iBATIS, Hibernate, and JPA: Which is right for you?

Object-relational mapping solutions compared

By K. L. Nitin, Ananya S., Mahalakshmi K., and S. Sangeetha, JavaWorld.com, 07/15/08

JPA uses a combination of annotation- and XML-based configuration. The XML file used for this purpose is persistence.xml, which is located in the application's `META-INF` directory. This file defines all the persistence units that are used by this application. Each persistence unit defines all the entity classes that are mapped to a single database. The persistence.xml file for the Employee application is shown in Listing 11.

Listing 11. persistence.xml

```
<persistence>
 <persistence-unit name="EmployeePU" transaction-type="RESOU
    <class>com.trial.Employee</class>
    <properties>
      <property name="toplink.jdbc.url" value="jdbc:mysql://
      <property name="toplink.jdbc.user" value="root"/>
      <property name="toplink.jdbc.driver" value="com.mysql.
      <property name="toplink.jdbc.password" value="infosys"
    </properties>
  </persistence-unit>
</persistence>
```

The persistence.xml file defines a persistence unit named `EmployeePU`. The configuration for the corresponding database is also included in the persistence unit. An application can have multiple persistence units that relate to different databases.

To summarize, JPA provides a standard POJO-based ORM solution for both Java SE and Java EE applications. It uses entity classes, entity managers, and persistence units to map and persist the domain objects and the tables in the database.

When to use JPA

JPA should be used when you need a standard Java-based persistence solution. JPA supports inheritance and polymorphism, both features of object-oriented programming. The downside of JPA is that it requires a provider that implements it. These vendor-specific tools also provide certain other features that are not defined as part of the JPA specification. One such feature is support for caching, which is not clearly defined in JPA but is well supported by Hibernate, one of the most popular frameworks that implements JPA. Also, JPA is defined to work with relational databases only. If your persistence solution needs to be extended to other types of data stores, like XML databases, then JPA is not the answer to your persistence problem.

pros and cons. Let's consider several parameters that will help you decide the best possible option among them for your requirements.

Simplicity

In the development of many applications, time is a major constraint, especially when team members need to be trained to use a particular framework. In such a scenario, iBATIS is the best option. It is the simplest of the three frameworks, because it only requires knowledge of SQL.

Complete ORM solution

Traditional ORM solutions like Hibernate and JPA should be used to leverage complete object-relational mapping. Hibernate and JPA map Java objects directly to database tables, whereas iBATIS maps Java objects to the results of SQL queries. In some applications, the objects in the domain model are designed according to the business logic and might not completely map to the data model. In such a scenario, iBATIS is the right choice.

Dependence on SQL

There has always been a demarcation between the people who are well versed in Java and those who are comfortable with SQL. For a proficient Java programmer who wants to use a persistence framework without much interaction with SQL, Hibernate is the best option, as it generates efficient SQL queries at runtime. However, if you want complete control over database querying using stored procedures, then iBATIS is the recommended solution. JPA also supports SQL through the `createNativeQuery()` method of the `EntityManager`.

Support for query languages

iBATIS strongly supports SQL, while Hibernate and JPA use their own query languages (HQL and JPQL, respectively), which are similar to SQL.

Performance

An application must perform well in order to succeed. Hibernate improves performance by providing caching facilities that help with faster retrieval of data from the database. iBATIS uses SQL queries that can be fine-tuned for better performance. The performance of JPA depends on that of the vendor implementation. The choice is particular to each application.

Portability across different relational databases

Sometimes, you will need to change the relational database that your application uses. If you use Hibernate as your persistence solution, then this issue is easily resolved, as it uses a database dialect property in the configuration file. Porting from one database to another is simply a matter of changing the dialect property to the appropriate value. Hibernate uses this property as a guide to generate SQL code that is specific to the given database.

As previously mentioned, iBATIS requires you to write your own SQL code; thus, an iBATIS application's portability is dependent on that SQL. If the queries are written using portable SQL, then iBATIS is also portable across different relational databases. On the other hand, the portability of JPA depends on the vendor implementation that is being used. JPA is portable across different implementations, like Hibernate and TopLink Essentials. So, if no vendor-specific features are used by the application, portability becomes a trivial issue.

Community support and documentation

Hibernate is a clear winner in this aspect. There are many Hibernate-focused forums where members actively respond to queries. iBATIS and JPA are catching up slowly in this regard.

Portability across non-Java platforms

iBATIS supports .Net and Ruby on Rails. Hibernate provides a persistence solution for .Net in the form of NHibernate. JPA, being a Java-specific API, obviously does not support any non-Java platform.

This comparison is summarized in Table 1.

Table 1. Persistence solutions compared

Features	**iBATIS**	**Hibernate**	**JPA**
Simplicity	Best	Good	Good
Complete ORM solution	Average	Best	Best
Adaptability to data model changes	Good	Average	Average
Complexity	Best	Average	Average
Dependence on SQL	Good	Average	Average
Performance	Best	Best	N/A*
Portability across different relational databases	Average	Best	N/A*
Portability to non-Java platforms	Best	Good	Not Supported
Community support and documentation	Average	Good	Good

* *The features supported by JPA are dependent on the persistence provider and the end result may vary accordingly.*

objects and relational models. However, iBATIS provides you with complete control over queries. Hibernate provides a complete ORM solution, but offers you no control over the queries. Hibernate is very popular and a large and active community provides support for new users. JPA also provides a complete ORM solution, and provides support for object-oriented programming features like inheritance and polymorphism, but its performance depends on the persistence provider.

The choice of a particular persistence mechanism is a matter of weighing all of the features discussed in the comparison section of this article. For most developers the decision will be made based on whether you require complete control over SQL for your application, need to auto-generate SQL, or just want an easy-to-program complete ORM solution.

Acknowledgements

The authors would like to sincerely acknowledge S. V. Subrahmanya (SVS) for his valuable guidance and support.

About the author

S. Sangeetha works as a technical architect at the E-Commerce Research Labs at Infosys Technologies. She has close to 10 years of experience in design and development of Java and Java EE applications. She has co-authored a book on Java EE architecture and also has written articles for JavaWorld and Java.net.

K. L. Nitin works at the E-Commerce Research Labs at Infosys Technologies. He is involved in the design and development of Java EE applications using Hibernate and JPA, and has expertise on agile frameworks like Ruby on Rails.

Ananya S. works at the E-Commerce Research Labs at Infosys Technologies. She has been involved in the design, development, and deployment of Java EE applications using JPA and iBATIS. She also has experience in programming with Ruby and Ruby on Rails.

Mahalakshmi K. works at the E-Commerce Research Labs at Infosys Technologies. She has experience in Java EE technologies and database programming. She is involved in the design and development of Java EE applications using Hibernate and JPA. She has also worked on application development using the Ruby on Rails framework.

< Prev 1 2 3 4 5 6 7

Print Feedback Like

DISQUS

Archived Discussions (Read only)

Forum migration complete *By Athen*

Forum migration update *By Athen*

Resources

Learn more about the three technologies discussed in this article from the project homepages:

- Hibernate
- iBATIS
- The Java Persistence API

"Get started with Hibernate" (Christian Bauer and Gavin King, JavaWorld, October 2004) is a short introduction to Hibernate written by its creator, Gavin King. *Excerpted from Hibernate in Action; Manning 2004.*)

Java Persistence with Hibernate (Christian Bauer and Gavin King; Manning, November 2006) is the updated second edition of *Hibernate in Action*. Also see iBATIS in Action (Clinton Begin, Brandon Goodin, and Larry Meadors, 2007).

For broader coverage of Java persistence solutions including JDBC, OpenJPA, and pureQuery, see Persistence in the Enterprise: A Guide to Persistence Technologies (Roland Barcia, Geoffrey Hambrick, Kyle Brown, Robert Peterson, Kulvir Singh Bhogal; IBM Press, May 2008).

Ted Neward introduces the so-called object-relational impedance mismatch in his blog post "The Vietnam of computer science."

"Flexible reporting with JasperReports and iBatis" (Scott Monahan, JavaWorld, December 2007) is a hands-on introduction to the iBatis Data Mapper framework.

"Understanding the Java Persistence API" (Aditi Das, JavaWorld, January 2008) is a two-part introduction to Java-platform persistence with OpenJPA.

Java-source.net lists a roundup of open source persistence frameworks for Java.

Visit the JavaWorld Java Enterprise Edition research center for more articles about enterprise data management and Java persistence solutions.

Also see Network World's IT Buyer's Guides: Side-by-side comparison of hundreds of