## Service Level Requirements (non-functional)

- Performance
- Scalability
- Reliability
- Availability
- Extensibility
- Maintainability
- Manageability
- Security

## Fault tolerance

Backups

- Hot = live copies, serve clients, sync continuously w/ master, master/slave
- Warm = sync at regular intervals, don't serve clients
- Cold = stable storage at regular intervals

Replication

- Active = all replicas handle requests
- Passive = same as warm backup

DNS round robin: simple load balancing, next machine on list, does not account for actual usage level of machine

CORBA

- Java will not allow CORBA over HTTP
- Objects communicate through ORBs
- IIOP used as protocol
- Interface and implementation repositories
- IDL translated to concrete language
- Static invocation using pre-compiled stubs & skeletons
- Move the state of an object but not behavior

RMI-IIOP

- No distributed GC
- No stub downloads
- Pass by value
- Goal is to marry RMI and CORBA
- EJB spec requires RMI-IIOP
- Must use PortableRemoteObject.narrow and JNDI
- Move state and behavior

Java IDL

- Used to access existing CORBA servers
- Connect RMI app to CORBA server

You should use RMI-IIOP with EJB

RMI-IIOP can be used with CORBA only if the remote interface was originally defined as an RMI interface

Protocols used to synchronously communicate with legacy systems:

- IIOP
- RMI+IIOP
- JRMP with JNI

EJB Structure

Remote Interface

- Extends javax.ejb.EJBObject
- Declare business methods clients can call
- Must throw RemoteException

In EJB 2.0 the remote interface is now the Component Interface

Home Interface

- Extends javax.ejb.EJBHome
- Create methods
    - Return remote interface
    - Throw remote CreateException, RemoteException
- Find methods for entity beans
    - Must implement at least findByPrimaryKey
    - ejbFind() returns primary key
    - Return single remote interface (EJBObject)
    - Return collection of remote interfaces (Enumeration)
- Remove methods

Bean Class

- Implements SessionBean or EntityBean
- Declare fields
- Declare business methods, must match declaration in remote interface
- EJB methods:
  - ejbCreate
  - ejbActivate / ejbPassivate
  - ejbRemove
- EJB methods for entity beans only:
  - ejbPostCreate
  - ejbLoad /ejbStore
  - ejbFind
- Provides implementations for
  - Methods in bean's home interface
  - Methods in bean's remote interface
  - EJB callback methods

In EJB 2.0 CMP the bean class is now an abstract class

Primary Key Class

- Implements serializable
- Fields must be public, no args constructor
- Must redefine equals and hashcode
- For CMP, must be a subset of container managed fields

When undefined primary keys are used, bean classes and interfaces refer to primary key using java.lang.Object

## Session beans

Stateless

- Only attached to client during execution
- Provides a service to the client
- When client calls create on remote interface the container takes an instance from Method Ready pool and attaches it to client

Stateful

- Bean handles a specific client
- All fields not transient must be serializable or primitive
- The following references are guaranteed by the container
    - Home/remote references of other beans
    - JNDI contexts and SessionContext
    - UserTransaction
- When passivated:
    - Close all open resources
    - Set non-transient, non-serializable fields to NULL

EJB methods

- setSessionContext, called before ejbCreate
- usually empty for stateless beans

Entity beans

CMP: performance not essential, standard data types
BMP: performance essential, complex data types

EJBCreate returns

- null BeanPK for CMP
- Primary key for BMP

Cannot take part in BMT

Primary keys remain undefined until bean is deployed

Implementation of create is optional

Remove called

- Remote reference is invalidated
- Data is deleted from the database

Concurrent access to shared data

The DML statements for inserting, deleting and updating data are coded in appropriate methods but since ejbLoad() and ejbStore() are callback methods, the container makes calls to them, as needed

## Bean Lifecycles

1. newInstance / constructor
2. setSessionContext
3. ejbCreate

Stateless

1. Not exist
2. Method ready (business methods stay in method ready)

Stateful

1. not exist
2. method ready / method ready in transaction
3. passivated

Entity

1. not exist
2. pooled
3. ready

EJB timeout property is vendor specific

## EJB Restrictions

- no use of static fields or methods
- no thread control
- no direct I/O
- no use of sockets

## Pooling

- The number of beans to pool is set at deployment

## Transactions

- Not supported : always exec outside current tx
- Supports: tx state left as is
- Required: when not in tx, one is created
- RequiresNew: a new tx is always started
- Mandatory: not being in tx = exception
- Never: being in tx = exception

Transactions are rolled back automatically with system exceptions, not app. exceptions

CMT: entity + session
BMT: session only (more granular)

EJB supports both implicit declarative tx and explicit JTA tx

## Transaction Management

Dirty read

- First transaction reads uncommitted changes made by second tx
- If second tx is rolled back, data is invalid

Repeatable read

- Data is guaranteed to look the same during a transaction

Phantom read:

- New records added to the database are detected by a tx thtat started before the insert

## Isolation levels

Read uncommitted

- Dirty, non-repeatable, phantom

Read committed

- Dirty prevented, non-repeatable and phantom can occur

Repeatable read

- Tx cannot change data being read by another tx
- Dirty and non-repeatable are prevented, phantom can occur

Serialized

- Tx has exclusive read and update privileges
- Dirty, non-repeatable and phantom are prevented

Messaging

Point to point (queue): multiple senders, single receiver

Publish-subscribe (topic): 1 sender, multiple receivers

Request-reply: synchronous messaging

Don’t use messaging when an instant response is required.

Synchronous messaging is good for transaction processing.

Applications exchange messages through virtual channels called destinations

Message driven beans

- Don’t have a home interface
- Clients don’t access MDB through interfaces

- 

Architectures

2-tier:

- Clients may be validation intensive
- Each client makes a direct connection with the server
- They are not very maintainable

N-tier J2EE:

- Client tier
- Web tier
- EIS tier
- EIS Integration tier

Don’t use DAO for StateFUL beans

JSP for presentation
Servlets for controller
Stateless session beans for catalog retrieval
Stateful session for shopping cart
Entity bean for order update
Entity bean for customer

Patterns – GoF

Bridge: creates a separation between abstractions and classes that implement those abstractions (example: JDBC)

Adapter: implements an interface known to its clients and provides an instance of a class not known to its clients

Abstract Factory: used to create many objects that are dependant on each other

Builder: separates construction and representation of an object

Factory Method: provides an interface for creating an object, defers creation to subclasses or helper classes

Prototype: create new objects by copying its prototype

Decorator: adds extra functionality to existing objects

Mediator: coordinate state changes between other objects by using one object (example: MOM)

Strategy: encapsulate an algorithm so it can be used interchangeably

Template method: lets subclasses redefine certain steps of an algorithm without changing the algorithm's structure

Memento: without violating encapsulation, capture and externalize an object's state so the object can be restored to this state later (example: undo, stateful beans)

State: allow an object to alter its behavior when its internal state changes, the object will appear to change its class

Composite: compose objects into tree structures to represent part-whole hierarchies, lets clients treat individual objects and compositions of objects uniformly

Proxy: provide a surrogate or placeholder for another object to control access to it

# Patterns – J2EE

Front controller

- provides a central point of entry (not for load balancing)
- manages:
  - Client request
  - Security
  - Delegation of business processing
  - Error handling
  - View selection
  - Content creation strategies

Service locator

- Improves performance with a cache
- Hides complexities of initial object creation, EJB lookups and object re-creation
- Multiple clients can reuse the same SLP

Fast lane reader

- Reduce the amount of code needed
- Used for large amounts of read-only data

Value object

- Snapshot of the state of an entity bean at a particular time

Internationalization

String and char “are used”
OutputStreamWriter : Unicode → 8 bit
InputStreamReader: 8 bit → Unicode
Java.util contains Locale and Date/time classes

UML

Interaction diagrams:

- Represent a dynamic view of the system
- Are isomorphic
- May show the structural organization of the objects that send or receive messages

Misc

Applets can connect back to the server from which they were downloaded

Applets can read a local file depending on browser security policy

Properties files should only store strings

You can add more files to a signed JAR archive since signing actually signs the individual files and not the JAR itself

HTTP is a connection based, stateless protocol

Package dependencies are not transitive

Use JCA to connect to EIS