Database Access

Rick Cattell
JavaSoft
1. Relational database connectivity: JDBC™
2. Higher-level relational APIs and tools
3. Object-oriented databases
4. Partners and summary
Overview

- Java™ language attractive for database applications
- Increasing interest for general applications, not just Internet
- Establish critical mass, libraries, tools for Java
- Have moved quickly with basic database connectivity
- Not trying to “do it all ourselves”: leverage partners, existing APIs
Why Java™ is Attractive for DB Applications

- Zero installation/administration
- Platform-independence
- Secure enterprise applications
- Internet connection
- Better language (than VB, C++, COBOL)
- Increased productivity (no clobbers, memory leaks, void* ...)
- However: the Java language alone is not enough
New Components for Enterprise Apps

Next two panels cover work in all of these areas
Database Connectivity

1. Simple JDBC API
2. Higher-Level APIs & Tools
3. ODMG Mapping
1. SQL Interface: JDBC

JDBC API

JDBC test suite

We provide JDBC pieces

Partners sell other pieces

Java code

Other code

DBMS

DBMS

DBMS

Published protocol

Proprietary protocol

ODBC Driver Manager

Driver 1

Driver 2

Listener

ODBC Driver Manager

Driver

Direct JDBC Driver

JDBC-Net Bridge

JDBC-ODBC Bridge

JDBC Driver Manager

ODBC Driver

DBMS
JDBC Product

- Existing C database APIs not practical for Java
- Need solution yesterday; easy to leverage ODBC
- Synergy for ISVs producing connectivity and tools
- Partnership with Intersolv and others
- See http://splash.javasoft.com/jdbc for more info

<table>
<thead>
<tr>
<th>March</th>
<th>June</th>
<th>September</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha Spec</td>
<td>FCS Spec, DM</td>
<td>More drivers available</td>
</tr>
<tr>
<td>Alpha DM</td>
<td>Drivers available</td>
<td>FCS ODBC bridge</td>
</tr>
<tr>
<td>Endorsees</td>
<td>Beta bridge, tests</td>
<td>FCS test suite</td>
</tr>
</tbody>
</table>
JDBC Design

• Generally patterned after ODBC and X/Open CLI
• Faithful to Java language with ease-of-use emphasis
• Low-level API; build other APIs on top
• Two kinds of users:
  – Programmers (need ease of use)
  – Programs (need completeness, performance)
• Two modes of operation:
  – Untrusted applets and drivers on Internet
  – Trusted code accessing company DBMS servers
JDBC Features

- Connection, Statement, ResultSet interfaces
- PreparedStatement and CallableStatement for compilation and stored procedures
- Metadata and dynamic access
- Dynamic driver selection and loading
- Database naming based on URLs; typically `jdbc:<subprotocol>:<identifier>`
- ... More details in Graham Hamilton’s talk
JDBC Example

Connection conn = 
   DriverManager.getConnection(
       "jdbc:odbc:sales");
Statement stmt =
   conn.createStatement();
ResultSet rs = stmt.executeQuery(
   "SELECT Name, Sales FROM Customers");
while (rs.next()) {
   String name = rs.getString("Name");
   int sales = rs.getInt("Sales");
   ...
2. Higher-Level Relational Tools and APIs

- Embedded SQL for Java language
- Integration with application-building tools
- Integration with 3-tier tools: RMI, CORBA
- Enterprise repository and administration tools
- Object/relational mapping
Object/Relational Mapping: Declarations

SQL: CREATE TABLE CUSTOMER (
    CUSTID INTEGER NOT NULL,
    ADDRESS VARCHAR(50),
    SALESREP INTEGER,
    PRIMARY KEY (CUSTID),
    FOREIGN KEY (SALESREP) REFERENCES SALES);

Java: class Customer {
    int CustID;
    String address;
    Sales salesRep
}

Automatically create Java class for each table in database schema (or vice versa), plus provide tools to embellish for many-to-one mappings, relationships, etc.
Object/Relational Mapping: Code

Transaction t = Transaction.create();
CustomerSet cs = CustomerSet.query(mycustomers);
Sales s = c.salesRep; /* follows foreign key */
c.address = newAddress; /* obtains write lock */
s.sales = s.sales + thisOrder;
t.commit(); /* writes c and s back to database */

Java transaction object, database object, etc;
transparently fetch/store Java objects from tables,
map foreign keys to Java references
3. Object Databases for Java

- Transparent persistence for Java objects
- Java ideal for object databases since “safe”
- Cached performance with more powerful data structures than relational
- More transparent and faster than “pickling”
- Not a JavaSoft product; working with ODMG consortium (Object Database Management Group): voting members GemStone, IBEX, O2, Object Design, Objectivity, POET, UniSQL, Versant
ODMG Timetable

- Formed working group, chaired by me
- Already reached agreement on initial specification
- Several vendors well along on implementation
- See [http://www.odmg.org/](http://www.odmg.org/) for more ODMG info

<table>
<thead>
<tr>
<th>April</th>
<th>June</th>
<th>December</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODMG WG Draft spec</td>
<td>Review spec Java hook?</td>
<td>Freeze spec Initial products</td>
</tr>
</tbody>
</table>
ODMG Java Binding

- ODMG class library: Database, transaction, query, and collection classes
- Persistence orthogonal to type: Existing classes can have persistent and transient instances
- Persistence by reachability: All objects reachable from designated named root database objects become persistent on transaction.commit
- ODMG object model: Integrity constraints, can share data with persistent C++ and Smalltalk programs
- Full database functionality: SQL2 superset OQL, transaction per thread, object locking
ODMG Example

```java
Database.open("University", DatabaseReadWrite);
Transaction t = new Transaction;
...
SetOfStudent mathematicians = Students.query
    "exists s in this.takes: s.course.name="math" ");
Student joe = Students.select("id = 4132");
Professor oldAdvisor = joe.advisor;
joe.advisor = newAdvisor;
joe.address = "123 Main Street";
t.commit()
...
```
## 4. Database Partners

<table>
<thead>
<tr>
<th>Companies</th>
<th>Products this year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bluestone</td>
<td>Sapphire/Web database app-builder</td>
</tr>
<tr>
<td>Borland</td>
<td>InterClient JDBC driver for InterBase, Latté app-builder</td>
</tr>
<tr>
<td>BulletProof</td>
<td>JAGG JDBC/ODBC drivers and JDesignerPro database app-builder</td>
</tr>
<tr>
<td>IBM</td>
<td>DB2 JDBC driver. CGI scripting, app-building tools, Java stored procedures/user functions / triggers, Data Access Builder and Visual Age object/relational mapping on JDBC</td>
</tr>
</tbody>
</table>
## Database Partners

<table>
<thead>
<tr>
<th>Companies</th>
<th>Product this year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Imaginary</td>
<td>Postgres95 and mSQL JDBC drivers, mSQL DBMS</td>
</tr>
<tr>
<td>Informix</td>
<td>Database app-building tools, DBMS</td>
</tr>
<tr>
<td>Intersolv</td>
<td>SequeLink JDBC-net driver, JDBC drivers, ODBC drivers, joint development of JDBC/ODBC bridge and test suites</td>
</tr>
<tr>
<td>O2 Technology</td>
<td>Object database, object/relational mapping</td>
</tr>
<tr>
<td>Object Design</td>
<td>Object database, object/relational mapping, JDBC driver</td>
</tr>
<tr>
<td>Open Horizon</td>
<td>Connection JDBC-net driver, security services, directory services, TP services</td>
</tr>
<tr>
<td>OpenLink</td>
<td>JDBC drivers</td>
</tr>
<tr>
<td>Oracle</td>
<td>Database application building tools, DBMS</td>
</tr>
<tr>
<td>POET</td>
<td>Object Database</td>
</tr>
</tbody>
</table>
## Database Partners

<table>
<thead>
<tr>
<th>Companies</th>
<th>Product this year</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAS</td>
<td>Java JDBC driver for Share*Net</td>
</tr>
<tr>
<td>SCO</td>
<td>SQL-Retriever JDBC/ODBC drivers</td>
</tr>
<tr>
<td>Spider Software</td>
<td>NetDynamics web/database application builder for JDBC and ODBC</td>
</tr>
<tr>
<td>Sybase</td>
<td>Optima++ Java database application development tools, DBMS</td>
</tr>
<tr>
<td>Symantec</td>
<td>JDBC drivers, Café app building tools and libraries</td>
</tr>
<tr>
<td>Versant</td>
<td>Object database</td>
</tr>
<tr>
<td>Visigenic Software</td>
<td>JDBC drivers, OpenChannel JDBC-net driver, ODBC drivers, source licensee partner</td>
</tr>
<tr>
<td>WebLogic</td>
<td>T3Server JDBC-net driver, dbKona API</td>
</tr>
</tbody>
</table>
Database Partners

<table>
<thead>
<tr>
<th>Companies</th>
<th>Product this year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working Set</td>
<td>DataRamp JDBC-net driver and server</td>
</tr>
<tr>
<td>XDB</td>
<td>JetConnectPro JDBC/ODBC drivers, DB GUI classes on AWT, JetStream RDBMS/gateway</td>
</tr>
</tbody>
</table>

For more partner info see JDBC web page and JavaOne panels on database access, engines, and tools.

Note: Listed product names are trademarks of their respective companies; Java, JDBC, and JDBC-Compliant are JavaSoft trademarks.
Summary

- JDBC: Quick to market, access legacy and relational database, direct use and generated code
- Higher-level: Object-relational mapping, embedded SQL, integration with tools/repository
- Object database: High-performance persistence for small footprint, embedded use
- Partners: Not trying to do it all ourselves; focus on APIs so pieces plug together