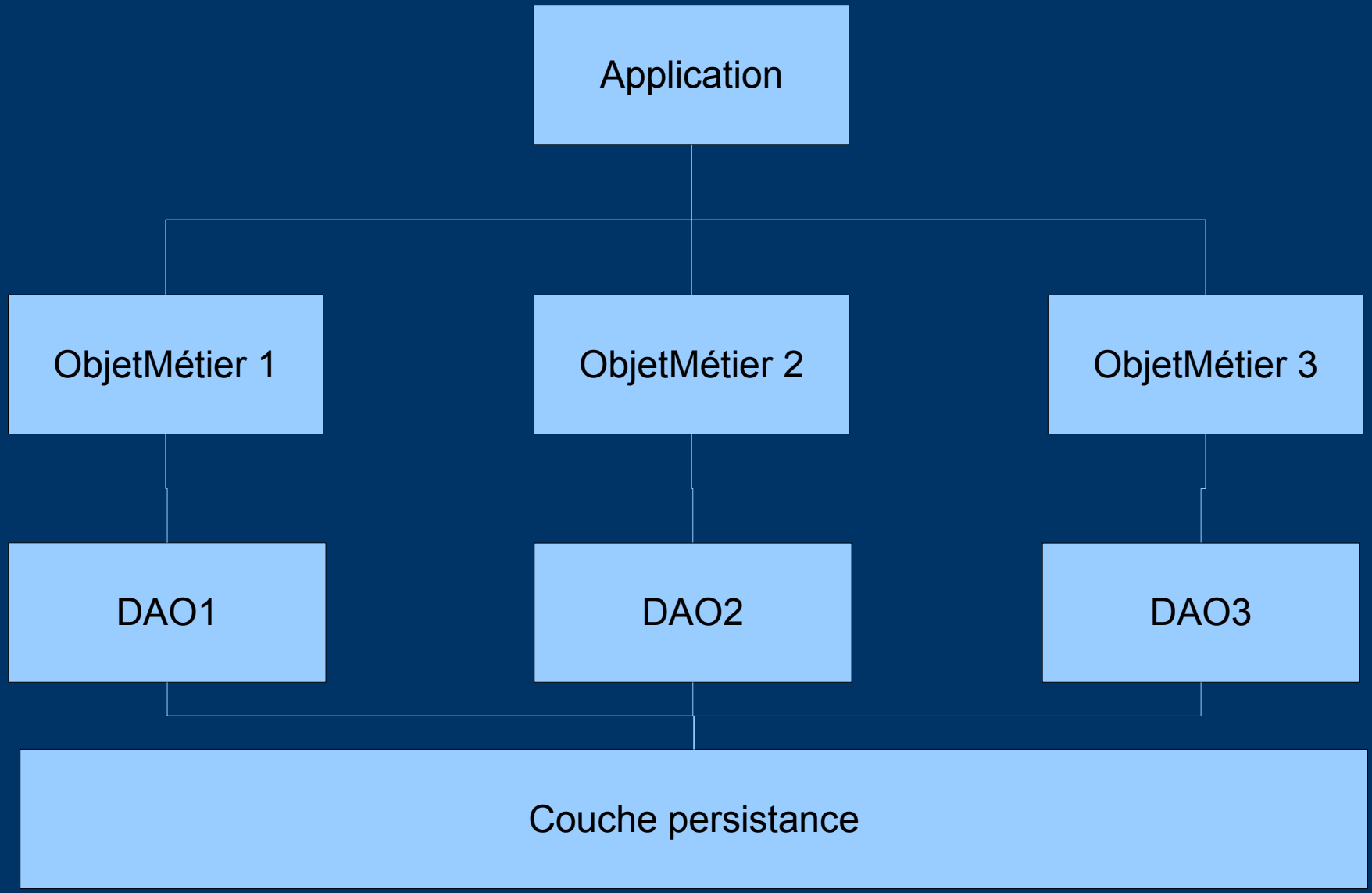
Bruno Mermet
Université du Havre
Dernière mise à jour : décembre 2009

Introduction

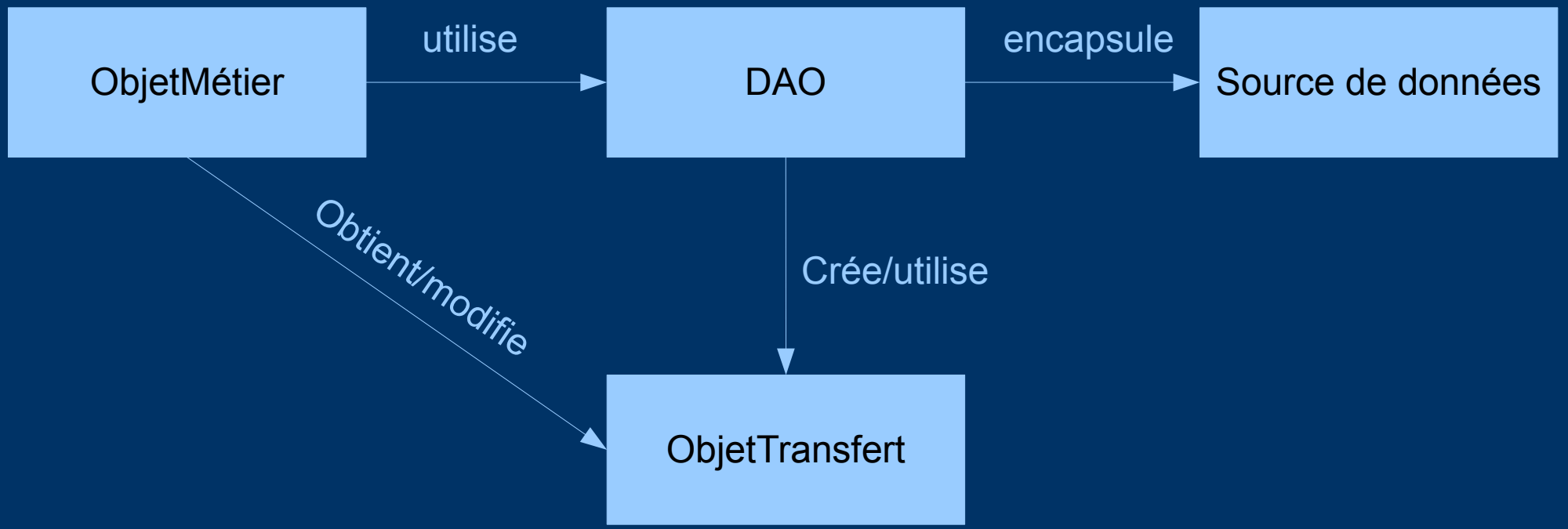
- Motivations
  - De nombreuses applications ont besoin de gérer des données persistantes
  - La persistance peut être géré par des moyens :
    - Différents selon la configuration de déploiement de l'appli
    - Évoluant dans le temps
- Principe
  - Isoler la couche de persistance du reste de l'appli
- Intérêt
  - L'essentiel de l'appli est indépendant de la persistance
- Référence

http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html

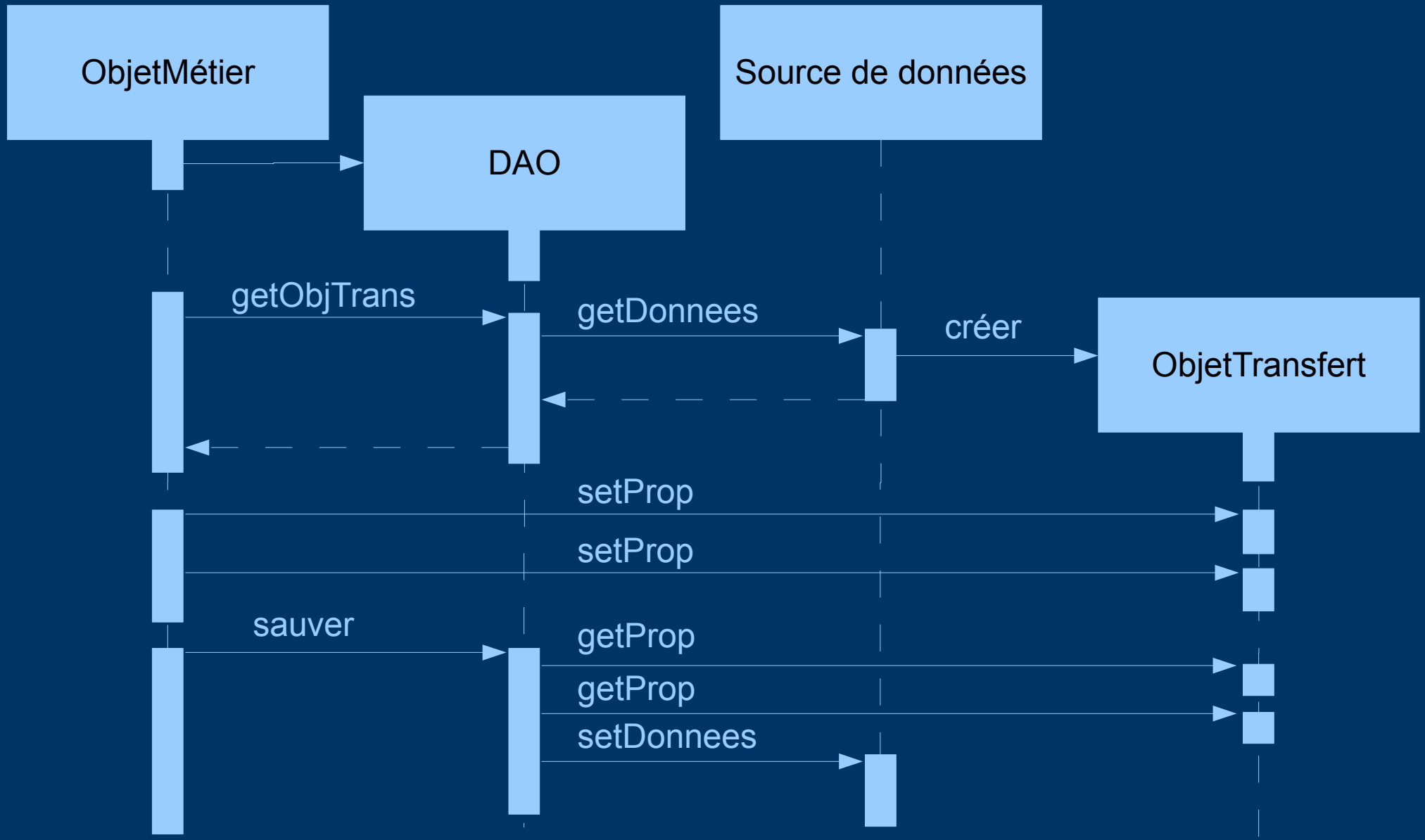
Principe général
Vue abstraite



Solution de base
Structure générale



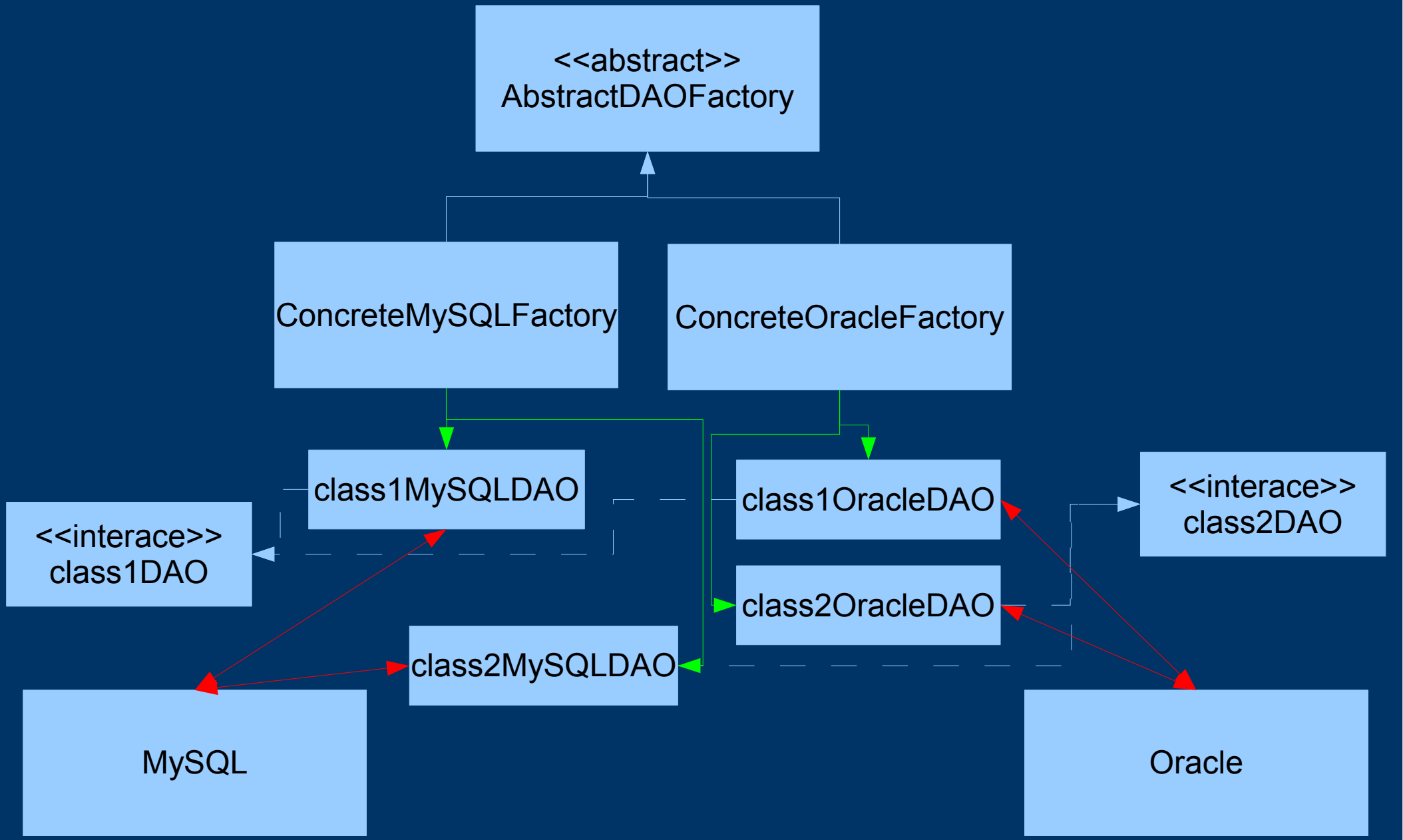
Solution de base exemple d'échanges de message



Remarques générales sur le modèle

- Un objet DAO par classe métier (pas dit dans le document de référence) et non par objet
  - Objets DAO peuvent implanter le pattern *Singleton*
- Couche de persistance uniforme pour un projet
  - Objets DAO instanciés par famille
  - Utilisation possible du pattern *Abstract Factory*
- Objet *Transfert* pas indispensable

Exemple d'instanciation (1)



Exemple d'instanciation (2)

```
Public abstract class AbstractDAOFactory {
        public abstract Class1DAO getClass1DAO();
        public abstract Class21DAO getClass2DAO();
        public static getAbstractDAOFactory(int sgbd) {
                switch(sgbd) {
                        case Mysql:
                                return new ConcreteMySQLFactory();
                                break;
                        case Moracle:
                                return new ConcreteOracleFactory();
                                break;
                }
        }
}

public class ConcreteMySQLFactory extends AbstractDAODactory {
        public Class1DAO getClass1DAO() {
                return Class1MySQLDAO.getInstance();
        public Class2DAO  getClass2DAO() {
                return Class2MySQLDAO.getInstance();
}
```

Exemple d'instanciation (3)

```
Public interface  Class1DAO {
	public Class1 load(int id);
	public void save(Class1 obj);
}

public class Class1MySQLDAO implements Class1DAO {
	private static instance = null;
	private Class1MySQLDAO() {
		// connexion à la base par exemple
	}
	public static Class1DAO getInstance() {
		if (instance == null) {
			instance = new Class1MySQLDAO();
		}
		return instance;
	}
	public Class1 load(int id) {
		// requête d'accès à la base
		return new Class1(params);
	}
	public void save(Class1 obj) {
		// requête d'update/insert suivant les cas
	}
}
```