Le mapping objet relationnel

Introduction au mapping objet relationnel avec Nhibernate

Cyril GRAUFFEL – IR3 – 2009/2010

Plan de présentation

1. Le modèle relationnel et le modèle objet
2. Le Mapping Objet Relationnel (ORM)
3. Introduction à NHibernate

Le modèle relationnel et le modèle objet

- Les problèmes de correspondance entre le modèle objet et le modèle relationnel:
  - Le modèle objet propose plus de fonctionnalités :
    - L'héritage, le polymorphisme
  - Les relations entre deux entités sont différentes
  - Les objets ne possèdent pas d'identifiant unique contrairement au modèle relationnel

Le modèle relationnel et le modèle objet

- Pour accéder à la base de données :
  - L'API JDBC pour Java;
  - ADO.NET pour .Net;
- Inconvénients :
  - Nécessite l'écriture de nombreuses lignes de codes répétitives
  - La liaison entre les objets et les tables est un travail de bas niveau

2. Le Mapping Objet Relationnel

- Plan :

  a. Définition

  b. Avantages / Inconvénients

  c. Les critères pour choisir un outil

  d. Les outils existants

Le Mapping Objet Relationnel

- Définition :
  - Concept permettant de connecter un modèle objet à un modèle relationnel.
  - Couche qui va interagir entre l'application et la base de données.
- Pourquoi utiliser ce concept?
  - Pas besoin de connaître l'ensemble des tables et des champs de la base de données
  - Faire abstraction de toute la partie SQL d'une application.

Le Mapping Objet Relationnel

- Avantages / Inconvénients :

  - Avantages :
    - Gain de temps au niveau du développement d'une application.
    - Abstraction de toute la partie SQL.
    - La portabilité de l'application d'un point de vue SGBD

  - Inconvénients :
    - L'optimisation des outils proposés
    - La difficulté à maîtriser les outils.

Le Mapping Objet Relationnel

- Les critères pour choisir un outil de mapping objet relationnel :
  - La facilité du mapping des tables avec les classes, des champs avec les attributs.
  - Les fonctionnalités de bases des modèles relationnel et objet.
  - Les performances et optimisations proposées : gestion du cache, chargement différé.
  - Les fonctionnalités avancées : gestion des sessions, des transactions.

Le Mapping Objet Relationnel

- Les différents outils existant :
  - Python : SQLAlchemy
  - Java : TopLink, Hibernate
  - C# : Nhibernate, DLinq

Introduction à NHibernate

a. Présentation de l'outil

b. La mise en place de Nhibernate
   a. Le fichier de configuration
   b. Le mapping
   c. Gestion des sessions et des transactions
   d. Exemple

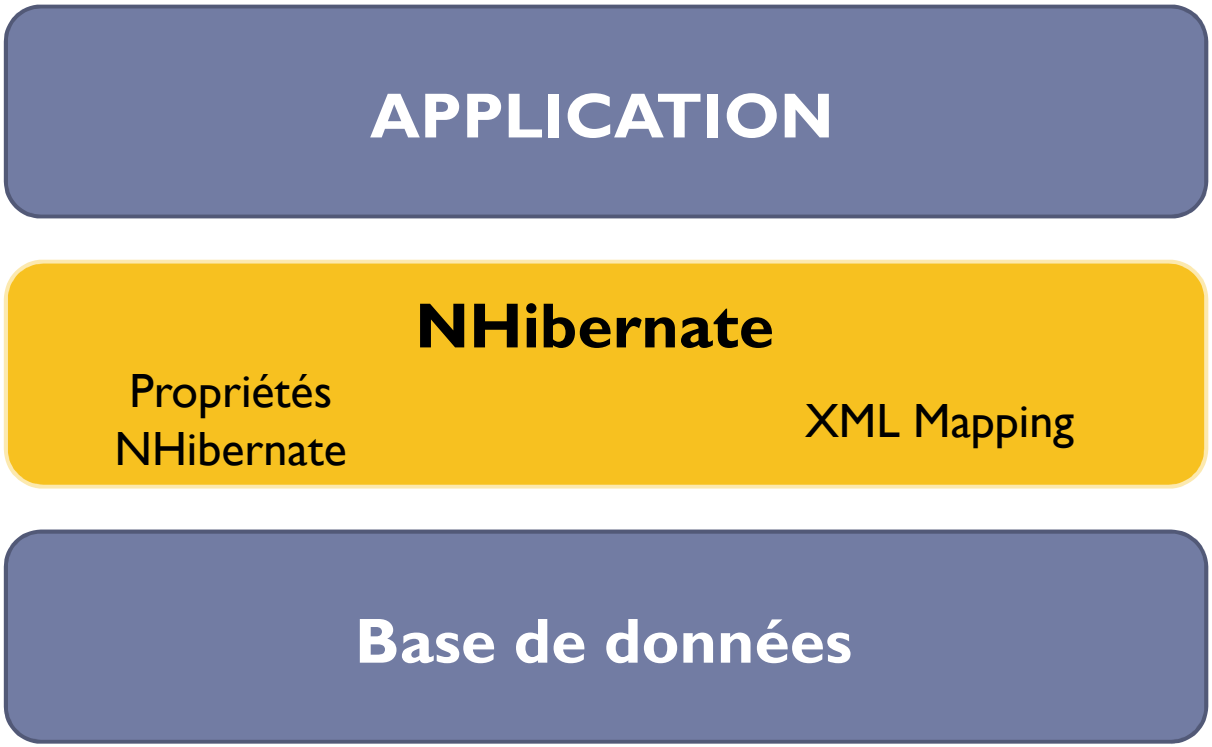
c. Fonctions avancées
   - L'héritage, gestion de performances, langage de requêtes

Présentation de l'outil

- ORM pour C#;
- Issue de l'ORM pour Java : Hibernate;
- Historique :
  - 2005 – 2006 : projet de JBoss.
  - Depuis 2006 : le support est assuré par la communauté NHibernate.

Présentation de l'outil

- Architecture :



La mise en place de NHibernate

- Comment ça marche?
  - 1 fichier de configuration qui regroupe l'ensemble des informations de la base de données.
  - 1 fichier xml par objet, représentant la correspondance entre la table et l'objet.

La mise en place de NHibernate

- Le fichier de configuration regroupe l'ensemble des informations de la base de données :
  - Le type de la base de données;
  - Le type de langage SQL à générer pour dialoguer avec la base de données;
  - La chaine de connexion à la base de données;

La mise en place de NHibernate

- Exemple :

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <section name="hibernate-configuration"
    type="NHibernate.Cfg.ConfigurationSectionHandler, NHibernate" />
  </configSections>
  <hibernate-configuration xmlns="urn:nhibernate-configuration-2.2">
    <session-factory>
      <property name="connection.provider">
        NHibernate.Connection.DriverConnectionProvider
      </property>
      <property name="connection.driver_class">
        NHibernate.Driver.MySqlDataDriver
      </property>
        …
```

La mise en place de NHibernate

```
…
<property name="connection.connection_string">
        Database=nhibernate;Data Source=127.0.0.1;User Id=root;Password=
      </property>
      <property name="dialect">
        NHibernate.Dialect.MySQLDialect
      </property>
      <property name="proxyfactory.factory_class">
        NHibernate.ByteCode.Castle.ProxyFactoryFactory, NHibernate.ByteCode.Castle
      </property>
      <property name="show_sql">true</property>
    </session-factory>
  </hibernate-configuration>
</configuration>
```

La mise en place de NHibernate

- Le mapping fait l'association entre un objet et une table de la base de données.

- Plusieurs étapes :

  - Création de la classe de l'objet;
    - [Objet].cs

  - Création du fichier de mapping;
    - [Objet].hbm.xml;

La mise en place de NHibernate

- Création de la classe : Parking.cs

```
public class Parking
   {
      private int id;
      private string name;

      //Constrcuteur
      ….

      public virtual int Id
      {
         get { return id; }
         set { id = value; }
      }

      public virtual string Name
      {
         get { return name; }
         set { name = value; }
      }
   }
```

La mise en place de NHibernate

- Création du fichier de mapping : Parking.hbm.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<hibernate-mapping xmlns="urn:nhibernate-mapping-2.2">
  <class name="Test.Parking,Test" table="parking“ >
    <id name="Id" column="id" type="Int32">
      <generator class="native" />
    </id>
    <property name="Name" column="parkingName" type="String"/>
  </class>
</hibernate-mapping>
```

La mise en place de NHibernate

- Gestion des sessions:
  - On récupère la configuration de la base de données
    - Configuration;
  - On récupère la factory des sessions
    - ISessionFactory;
  - On récupère une session de la factory
    - ISession;

La mise en place de NHibernate

- L'objet NHibernate.Cfg.Configuration
  - Informations de connection à la base de données et les propriétés de Nhibernate (App.config)

```
Configuration conf = new Configuration() .Configure("App.config");
```

  - Informations de mapping : les différents fichiers de mapping

```
Configuration conf = conf.AddAssembly("[objet].hbm.xml");
```

La mise en place de NHibernate

- Comment obtenir une session ?
  - Récupérer une instance de ISessionFactory de la configuration
    - ISessionFactory factory = conf. BuildSessionFactory();
  - Récupérer une instance de ISession de la Factory
    - ISession session = factory.OpenSession();

- La session :
  - 3 Actions :
    - Save
    - Update
    - Delete
  - Exemple : session.Save(objet);

La mise en place de NHibernate

- Que permet une transaction?
  - Valider une action de la session : Commit();
  - Annuler une action de la session : Rollback();

```
try{
    ISession session = …
    ITransaction tx = session.BeginTransaction();
    …
    tx.Commit();
    session.Close();
} catch(Exception ex){
    tx.Rollback();
}
```

Exemple avec NHibernate

- Exemple simple de l'utilisation de NHibernate :
  - Création d'un objet Parking et sauvegarde de ce dernier en base de données;
  - Même exemple avec une autre base de données;

Fonctions avancées

- Pour l'héritage, nous avons 3 solutions :
  - Table per class hierarchy : balise <subclass>
    - Une classe par type d'entité mais une seule et même table dans la base de données.
    - Un champ discriminant dans la table.
  - Table per subclass : balise <joined-subclass>
    - Autant de tables que d'entités.
  - Table per concrete class : mapping simple
    - Autant de table que d'entités concrètes, mais rarement utilisé due à sa limitation.

Fonctions avancées

- Gestion des performances :
  - Gestion des collections d'objets utilise le **lazy-loading**.
    - Attribut **lazy** avec la valeur **true** ou **false**;
  - Ne charge uniquement les objets lors de leur accès.
- Qu'est ce que l'on y gagne ?
  - Temps de chargement des données d'une application.
  - Économie de ressource mémoire et de charge serveur coté SGBD.

Fonctions avancées

- Implémente son propre langage d'interrogation, 2 solutions :
  - Construction de critères et de filtres
    - Grâce aux objets **ICriteria**
  - Utilisation d'un véritable langage d'interrogation objet :
    - Le HQL (Hibernate Query Language)
    - Utilisation de l'objet **IQuery**
    - Langage de requête totalement orienté objet.

Conclusion

- La correspondance entre le modèle objet et le modèle relationnel n'est pas si facile que l'on peut croire

- Les outils ORM:
  - Facilite cette correspondance.
  - Gain de temps au niveau du développement
  - Portabilité de l'application.

Bibliographie

- Nombreux sites parmis lesquels :
  - Livre blanc sur les **ORM** : http://www.dotnetguru.org/articles/Persistance/livreblanc/ormapping.htm
  - Site du framework **Hibernate** et **NHibernate** : http://www.hibernate.org/
  - Blog :
    - http://www.technologies-dotnet.be
    - http://blog.olivier-duval.info/
  - http://fr.wikipedia.org/wiki/Mapping_objet-relationnel
  - http://www.developpez.com

Question