



# JAVAPOLIS

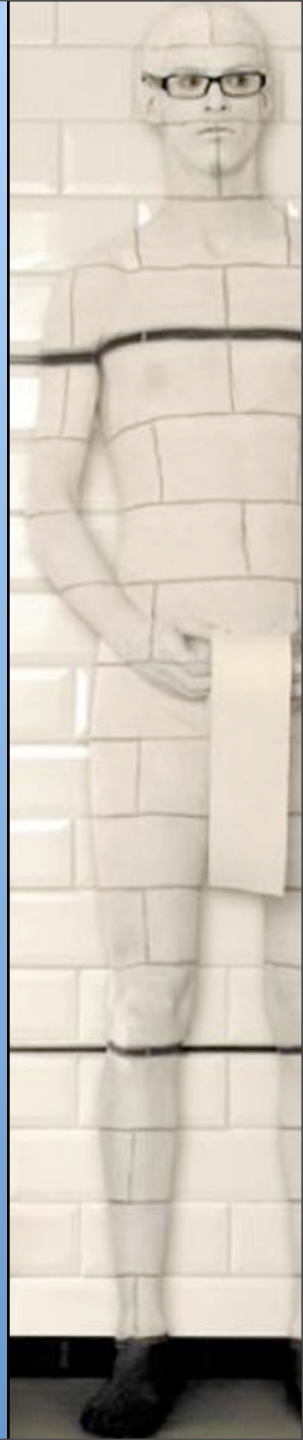
11 - 15 DECEMBER ■ ANTWERP ■ BELGIUM





# Java EE Enhancements for Real World Deployments

Nagesh Susarla  
Staff Software Engineer  
BEA Systems



# Overall Presentation Goal

---

Get an understanding of the latest application packaging, upgrade and configuration features in a Java EE environment

*It covers in detail the **application library** features allowing Java EE **EARs** to become composite applications of reusable component libraries.*



# Speaker's Qualifications

---

- 👤 Nagesh Susarla is a Staff Software Engineer at BEA Systems
- 👤 Nagesh Susarla leads the *Java EE Application Container & WebApp* team
- 👤 Nagesh Susarla has in the past worked on building the WebLogic Servlet and JSP containers
- 👤 Nagesh Susarla writes frequently on WebLogic Server internals on [dev2dev.bea.com](http://dev2dev.bea.com)



# Beyond the APIs ...

---

Going beyond the Java EE programming APIs with features for packaging, upgrading, and maintaining enterprise applications.

5



# Outline

---

- ☕ Why Share?
  - ⇒ Application Libraries – What?
- ☕ Production Redeployment
  - ⇒ Interruption-free?
- ☕ Configuration Changes
  - ⇒ Can it be external please?



# Outline

---

## Why Share?

- ⇒ Java EE Deployment Units
- ⇒ Common Libraries & Packaging
- ⇒ Java EE Optional Packages
- ⇒ Application Libraries
- ⇒ Web Application Libraries

 Production Redeployment

 Configuration Changes



# Java EE Deployment Units

---

- ☕ EAR (Enterprise Archives)
  - ⇒ WAR (Web Application Archives)
  - ⇒ EJB (Enterprise Java Beans)
  - ⇒ RAR (Connector Archives)
  - ⇒ Application Clients



# Common Libraries

---

- ☕ Common libraries are used by many applications
  - ➔ Developed in-house
  - ➔ Frameworks (Struts, Webwork, Spring, Portal)
- ☕ Jar containing classes.
- ☕ EAR containing
  - ➔ EJBs
  - ➔ WebApp modules containing classes and resources.

9



# Packaging Common Libraries

---

- ☕ Package common modules:
  - ➔ In the System Classpath.
  - ➔ Within each application.
- ☕ New versions require manually updating Application.
- ☕ Updating difficult when modules contain resources and classes.



# Problems

---

## Scoping

- ➔ All Modules are tightly scoped.
- ➔ Modules cannot reference modules residing outside an EAR

 Delivering applications as a single EAR is limiting.



# Java EE Optional Packages

---

- ☕ Share Plain Jars (w/ Classes)
- ☕ Any Module type can refer to an optional Jar.
  - ➔ Via META-INF/MANIFEST.MF

```
Extension-List: optpack1 [optpackn]  
optpack1-Extension-Name: library-name  
optpack1-Specification-Version: (optional)  
optpack1-Implementation-Version: (optional)
```



# Java EE Optional Packages

---

- ☕ Added to Module Classpath
- ☕ Match package with highest specification/implementation-version
- ☕ Descriptors are ignored



# Application Libraries

---

- ☕ Goals
- ☕ Supported Types
- ☕ Semantics
- ☕ Deployment
- ☕ Referencing
- ☕ Portability



# Application Libraries - Goals

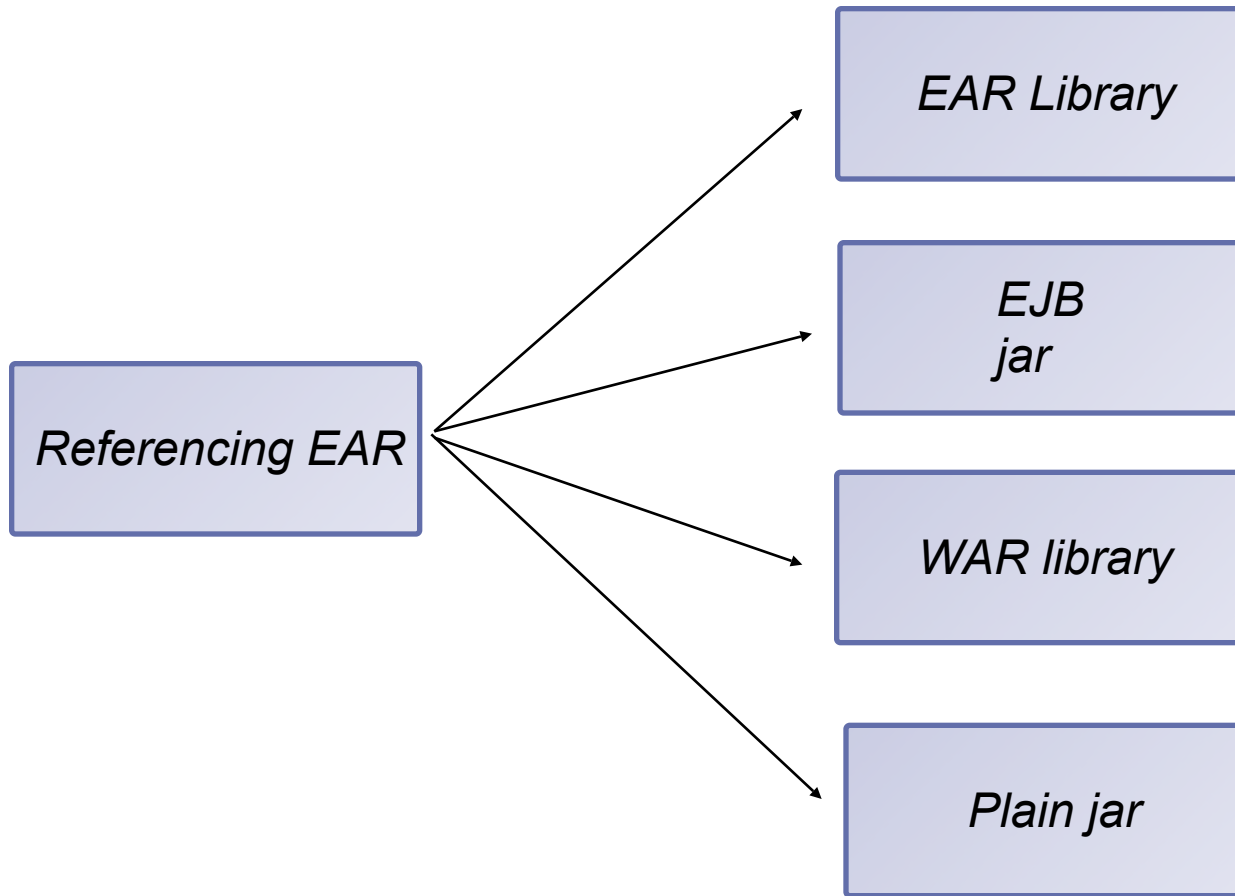
---

- ☕ Sharing of external code libraries
- ☕ Portability of library based applications
  - ➔ Older WebLogic Server Releases
  - ➔ Other Application Servers
- ☕ Require limited changes to existing applications to be packaged as a library.
- ☕ Provide Versioning
- ☕ Distribution and management.

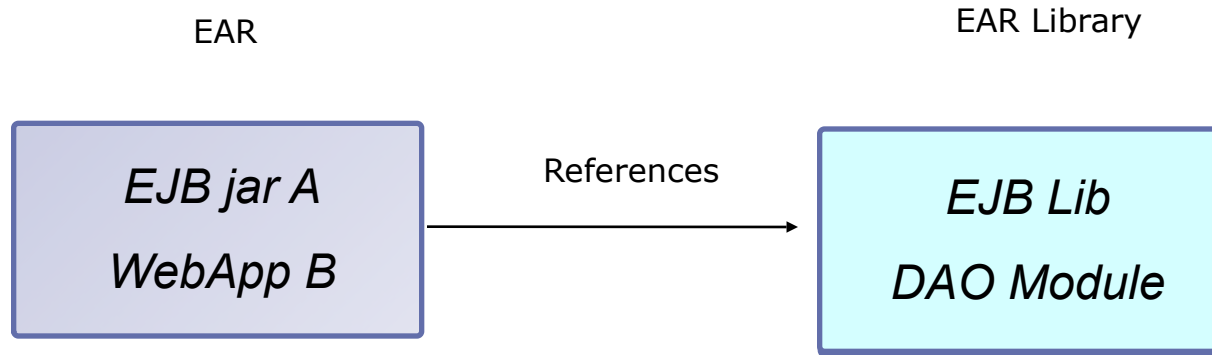
15



# Application Libraries - Types



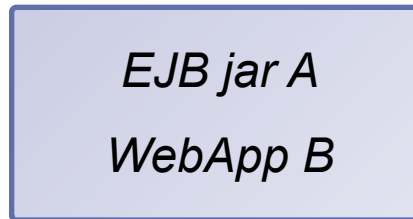
# Application Libraries - Semantics



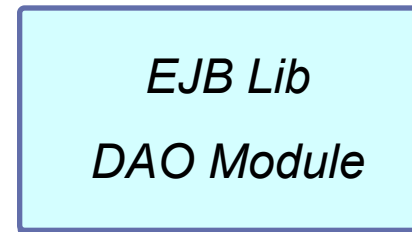
# Application Libraries - Semantics

---

EAR



EAR Library



17

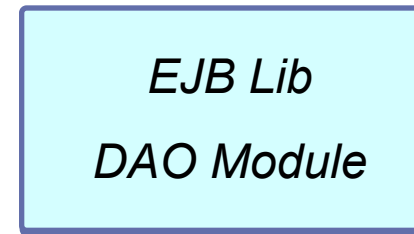


# Application Libraries - Semantics

---

EAR

EAR Library



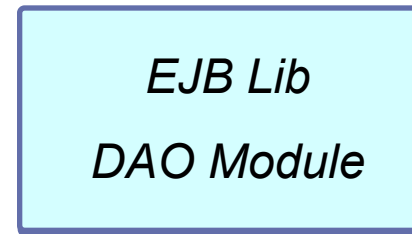
17



# Application Libraries - Semantics

---

EAR Library



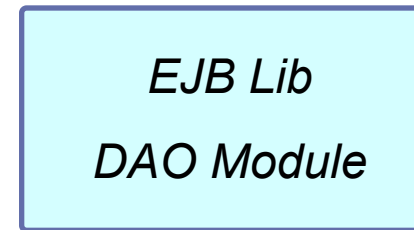
17



# Application Libraries - Semantics

---

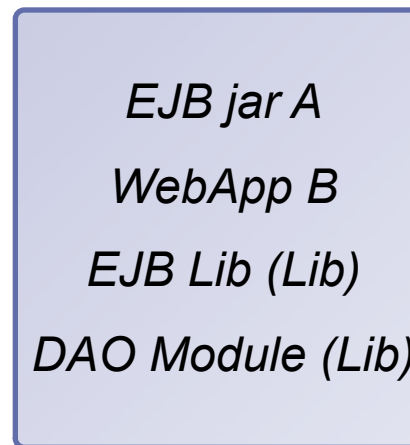
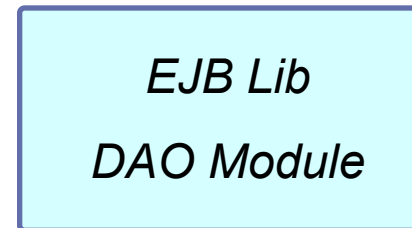
EAR Library



# Application Libraries - Semantics

---

EAR Library



Merged View

17



# Application Libraries - Semantics

---

- ☕ Application(2) + Lib(2) => App'(4)
- ☕ Applications can reference:
  - ➔ EAR Libraries: application.xml & weblogic-application.xml merged
  - ➔ EJB jars: Included to the list to be deployed.
  - ➔ WARs
  - ➔ Plain jars: Added to the App CP
- ☕ Precedence order: App > Lib1 > Lib2 .

18



# Application Libraries - Semantics

---

- ☕ Resource and ClassLoading
  - ➡ Libraries merged with referencing application.
  - ➡ No shared classloader for libraries
    - Complicates redeployment of libraries and referencing applications.



# Application Library - Deployment

---

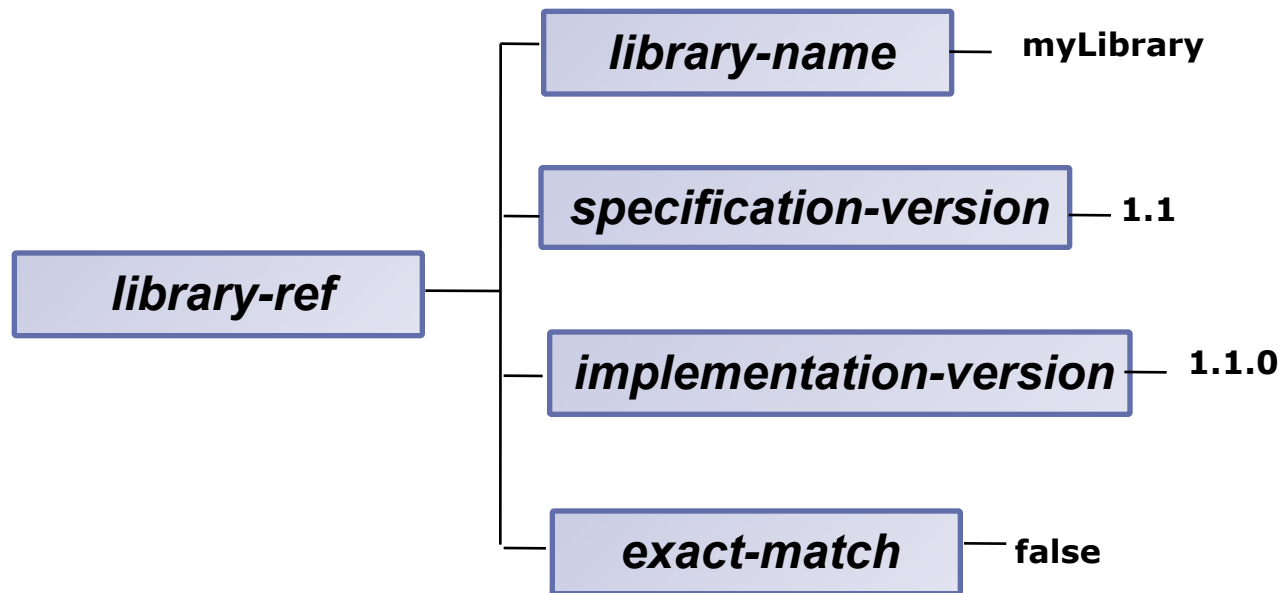
- ☕ Package an EAR, EJB jar or WAR
- ☕ Add *META-INF/MANIFEST.MF* :
  - Extension-Name: libName
  - Specification-Version: 1.1
  - Implementation-Version: 0.9
- ☕ Multiple specification-versions may be deployed
- ☕ Deploy - Multiple libraries same specification-version but different implementation-version

20



# Application Libraries - Referencing

- ☕ Add library-ref element to referencing EAR's weblogic-application.xml.



# Application Libraries - Referencing

DAO Lib A

***Spec-version: 1.1***

***Impl-Version: 1.3***

***Spec-version=1.1***  
***exact-match=true***

StableTeam  
App

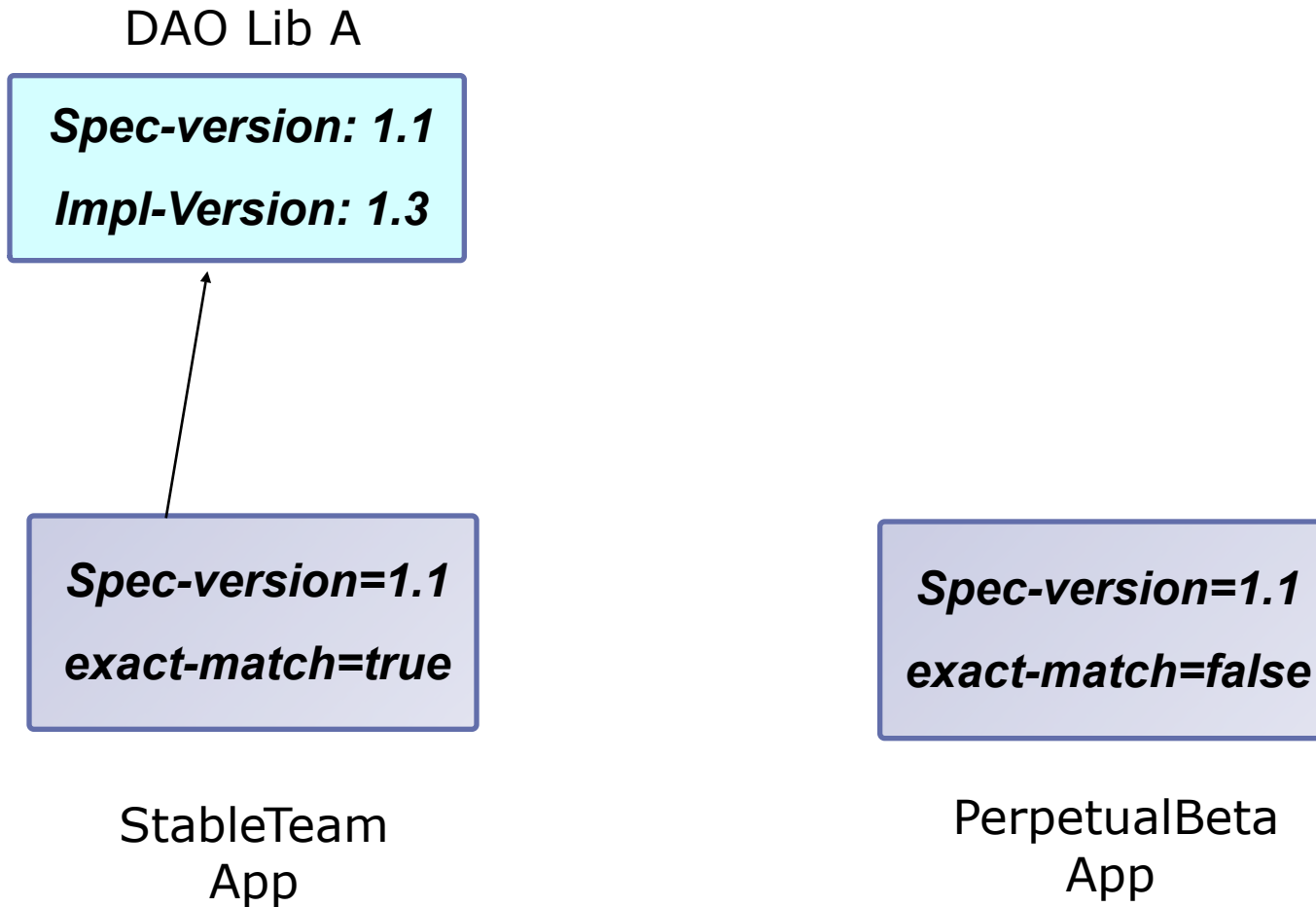
***Spec-version=1.1***  
***exact-match=false***

PerpetualBeta  
App

22



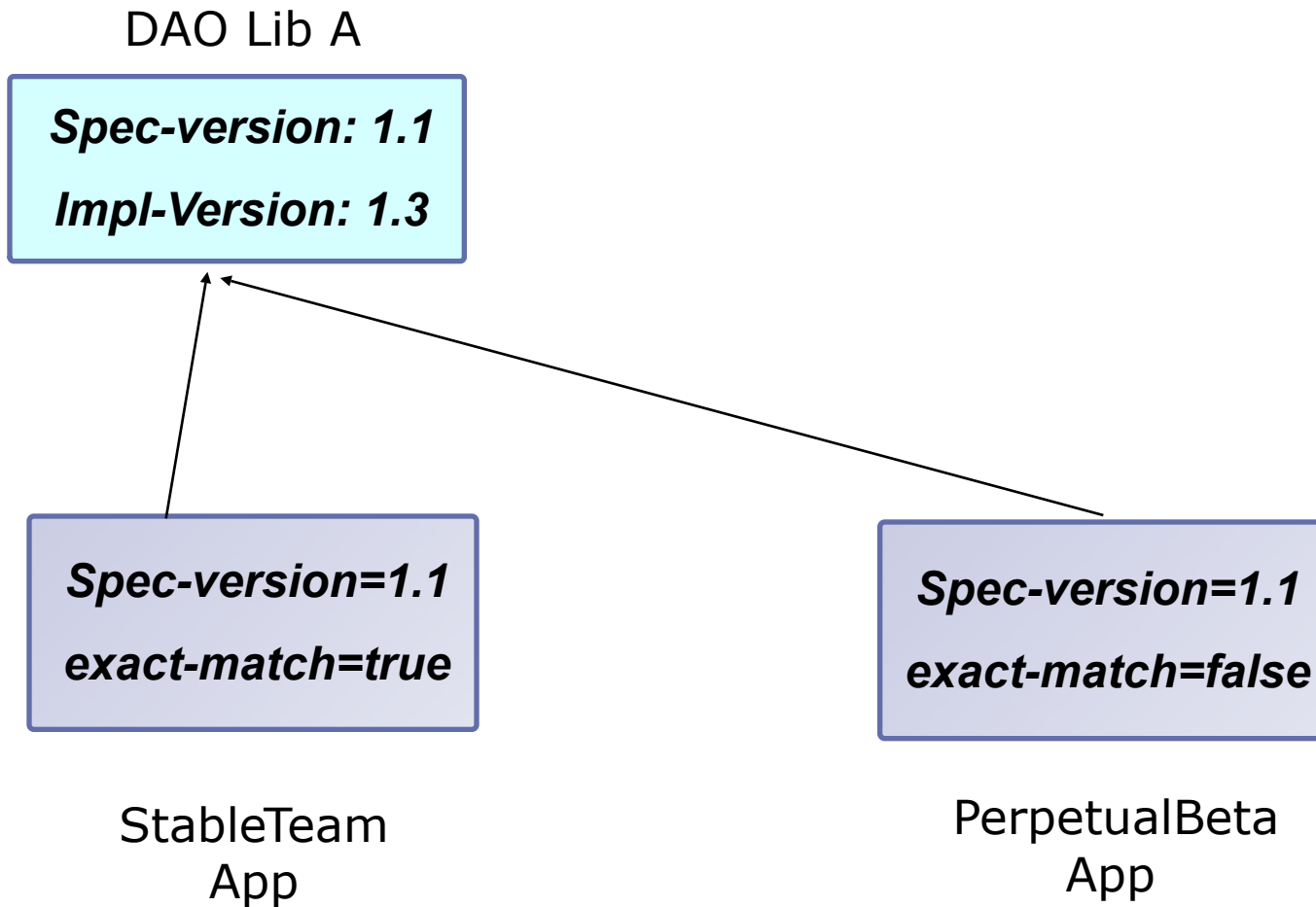
# Application Libraries - Referencing



22



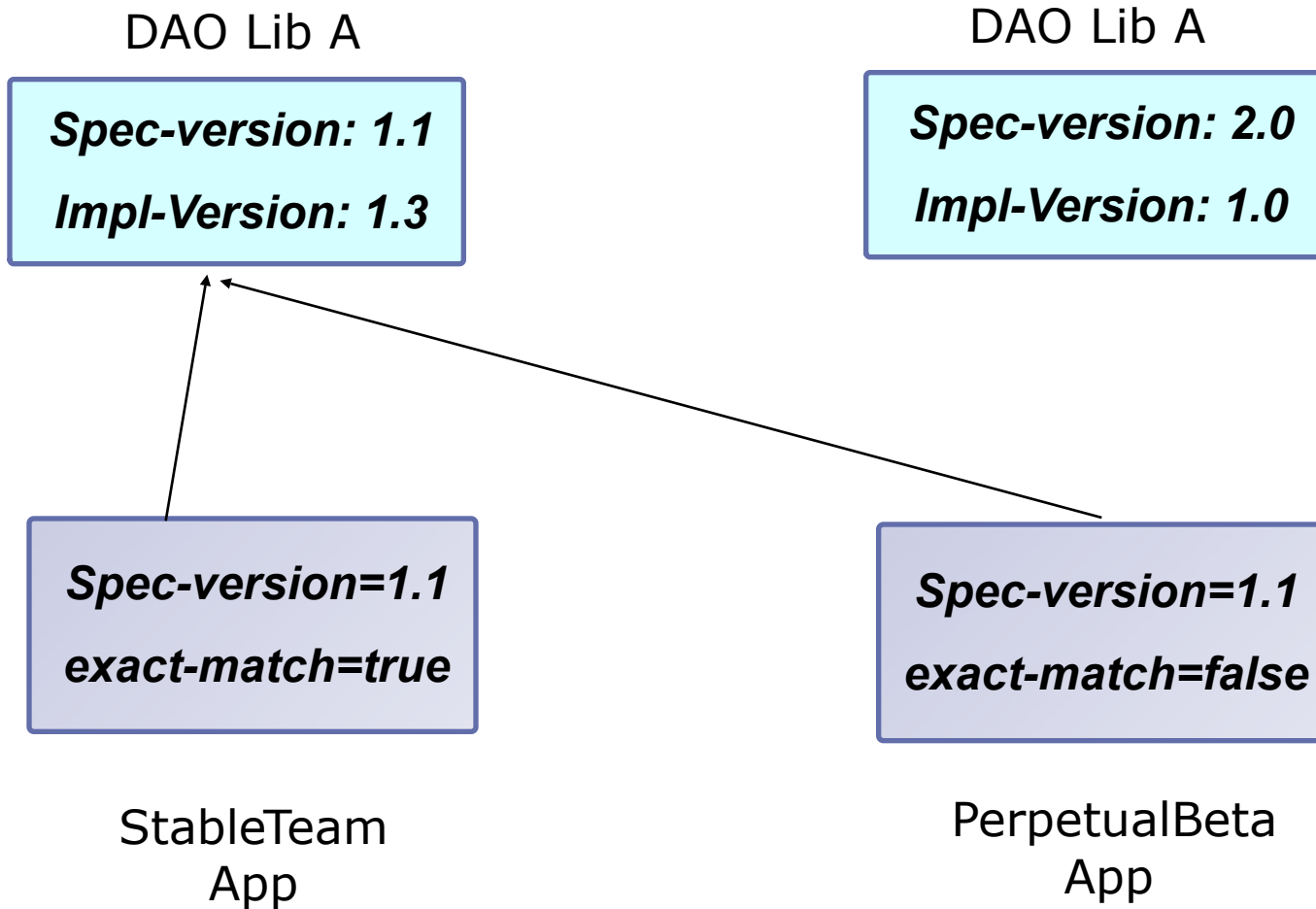
# Application Libraries - Referencing



22



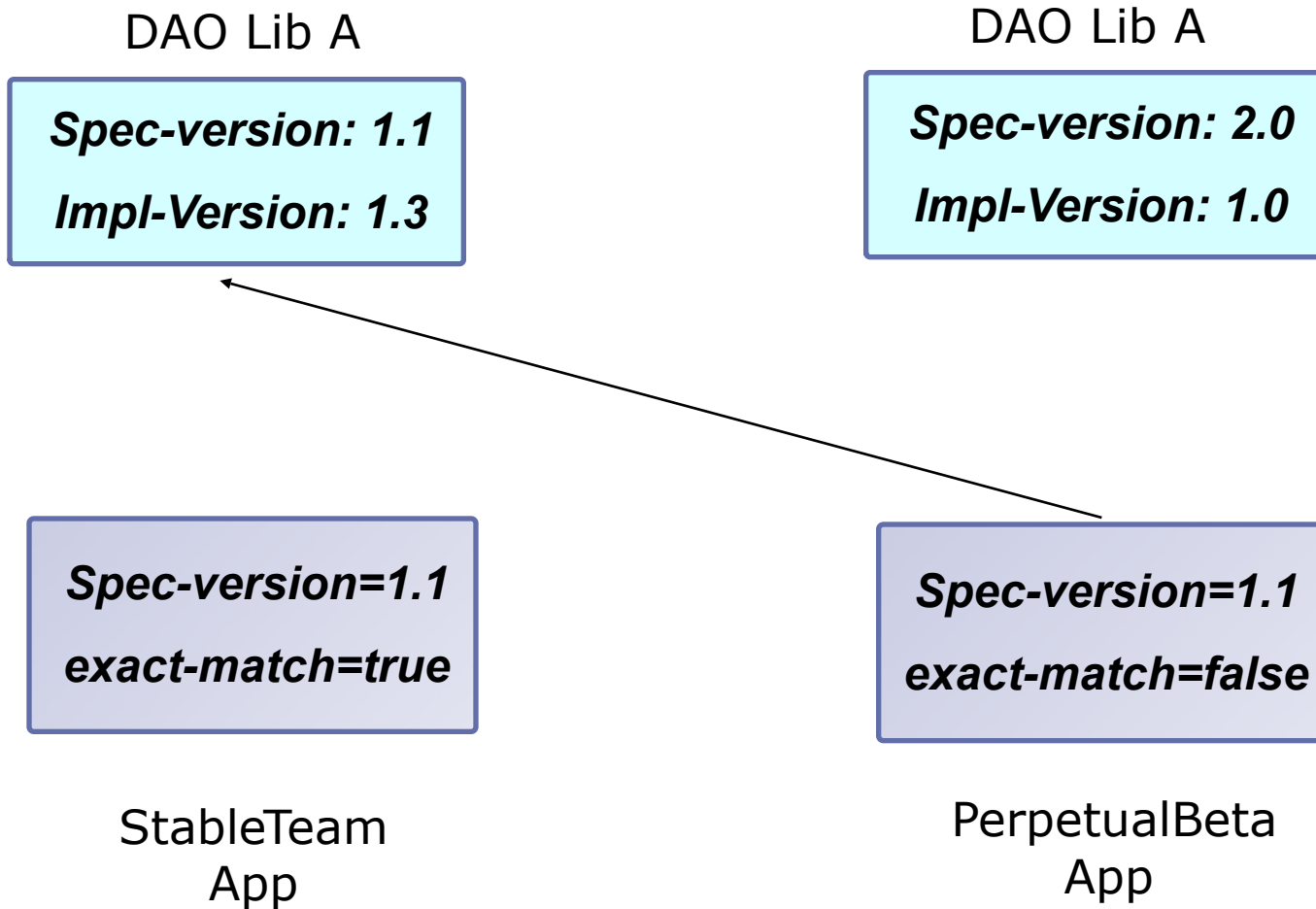
# Application Libraries - Referencing



22



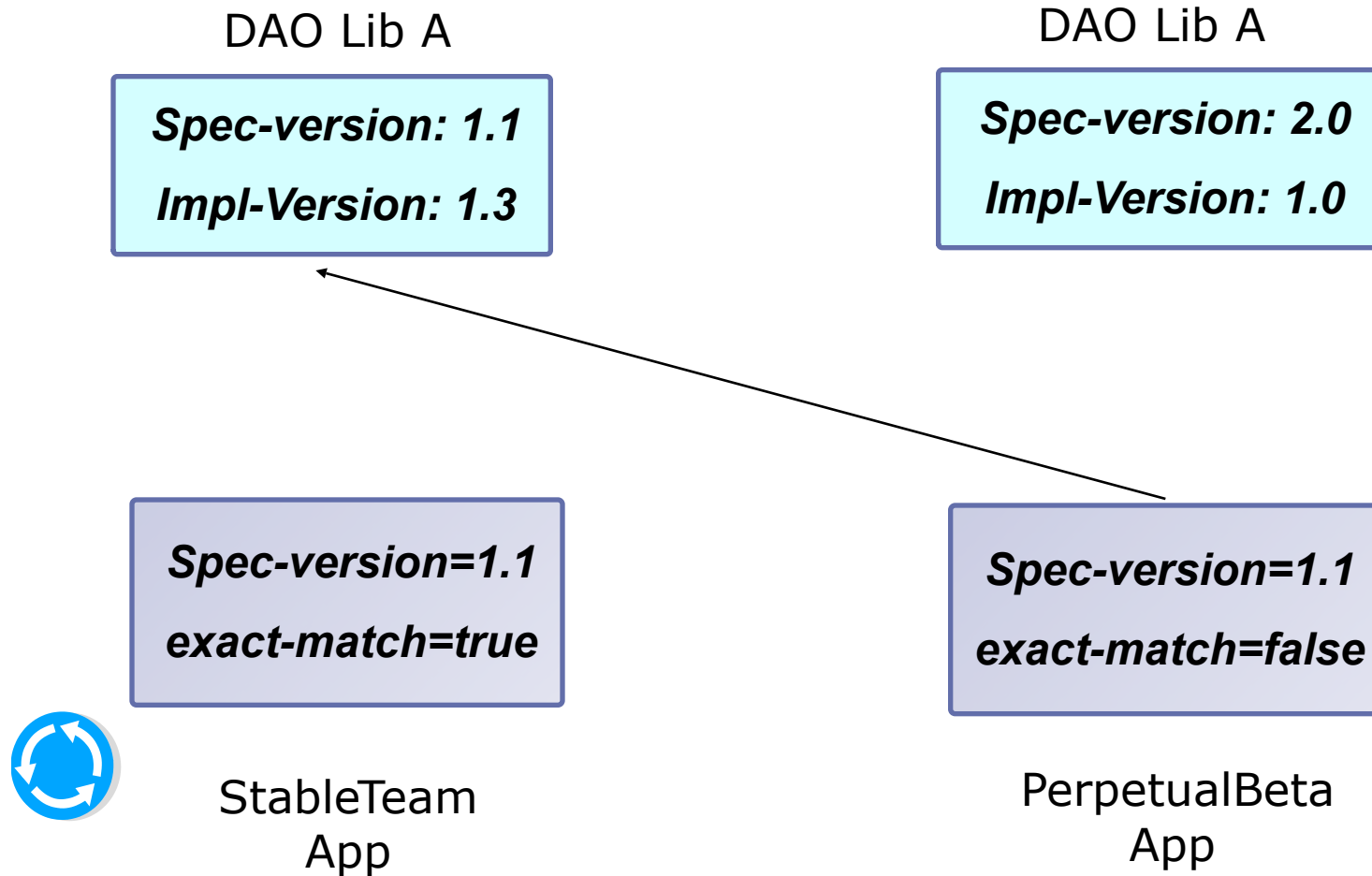
# Application Libraries - Referencing



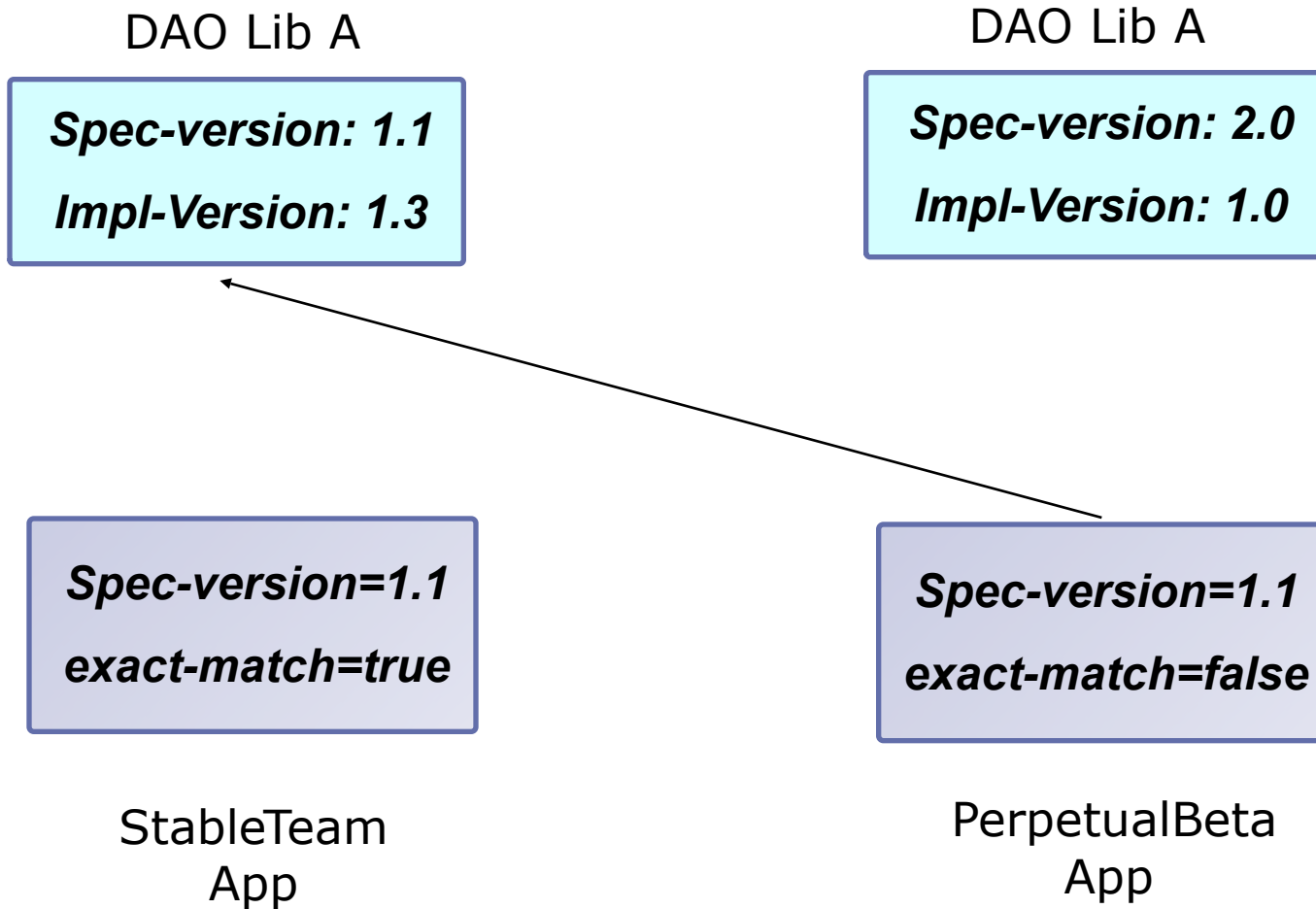
22



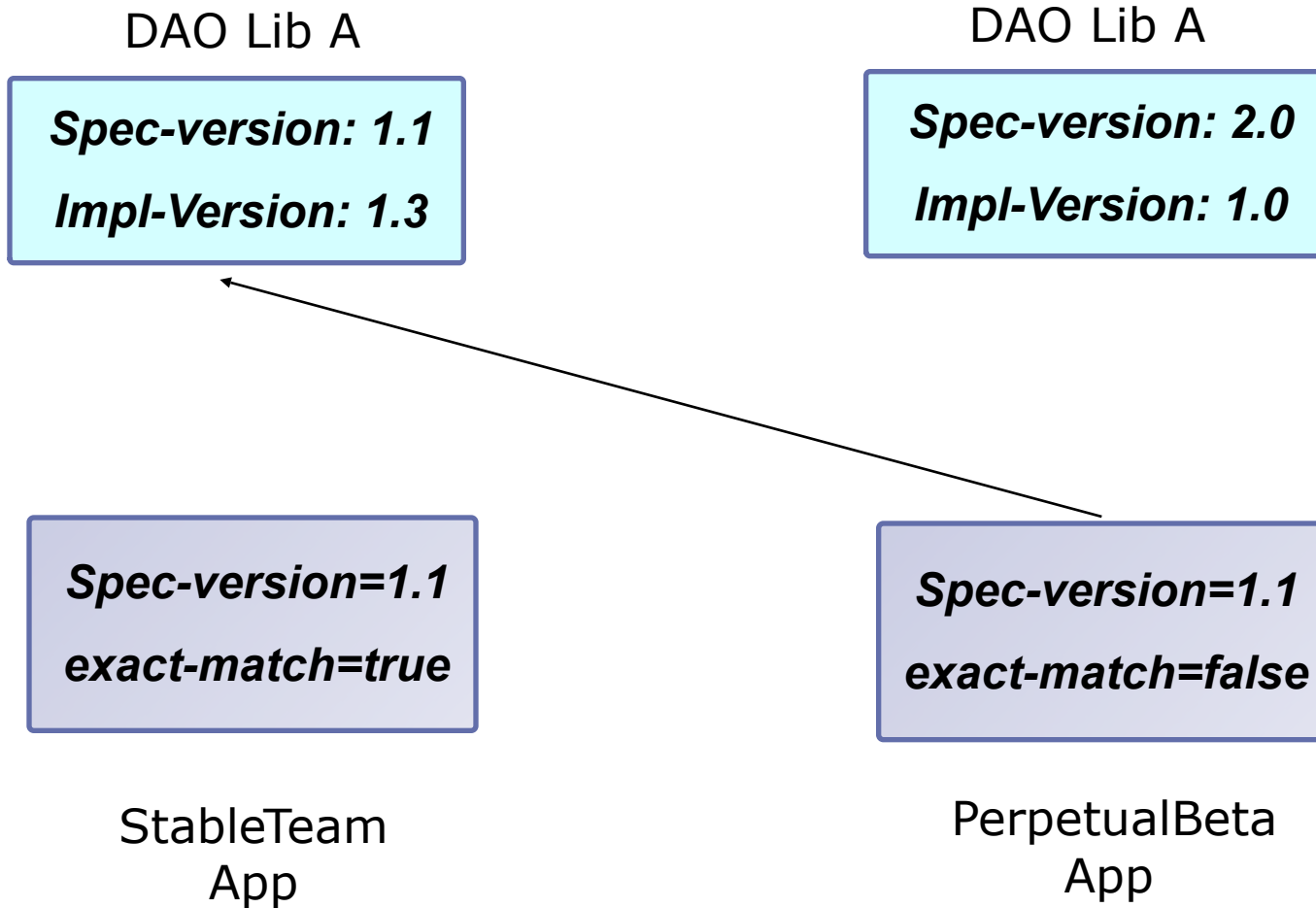
# Application Libraries - Referencing



# Application Libraries - Referencing



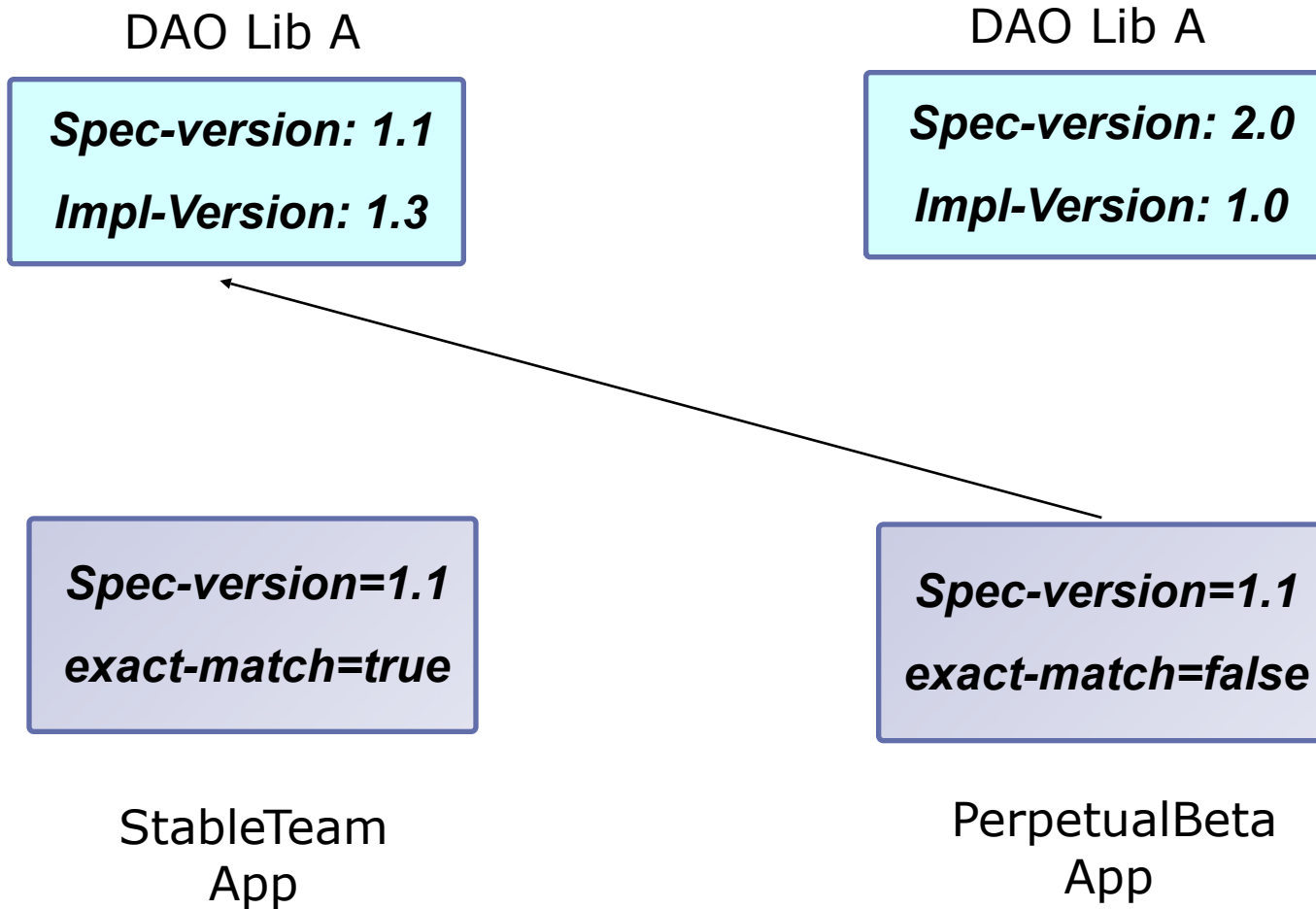
# Application Libraries - Referencing



22



# Application Libraries - Referencing



22



# Application Libraries - Referencing

DAO Lib A

***Spec-version: 1.1***  
***Impl-Version: 1.3***

DAO Lib A

***Spec-version: 2.0***  
***Impl-Version: 1.0***

***Spec-version=1.1***  
***exact-match=true***

***Spec-version=1.1***  
***exact-match=false***

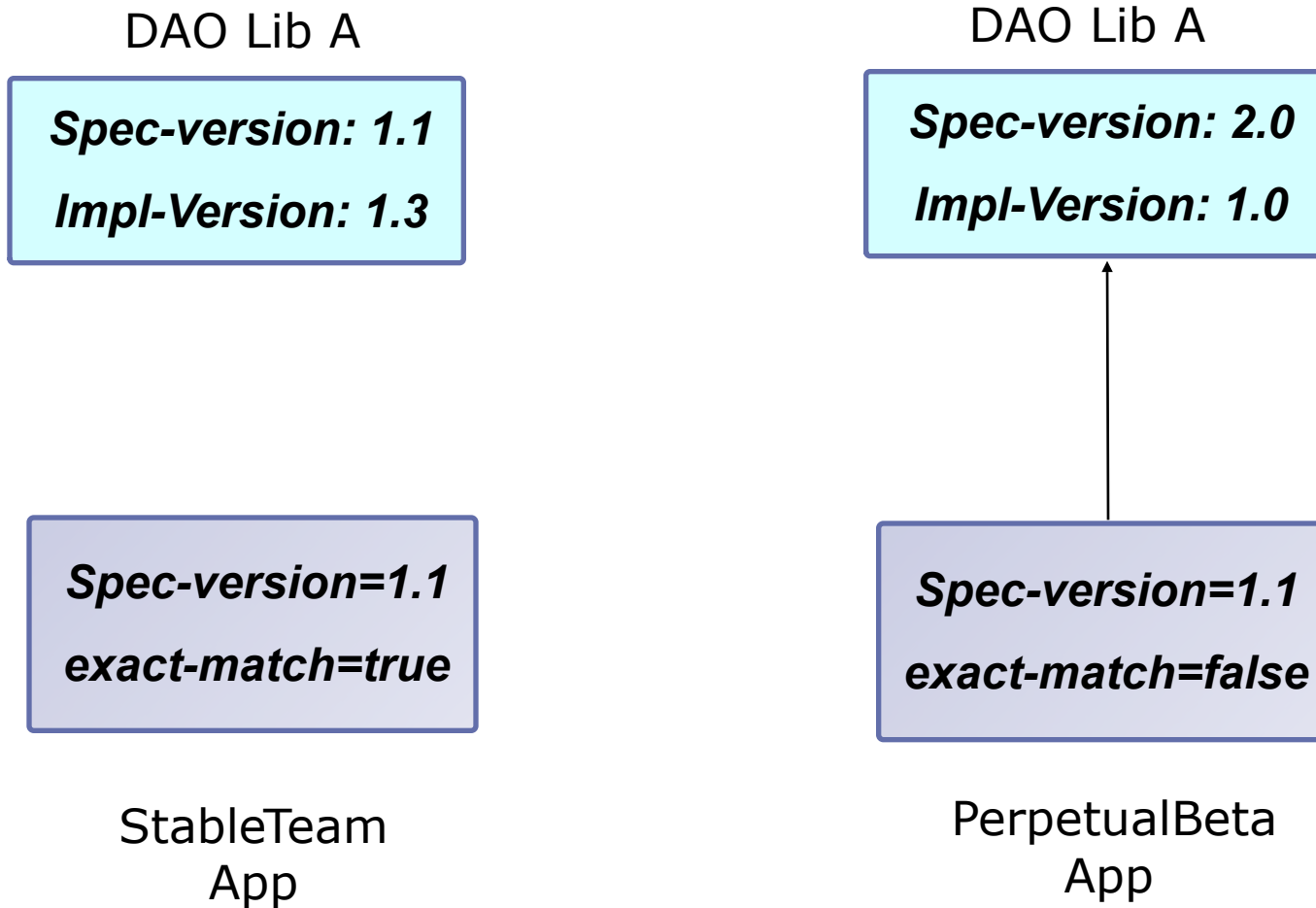
StableTeam  
App

PerpetualBeta  
App

22



# Application Libraries - Referencing



# Application Libraries – Portability

---

- 🤖 Goal: Deploy library based apps to older releases of WebLogic Server or to other application servers
- 🤖 Command line tool called “*weblogic.appmerge*”
- 🤖 Uses the same code as the runtime Container.
- 🤖 Physically merges the libraries with the application.
- 🤖 Merges Descriptors and writes them to disk.



# Web Application Libraries

---

- ☕ Similar to Application libraries
- ☕ WAR can refer to WAR libraries
- ☕ Useful in referring to web frameworks
- ☕ Nested libraries are supported
- ☕ web.xml is merged.
  - ➔ Filters, servlets from library visible to referencing web app



# Web Application Libraries (cont'd)

---

- ☕ Example: JSTL and JSF are shipped with WebLogic Server as WAR libraries.
- ☕ Semantics and Deployment are similar to Application Libraries.
- ☕ Resources and classes are returned giving precedence to the referencing WAR.



# Outline

---

- ☕ Why Share?
  - ⇒ Application Libraries – What?
- ☕ **Production Redeployment**
  - ⇒ **Interruption-free?**
- ☕ Configuration Changes
  - ⇒ Can it be external please?



# Production Redeployment

---

- ☕ Motivation
- ☕ Goals
- ☕ Multiple Versions
- ☕ Versioned Artifacts
- ☕ Automatic Retirement
- ☕ Testing New Application Version
- ☕ Rollback
- ☕ Managing Versions



# Production Redeployment

---

## Challenge

- ⇒ New development practices demand constant updates to your application
- ⇒ Ensure application availability during application upgrade

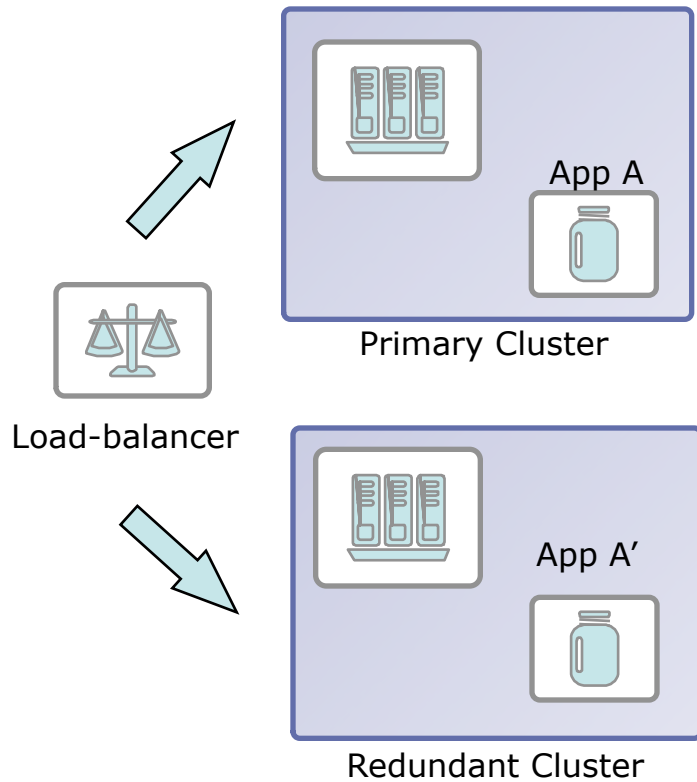
## Current Approach

- ⇒ Scheduled Downtime
- ⇒ Redundant cluster/domain with hardware load-balancer

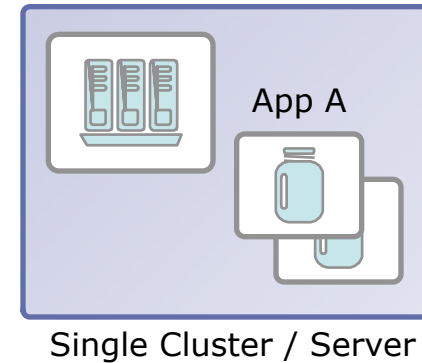


# Side by Side - Motivation

## Current Approach: Hardware-based LB



## WebLogic: Software-based Versioning



# Side by Side Deployment - Goals

---

- ☕ Allow multiple versions to coexist
- ☕ Retirement policy – Graceful, timeout
- ☕ Ability to Test new version before opening it to the “world”
- ☕ Rollback to the previous version
- ☕ Managing Versions



# Multiple versions

---

- ☕ Version isolation of the same app (EAR or WAR)
  - ➔ New Requests routed to active version
  - ➔ Existing client requests can finish up with retiring version
- ☕ Unshared local/app-scoped resources
- ☕ Shared Global resources
- ☕ Applications recommended to be self-contained



# Versioned Artifacts

---

## JNDI

- ⇒ Each version has its own app-scoped JNDI tree
- ⇒ Global bindings are versioned
  - Current version maintained in Work Context
  - Transparent lookup and bind/unbind/rebind

## MBeans

- ⇒ Application and Runtime MBeans
- ⇒ AppDeploymentMBean



# Automatic Retirement

---

- ☕ Old version retired after new version is deployed
- ☕ Policies:
  - ➡ Graceful: wait for all in-flight work to be done
    - In-Flight work: HTTP Sessions and Transactions
    - Can be preempted by undeploy
  - ➡ Timeout: Specify timeout period on redeploy



# Testing New App Version

---

- ☕ Deploy new version in Admin-mode
  - ➔ Version Info in META-INF/MANIFEST.MF
- ☕ Admin Mode:
  - ➔ Accessible via Administration channel only
  - ➔ WebApp Requests and JNDI lookups
- ☕ Who can use Admin Mode?
  - ➔ Members of the “AdminChannelUsers” group
- ☕ Promote Admin-mode version to Active

34



# Rollback

---

- ☕ Rollback to previous version
  - ➔ Redeploy previous version
  - ➔ Pending retirement of previous version  
Cancelled



# Managing Versions

---

- ☕ Monitoring of all versions via WebLogic Console
  - ➔ Status: admin-mode, active, retiring, scheduled to be retired
  - ➔ In-flight work: HTTP Sessions
- ☕ Redeploy new versions
- ☕ Undeploy retiring versions



# Outline

---

- ☕ Why Share?
  - ➔ Application Libraries – What?
- ☕ Production Redeployment
  - ➔ Interruption-free?
- ☕ **Configuration Changes**



# Configuration Changes

---

- ☕ Motivation
- ☕ What are Deployment Plans
- ☕ Example
- ☕ Tool Support



# Deployment Plans - Motivation

---

39



# Deployment Plans - Motivation

---

## Java EE 5

- ⇒ Deployment Descriptors optional.
- ⇒ Annotations in source code.
- ⇒ Configuration spread over descriptors/ annotations
- ⇒ Ease of Development.
- ⇒ Administrator?



# Deployment Plans - Motivation

---

## Java EE 5

- ⇒ Deployment Descriptors optional.
- ⇒ Annotations in source code.
- ⇒ Configuration spread over descriptors/ annotations
- ⇒ Ease of Development.
- ⇒ Administrator?

 Need a way to externally configure application environment.



# Deployment Plans - What

---

- ☕ Optional xml file that resides **outside** the application.
- ☕ Set/Override environment property values.
  - ➔ External resources (jndi-name)
  - ➔ Tuning parameters (session-timeout, max-cache)
- ☕ Cannot change non-configurable properties (ejb-name).
- ☕ Uses XPath to declare/override values.

40



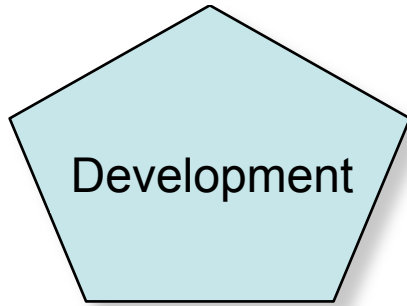
# Deployment Plans

---



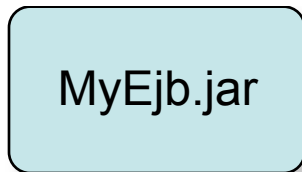
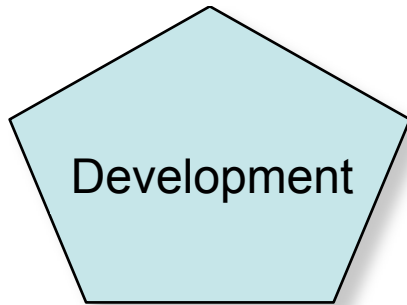
# Deployment Plans

---



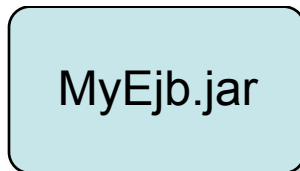
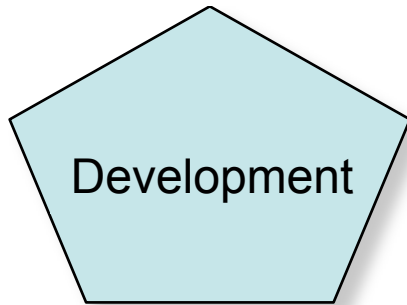
# Deployment Plans

---



# Deployment Plans

---

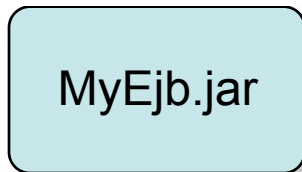
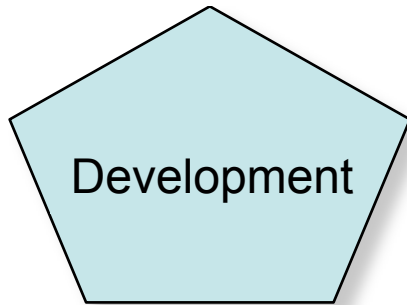


```
public class MyBean ... {  
    @Resource DataSource mySqlDataSource;  
    ....  
}
```



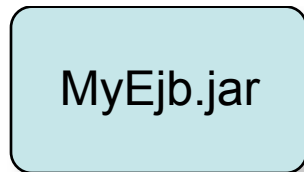
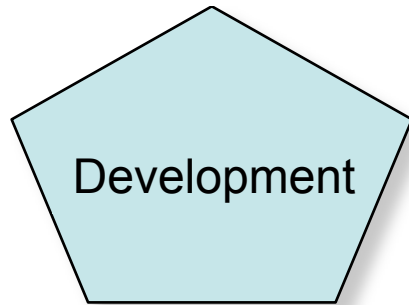
# Deployment Plans

---



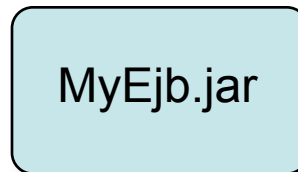
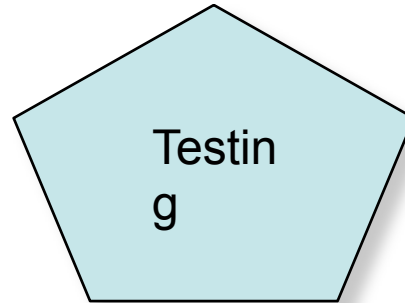
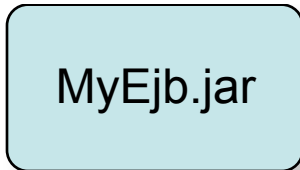
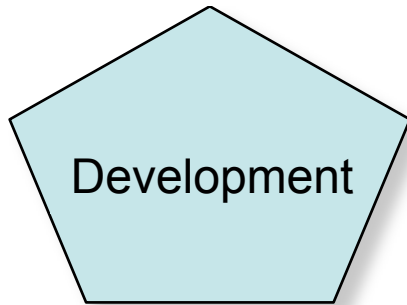
# Deployment Plans

---

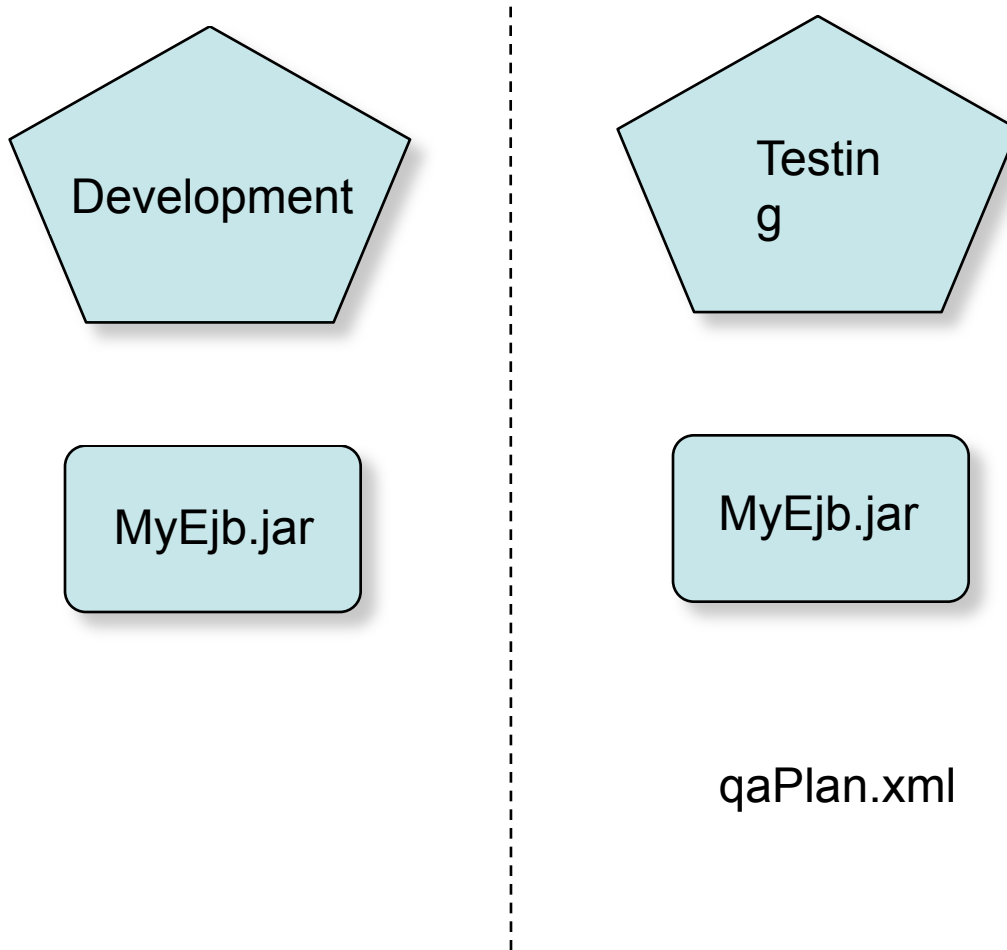


# Deployment Plans

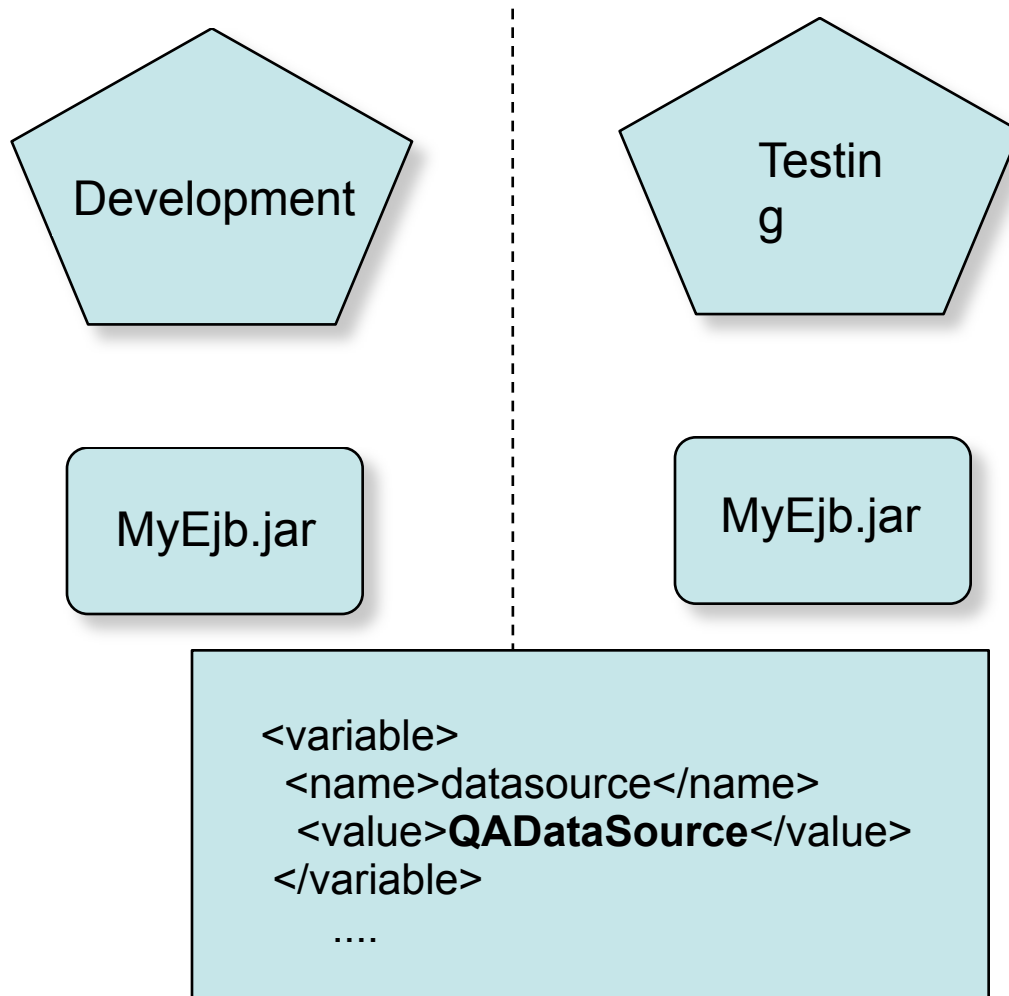
---



# Deployment Plans



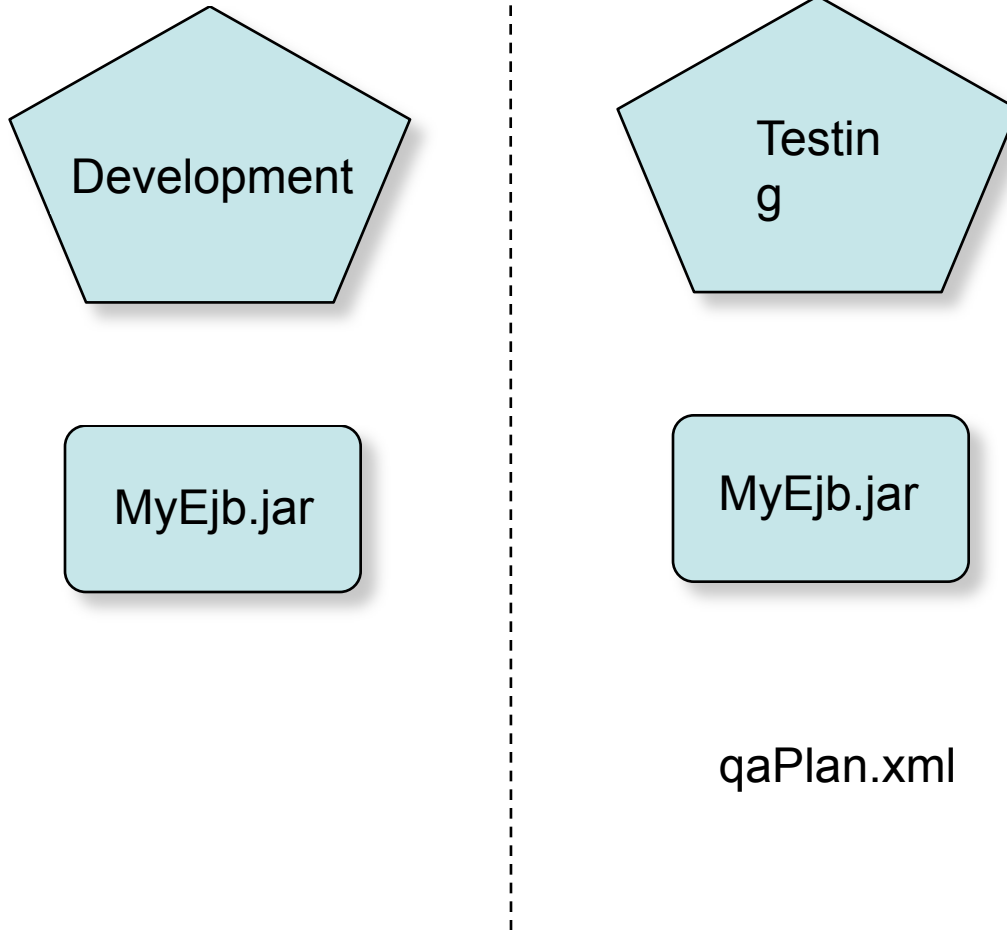
# Deployment Plans



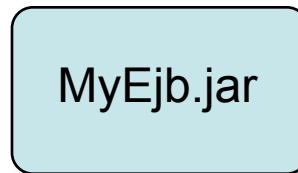
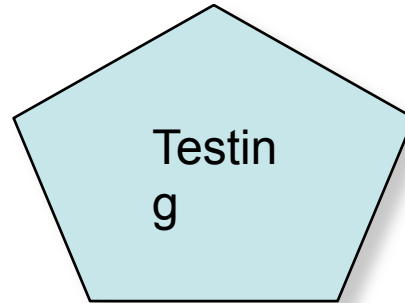
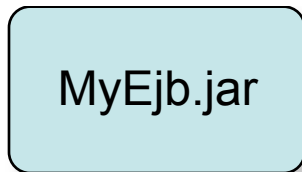
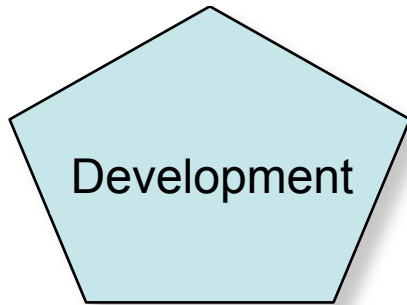
41



# Deployment Plans



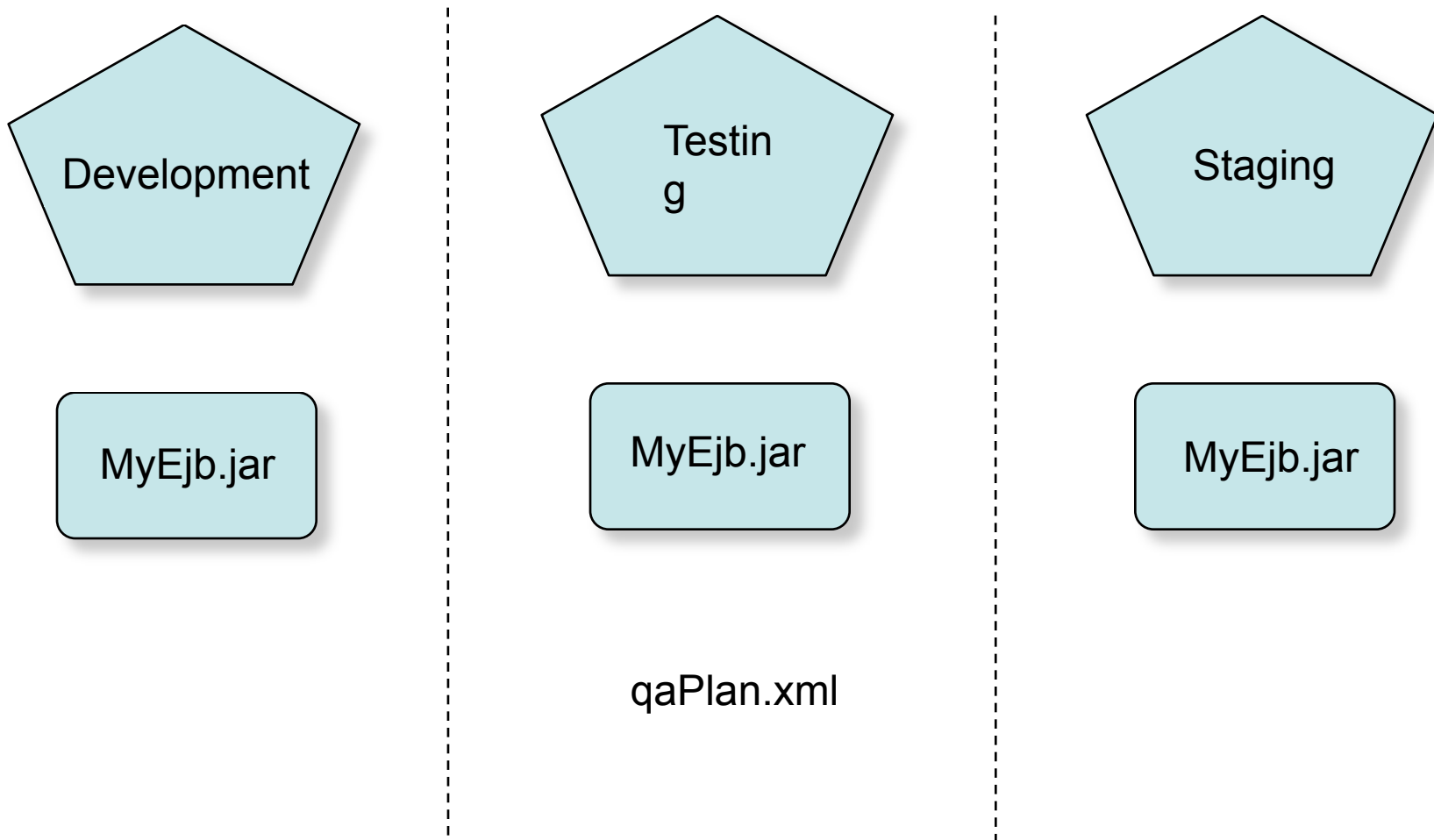
# Deployment Plans



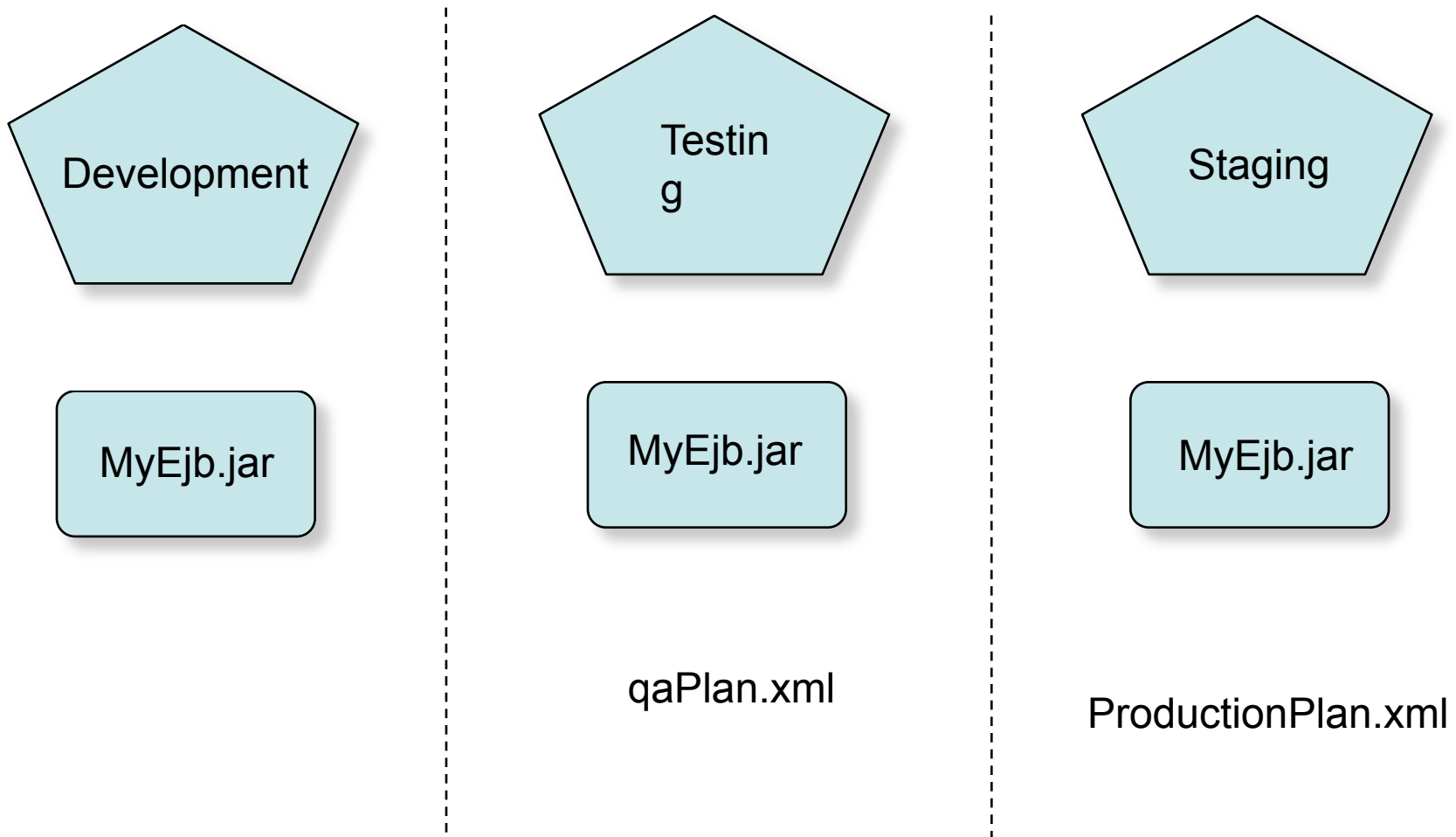
qaPlan.xml



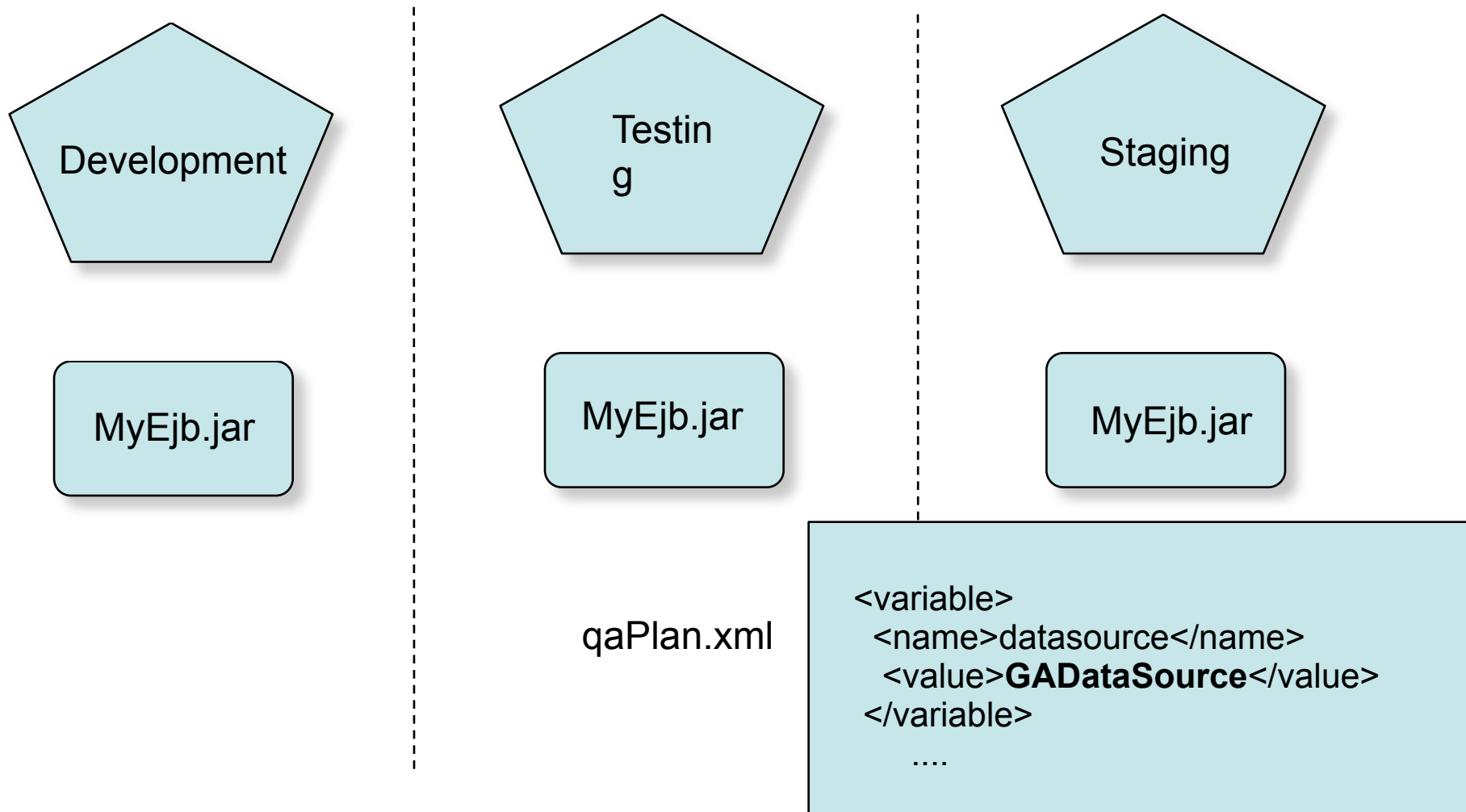
# Deployment Plans



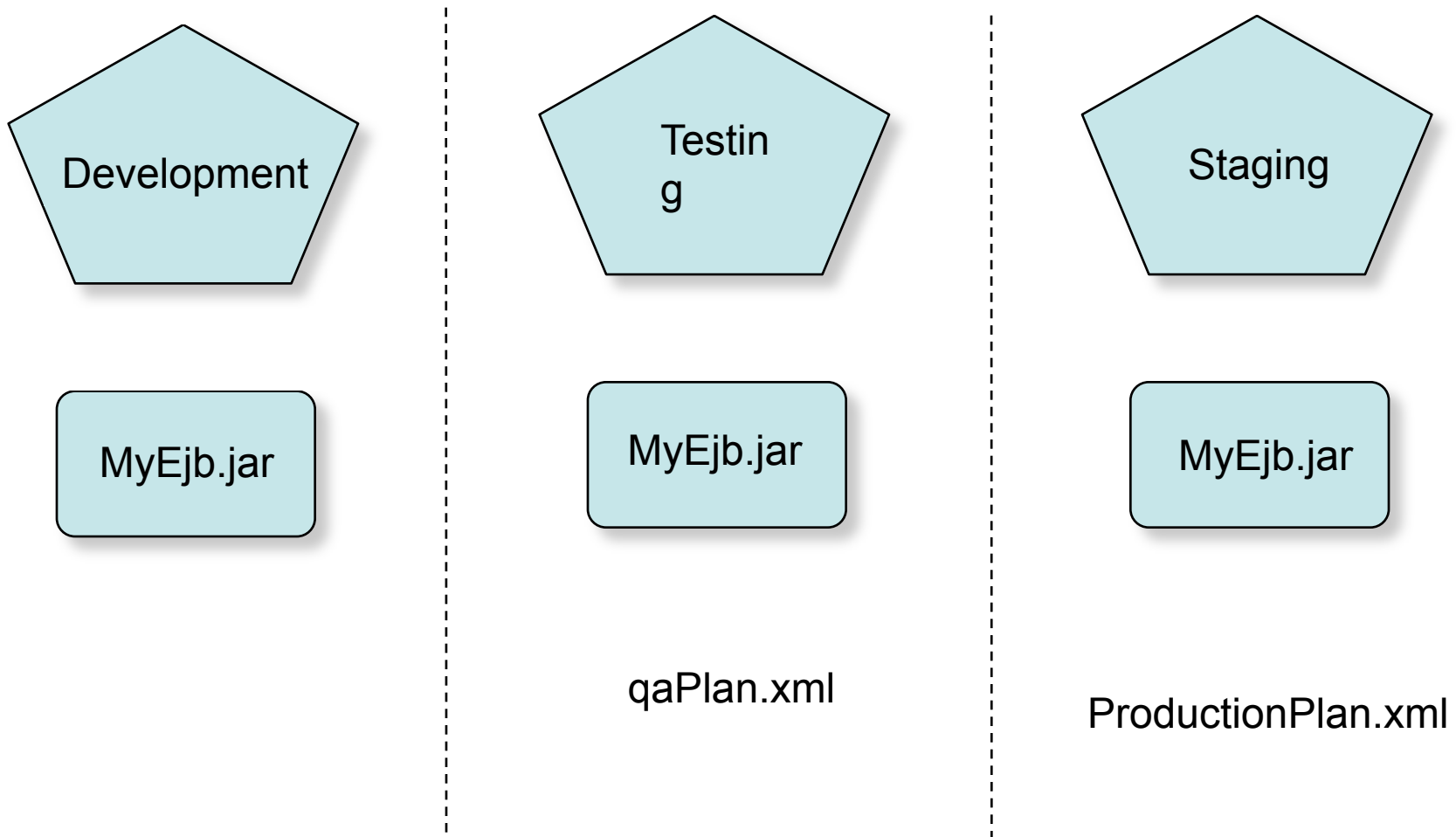
# Deployment Plans



# Deployment Plans



# Deployment Plans



# Deployment Plans – Tool Support

---

- ☕ Use *weblogic.PlanGenerator* for plan templates from an application.
- ☕ WebLogic Console
- ☕ `weblogic.Deployer -plan`



# Summary

---

- ☕ Application Libraries aid modular development and maintenance of Real World Applications
- ☕ Disruption free side-by-side deployments provide:
  - ➔ Higher Quality of Service
  - ➔ Test before exposing to the world
- ☕ Deployment Plans let administrators alter configuration via external plans





# Q&A

