

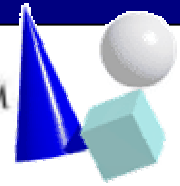
A Critical Analysis of JDO in the Context of J2EE

Axel Korthaus, Matthias Merz

Department of Information Systems

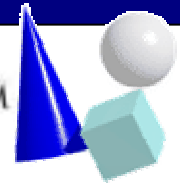
University of Mannheim

{korthaus|merz}@wifo3.uni-mannheim.de



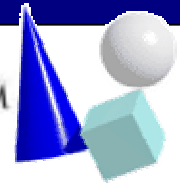
Agenda

1. Enterprise JavaBeans (EJB)
2. Java Data Objects (JDO)
3. Comparison of EJB (BMP, CMP) and JDO
in the J2EE Context
4. Performance Measurement
5. Conclusion



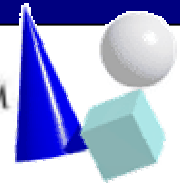
Agenda

1. Enterprise JavaBeans (EJB)
2. Java Data Objects (JDO)
3. Comparison of EJB (BMP, CMP) and JDO
in the J2EE Context
4. Performance Measurement
5. Conclusion



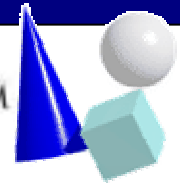
Enterprise JavaBeans (EJB)

- Server-side component architecture
- Frees the developer from having to deal with code that handles transactional behaviour, security, connection pooling, threading etc.
- Developers can fully concentrate on programming only the business logic
- Complex process of getting an Enterprise Java Bean running (developing source, deployment descriptor, compiling, packing and deploying)



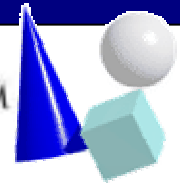
Enterprise JavaBeans (EJB)

- Bean-Managed Persistence (BMP)
 - Bean developer is responsible for writing code to save the bean's state
- Container-Managed Persistence (CMP)
 - No persistence code has to be written by the developer
 - Deployment descriptor
 - Bean implementation class is independent of the concrete data store



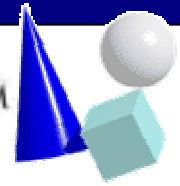
Agenda

1. Enterprise JavaBeans (EJB)
2. Java Data Objects (JDO)
3. Comparison of EJB (BMP, CMP) and JDO
in the J2EE Context
4. Performance Measurement
5. Conclusion



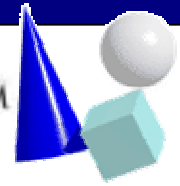
Java Data Objects (JDO)

- JDO 1.0 provides transparent and standardized access to persistent objects
- JDO provides persistence by reachability and also a lot of important features
- Enables the use of various data store types
- XML-based persistence descriptor
- Byte-code enhancer
- JDO Query Language (JDOQL) based on conventional Java operators



Agenda

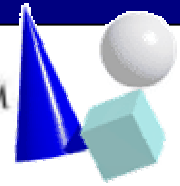
1. Enterprise JavaBeans (EJB)
2. Java Data Objects (JDO)
3. Comparison of EJB (BMP, CMP) and JDO in the J2EE Context
4. Performance Measurement
5. Conclusion



Comparison of BMP, CMP and JDO

Options to get persistence in the J2EE context:

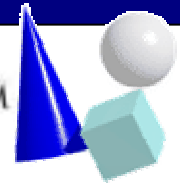
- Entity beans with CMP
- Entity beans with BMP (e.g. JDBC or JDO)
- Stateless session bean façade together with JDO or JDBC



Comparison of BMP, CMP and JDO

CMP vs. BMP

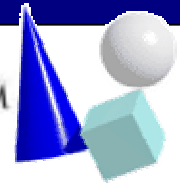
- CMP is “normally” the faster and better approach:
 - Container generates and manages persistence logic
 - If tuned properly, CMP entity beans are much higher performing than BMP entity beans
 - Container-managed relationships
- Some scenarios where BMP for use in an entity bean can be appropriate or might even be required



Comparison of BMP, CMP and JDO

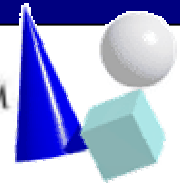
CMP vs. stateless session bean

- Stateless Session Beans using JDBC or JDO can be sufficient to have a more lightweight approach to persistence
- Using JDBC to implement the persistence layer has numerous drawbacks
- JDO appears to be the best technological approach to be combined with stateless session beans in a J2EE environment



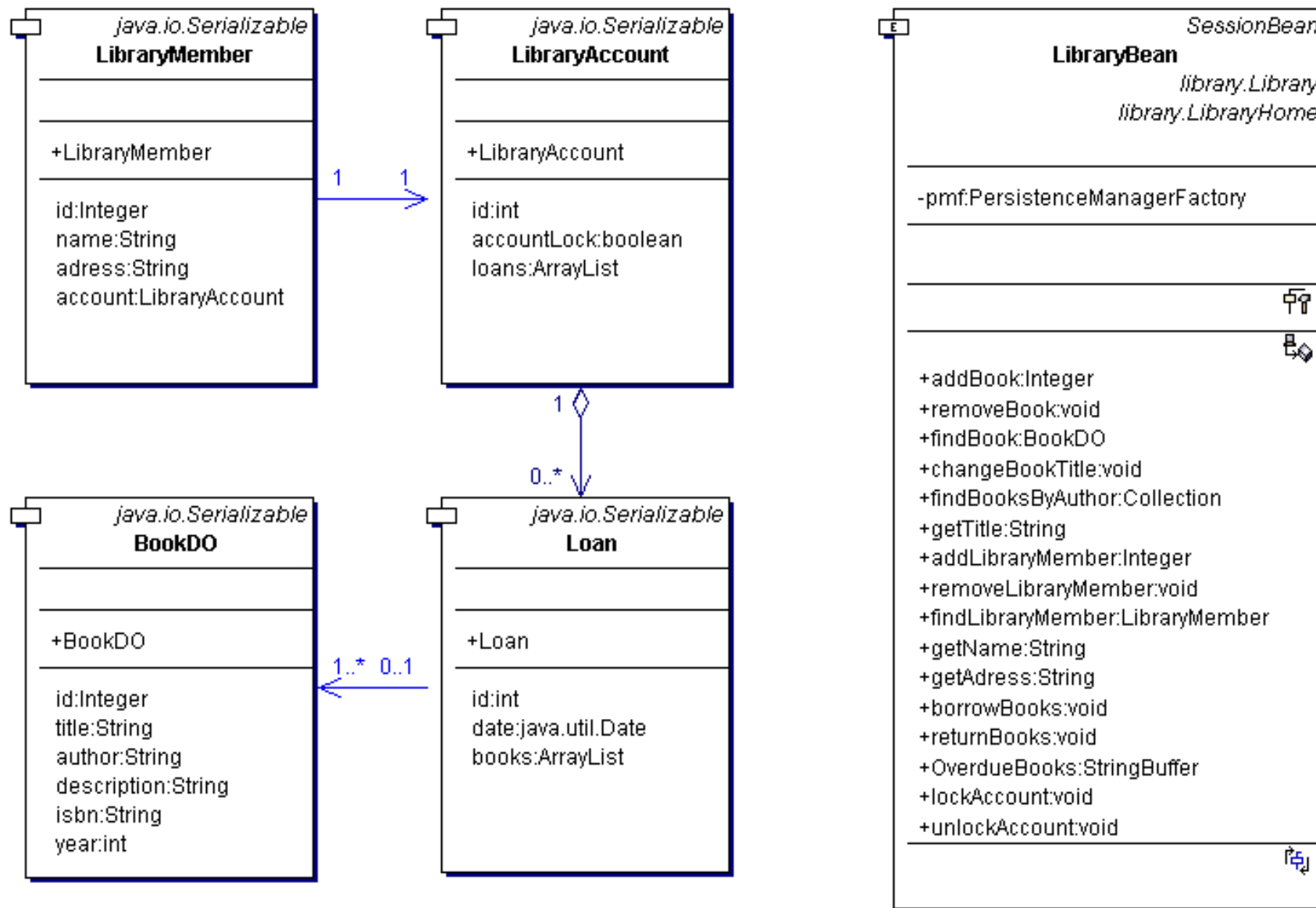
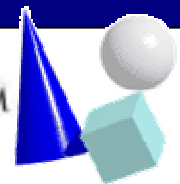
Comparison of BMP, CMP and JDO

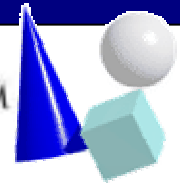
- Strengths of JDO
 - Avoids the heavyweight entity bean approach
 - Database and JDO-vendor independence
 - JDO objects can be built, run, and debugged outside the application server environment
- Deficiencies of JDO:
 - JDOQL (error-prone String filters)
 - Optional features
 - Byte code enhancement
 - No managed (inverse) relationship support



Agenda

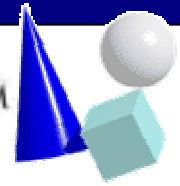
1. Enterprise JavaBeans (EJB)
2. Java Data Objects (JDO)
3. Comparison of EJB (BMP, CMP) and JDO
in the J2EE Context
4. Performance Measurement
5. Conclusion





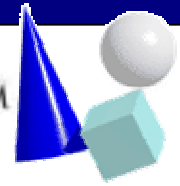
Performance Measurement

- JDO and CMP
 - A session bean façade encapsulates the business and persistence logic, which is the same for both implementations
 - The Data Transfer Object pattern is used to encapsulate all relevant attributes of a book
- CMP
 - The business objects are represented by entity beans (only local interfaces used)
- JDO
 - Plain Java objects, that had to be enhanced



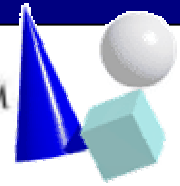
Performance Measurement

- A complex business method `OverdueBooks()` involves not only the retrieval of instances of one single class, but also requires navigation across associations.
- First, a query retrieves a collection of all members, then the corresponding account is fetched and, if existing, loan objects for the account are retrieved.
- If at least one loan object exists and the loan period has been exceeded, the corresponding books are fetched and added to the result object.



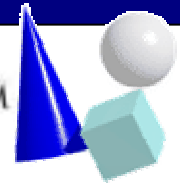
Performance Measurement

- JDO-based solution:
 - Average execution time: 61.3 sec
- CMP-based solution
 - Average execution time: 17.4 sec
- Calling `OverdueBooks()` twice:
 - JDO-based solution had an enormous speedup and now only needed 6 sec which must be due to intelligent caching



Agenda

1. Enterprise JavaBeans (EJB)
2. Java Data Objects (JDO)
3. Comparison of EJB (BMP, CMP) and JDO
in the J2EE Context
4. Performance Measurement
5. Conclusion



Conclusion

- Best solutions regarding performance and simplicity:
 - Entity beans with CMP (version ≥ 2.0) or
 - Session bean façade approach combined with JDO.
- If automatically managed bi-directional relationships can be dispensed with, JDO will be a good choice
- The use of JDO is especially worth while if frequent accesses to the same objects occur