BMP vs. CMP in Entity EJBs

Acronyms Gone Wild (AGW)

Dave Landers
dave.landers@bea.com
BEA Systems, Inc.
Review

• Enterprise Java Beans
  – Multi-tier Component Architecture

• Two kinds of EJBs
  – Session Beans
    • Behaviors & Business Logic
    • Stateless or Stateful
  – Entity Beans
    • Data Model
    • Persisted to Database (usually)
Entity EJB Persistence

- Entity EJB represents data
  - Often thought of as a row in Relational Database (RDBMS) table
    - Although it may span multiple rows, tables
    - May not be RDBMS at all
- Data is persistent
  - Outlives the client
  - Survives server crash
- Entity object is a cached view of the persistent store
Persistence

- Entity instance state must be synchronized with persistent store on "Life-Cycle" boundaries
  - Load, Store called before and after every method
    - If transactional, load only at first method call for instance
    - If transactional, store called at commit
Entity Bean Life Cycle

- create
- load
- store
- remove
- finders
Handling Persistence

• Two ways to do the persistence:
  – Bean Managed Persistence (BMP)
    • You write the code
  – Container Managed Persistence (CMP)
    • Container handles saving the data
Bean Managed Persistence

• The Bean Developer writes persistence and finder code
• Usually JDBC / SQL code in EntityBean implementation
BMP Advantages

• Developer has full control of persistence mechanics
  – Deployer has little control, unless developer has provided it
• Lots of options
  – Non-RDBMS and Legacy systems
  – Multiple databases, tables, etc.
  – Aggregate / Coarse-Grained Entities
  – Complex finder methods
BMP Disadvantages

• May be tied to a single database
  – Use database specific features
• Lots of detailed code to write
  – Error prone
  – Boring
  – No check on Table and Column names until Runtime
  – Often uses “CAP” model for code reuse
• Copy And Paste
Container Managed Persistence

- Container persists bean for you
- Persistence specified at deployment rather than development
  - Developer provides mapping information to deployer (i.e. documentation)
CMP Advantages

• Easy
• Straightforward
• Fewer Errors
• Containers often check Table and Column mappings at Deploy time
• More Portable to multiple databases
CMP Disadvantages

• Stuck with container’s idea of persistence
• Less flexibility
• Fewer advanced features
• Database portability limited by container features
• Container portability limited by features you require
• Not for “Control Freaks”
CMP Support

• Most Application Server vendors support Basic CMP
  – RDBMS
  – One Bean, One Table
  – Simple field mapping (primitive and basic JDBC types)

• Feature-rich and Specialized containers often available
  – Object / Relational (O/R) Mapping
  – Object Databases (ODBMS)
Entity EJB Persistence History (so far...)

• EJB 1.0 (March 1998)
  – Entity Beans optional
  – CMP allowed, not specified how
  – Up to the server vendor
  – Deployment Descriptor as Serialized Object, not specified how to create it
History...

• EJB 1.1 (December 1999)
  – Entity support mandatory
  – CMP still not fully specified
    • cmp-field descriptor used to identify persistent fields
    • Not required to map to database
    • May use serialization
  – Some changes to support deployment choice of persistence
    • ejbCreate return - null vs void

  – Deployment Descriptor now in XML
History...

• EJB 2.0 (Public Draft 2)
  – CMP specified
    • Field mappings
    • Finder syntax
  – Aggregate or Coarse-Grained Entities supported
  – Entity-to-Entity relationships supported
...Future

- Next Steps (from 2.0 spec)
  - Pluggable Persistence Managers
  - Read-Only Entities with CMP
  - Aggregate operations for EJB QL finder methods
BMP Responsibility

- Implement persistence in:
  * `ejbCreate()`
  * `ejbRemove()`
  * `ejbFindByPrimaryKey()`
  * `ejbFind<foo>()`

* These methods are defined in the Home Interface
  - `create()`
  - `findByPrimaryKey()`
  - `remove()`
  - `findBy<foo>()`
BMP Responsibility

- **ejbCreate(createArgs...)**
  - SQL INSERT
  - Set state of object instance

- **ejbLoad()**
  - SQL SELECT on Primary Key
  - Set state of object instance

- **ejbStore()**
  - Get state from object instance
  - SQL UPDATE
BMP Responsibility

- `ejbRemove()`
  - SQL DELETE on Primary Key
- `ejbFindByPrimaryKey()`
  - SQL SELECT on Primary Key
- `ejbFind<foo>( ... )`
  - SQL SELECT on arguments
  - Return Primary Key or Collection of Primary Keys
BMP Persistence Example
CMP Responsibility

- **ejbCreate( createArgs...)**
  - Set state of object instance from createArgs
- **ejbLoad()**
  - Set state of object instance
- **ejbStore()**
  - Usually nothing to do
- **ejbRemove()**
  - Usually nothing to do
CMP Responsibility

• `ejbFindByPrimaryKey()`
  – Automatically generated

• `ejbFind<foo>( ... )`
  – Automatically generated

• Deployment
  – Set mapping of Bean to Table and Fields to Columns
  – Specify queries for finders
CMP Example - 1.1
CMP in EJB 2.0

• Container Managed Relationships
  – To other Entities
  – To Dependent Objects
  – Bidirectional or Unidirectional
  – One-to-One, One-to-Many or Many-to-Many

• Persistence Mappings for Fields and Relationships defined in Deployment Descriptors
CMP in EJB 2.0

- Finder Query Language (EJB QL)
  - Use in Deployment Descriptor
  - Also used by Select Methods
    - Not exposed to the client
    - Used by the bean’s business methods
    - Allows all SQL to be specified in deployment
    - Can return non-Entity types
  - Very SQL-like:
    - SELECT sss FROM ttt WHERE www
  - Can refer to other Entity beans (in same ejb-jar file)
CMP in EJB 2.0

- EJB Bean implementation class is abstract
  - Abstract get/set methods for fields
    - Used by Persistence Manager
    - May be exposed via Remote Interface
  - Abstract get/set methods for relationships
    - Used by Persistence Manager
    - Not exposed via Remote Interface
  - Implementation for Business Methods
    - Use abstract get/set methods to access persistent data
CMP Example - 2.0
Container Capabilities

• Default Containers
  – Basic field mapping and Finders
  – May provide database mapping checks at deployment
    • Table and column names and data types
Containers

• O/R Mapping Extensions
  – TOPLink (WebGain), CocoBase (Thought, Inc.), etc.
  – Available for many servers
  – Use as O/R Mapping for BMP or Container for CMP
  – Advanced mapping, relationships, finders, etc.
  – Performance enhancements and caching
Containers

• ODBMS Extensions
  – Versant, ObjectStore (eXcelon), etc.
  – Object persistence with BMP or Containers available for CMP

• Object Caches
  – Javlin (eXcelon)
CMP Sounds Great

What’s the catch?
Limitations of CMP

- Simple mapping support in EJB 1.1 containers
- Limited data type support in container
  - When mapping to advanced database types
  - Can you translate to/from more primitive types in business methods?
Limitations of CMP

• Coarse-grained Entities (pre-2.0)
  – Dependent object and entity relationship support

• Multiple databases or tables
  – Some containers may support this
  – Many do not

• Data aliasing
  – When multiple Entities map to the same database fields
Limitations of CMP

• Eager Loading
  – Of large, coarse-grained entities

• Legacy or non-RDBMS systems
  – When no container exists for such a system
Limitations of CMP

• Limited finder syntax
  – Especially in EJB 1.x
  – Need unavailable SQL features (ORDER BY, GROUP BY, COUNT(*), etc.) or advanced search capability (like Oracle’s InterMedia)

• Consider doing these things in a Session Bean with JDBC/ SQLJ
Can’t Live With CMP?

BMP Strategies to Ease the Pain
BMP Strategies

- Use base classes, examples, templates, and helper classes to reduce chances of errors
  - Don’t forget your Java and Object Oriented skills!
- Give Dependant objects their own persistence “manager”
  - Rather than entity doing the persistence
  - Promote reuse of these objects for multiple entities
BMP Strategies

• Defer options to Deployer
  – Deploy time choice of CMP or BMP implementation helper class
  – Use table- and field- mappings (in environment properties) to allow deploy-time tweaking of BMP code

• Read the CMP section of the 2.0 Spec
  – Try to implement your BMP consistent with that model
BMP Strategies

• Use a O/R Mapping Tool or ODBMS for Entity’s BMP Persistence Helper class
• Code generation tools for BMP?
• In an Application
  – Use CMP for simpler beans, BMP for others
SQLJ for BMP

- SQLJ lightens the bulk of the BMP code
- SQLJ can check SQL mappings at compile time (vs. runtime)

- Depend on Database vendor for compiler and runtime libraries
  - Available for DB2, Informix, Oracle
    - According to www.sqlj.org
SQLJ Example
What About Performance?
Performance

- Testing is difficult (depends on system requirements)
  - Load vs. Single client
  - Concurrent data access vs. private data per client
  - Reads vs. Writes
  - Transactions
Performance

• In general
  – BMP gives you control
    • Performance is more likely to be predictable when porting to other vendors
  – CMP has more opportunity to optimize
    • Container knowledge
    • Transaction state
    • Other beans
  – Vendor may supply CMP as convenience feature rather than performance feature
Performance

Tools such as O/R mapping can help
... or hurt

- Use container plug-in if available
- Or use tool to implement BMP
- May do lots of caching
  - Improves performance
  - More memory, sometimes GC issues if you’re not careful
- May take transaction responsibility from database
  - Can be good or bad, depending on database and system
Performance

– Transactions
  • Cost to begin and commit/rollback
    – Avoid unnecessary transactions
    – Many, short transactions are costly
  • More efficient access within transaction for multiple calls on one Entity
    – ejbLoad / ejbStore usually called once per transaction rather than once per method
  • Watch Entity Locking
    – Long transactions can hurt concurrency
    – Watch for Deadlock (due to access order and mixed transaction states)
References

• EJB Specifications
  http://java.sun.com/products.ejb/docs.html
• Other EJB references
  http://java.sun.com/ejb
• SQLJ
  http://www.sqlj.org
References

• Enterprise JavaBeans
  Richard Monson-Haefel
  • O’Reilly

• Mastering Enterprise Java Beans
  Ed Roman
  • John Wiley & Sons
  • http://www.theserverside.com
Other Talks

• Willy Farrell
  – Enterprise Java Beans
• Paul Giangarra
  – Distributed Flying Objects…
• Bryan Zarnett
  – J2EE Anti-Patterns
Thank You

Please fill out the Evaluation Form