EJB

Prashant Lamba, Niketu Parekh, and William Pugh

Some notes

- Sun’s j2ee ri fails in bad and mysterious ways under Java 1.5
- The j2ee 1.3.1 implementation is certified under Java 1.5
- Generally works under 1.4

What are EJB?

- Some RMI
- Some software component technology/terminology
- Ability to intercept calls and add system-level functionality
- Functionality that can be added
  - transactions, security, activation/passivation
  - …

Kinds of beans

- Entity beans
  - represents thing in persistent store
- Message-driven beans
  - listeners
  - we won’t be talking much about these today
- Session beans
  - stateless
  - stateful
Bean clients

- Clients can only get references to session and entity beans
- Can create messages that are processed by message driven beans

Session/Entity confusion

- Clients see same interfaces for both session and entities beans
- Some methods only make sense on entity beans, some only on session beans
- Some methods have different semantics on entity beans

Entity beans

- An entity bean is backed by an entry in a persistent store
  - removing an entity bean deletes the entry from the persistent store
- You can search for entity beans
  - by primary key, or there is even a query language

Stateful Session Beans

- Maintains state over a series of calls from a client
- Not shared between clients
- Can be (almost) silently dropped after timeout
  - not for storing important data
- Cannot search for session bean
Assumptions

- Initially, we’ll assume that all calls to EJB’s are from remote machines
  - at least, via RMI
  - assumption relaxed in EJB 2.0, we’ll talk about it later
- Clients never get to talk directly to a bean
  - would allow circumvention of security and transaction checks

Home sweet home

- Home interface allows beans to be created, located and removed
- Factory design pattern
  - I would have preferred factory to home as a naming convention
  - But we are stuck with home
- User specifies interface only, no code

Component interface

- Called remote interface in a lot of the literature
  - made more sense when all client references were remote
- User specifies interface only, no code
Responsibilities

• You provide
  – Component interface
  – Home interface
  – Bean implementation class
• Container generates
  – Home implementation and stubs
  – EJBOBJECT implementation and stubs

Bean doesn’t implement component interface

• Typically, the bean doesn’t implement the component interface
• A bean is not an instance of a something that can be invoked by a client
• How do you know that a bean can handle all the methods supported by the interface?
  – tools

If this worries/bothers you

Narrowing

• Normally, in Java/RMI, you can just downcast a reference returned from a lookup
  – e.g., RemotePortal p
    p = (RemotePortal) Naming.lookup(pAddress);
• Not in EJB
  Context ic = new InitialContext();
  Object o = ic.lookup("Advisor");
  AdviceHome home = (AdviceHome)
    PortableRemoteObject.narrow(o, AdviceHome.class);
Deal with it

• This is ugly, but necessary
• Deal with it
• RMI may be implemented with IIOP/CORBA
  – in which case, downcast may actually have to do something
• If nothing needed, it is cheap

EJBOBJECT interface

- EJBHome
  - getEJBHome()}
- Handle
  - getHandle()
- Object
  - getPrimaryKey()
- boolean
  - isIdentical(EJBOBJECT obj)
- void
  - remove()

EJBJHOME interface

- EJBJMetaData
  - getEJBJMetaData()
- HomeHandle
  - getHomeHandle()
- void
  - remove(Handle handle)
- void
  - remove(Object primaryKey)

Handles

- Handles are persistent references to EJBOBJECTs
- Can be serialized, passed between machines
  – stored in Servlet sessions
isIdentical

- Used to determine whether two EJBObjec
  stubs refer to the same bean
- Stateless session beans
  - true if they come from the same home
- Stateful session beans
  - false for any two distinct stubs
- Entity beans
  - true if entities have same primary key and same home

Session/Entity Bean Creation

- Home interface must have create(...) methods
- EJB must have matching ejbCreate(...) interfaces
- Stateless session beans should have only no-argument create methods

Removal

- Removing an entity bean deletes the corresponding info from persistent store
- Removing a session bean says that you are done with it

Stateful Session Bean Lifecycle

- bean removed or times out
- bean inactive
- create method in home invoked
- method ready
- exception thrown
- passivated
Lifecycle notifications

- nowhere
- garbage
- method
- ready
- passivated

exception thrown

business method

Stateless Session Beans

- No state that lasts longer than a single call from a client
  - Only one call at a time on a stateless session bean
  - Can have instance variables / state that persist for the duration of one call
  - Can keep resources, debugging info, caches
- Different invocations from a client can be routed to different beans

Local interfaces

- EJB2.0 adds local interfaces for components and homes
  - extend EJBLocalHome and EJBLocalObject
- Local objects don’t have handles
- Call by reference rather than deep copy
- Methods don’t throw remote exception

Mix and Match

- No need for one bean to support both Local and Remote interfaces
  - rather different
- Client of bean must know what it is getting