

# Advanced Topics

# Objectives

After completing this lesson, you should be able to:

- List the best practices of using Service Bus
- Improve performance by caching service results
- Define SLA alert rules
- Explain how Oracle MFT works with OSB to handle large message transfers
- Describe message resequencing
- Build and deploy Service Bus projects with Maven



# Agenda

- Best practices
  - Service result caching
  - SLA alerts
  - Integrating with MFT
  - Message resequencing
  - Continuous integration with Maven

# Principles and Best Practices of Using Service Bus

- Keeping business logic out of the OSB layer
- Implementing well-defined integration patterns in the OSB, such as message transformation and routing
- Exposing standards-based interfaces from the OSB, wrapping proprietary interfaces with web services or JMS interfaces
- Transforming proprietary message formats to common business objects and hiding proprietary interface behavior
- Always configuring the service-level error handler
- Using Actions like Delete, Insert, Replace for minor update to a document

# BPEL Versus Service Bus

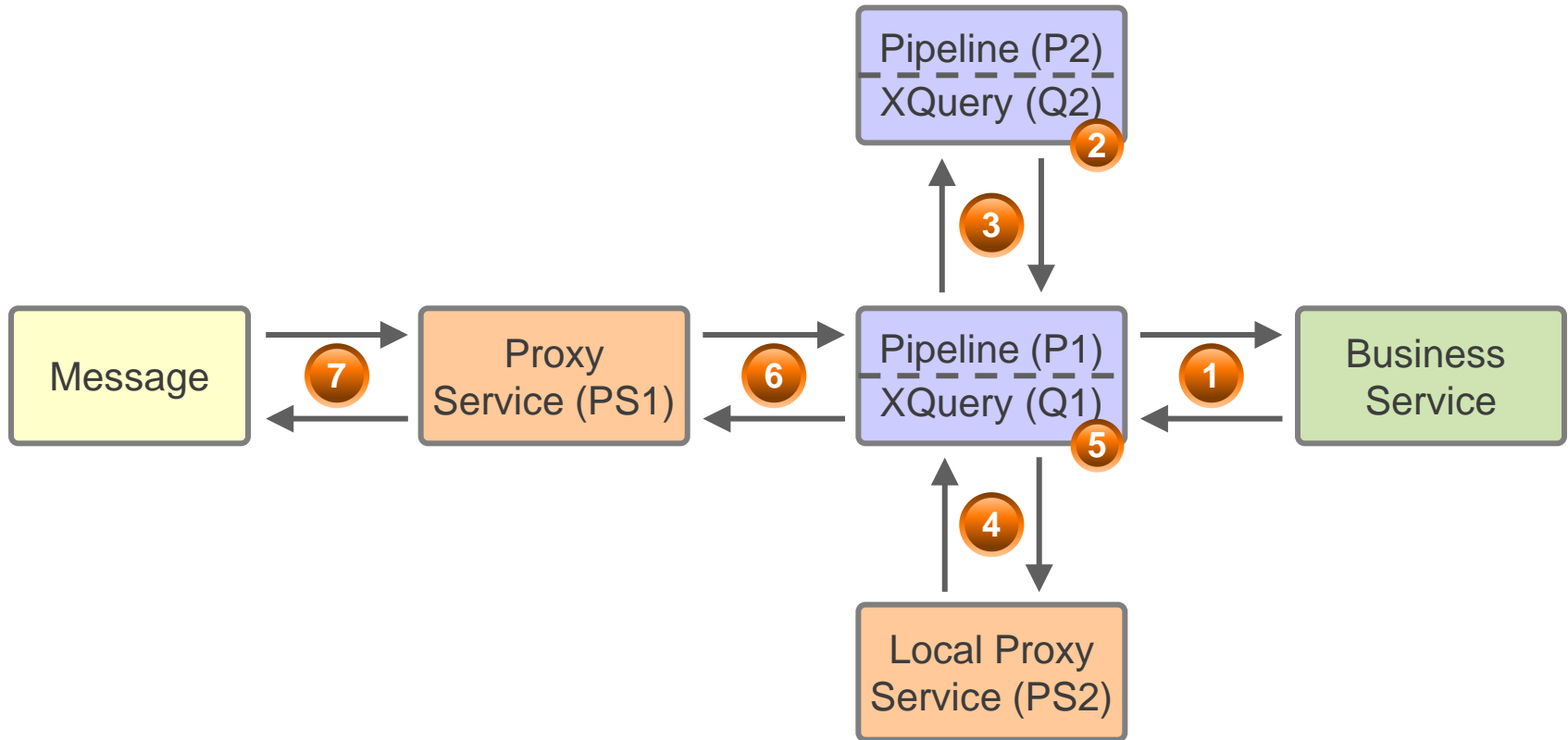
## **BPEL**

- Orchestrating service calls
- Primarily for stateful and long running processes
- Implementing SCA composite services
- Integrating Business Rules and Human Workflow

## **Service Bus**

- Implementing VETRO pattern
- Value mapping and cross-reference tables for canonical data models
- Stateless messaging

# Testing Approaches

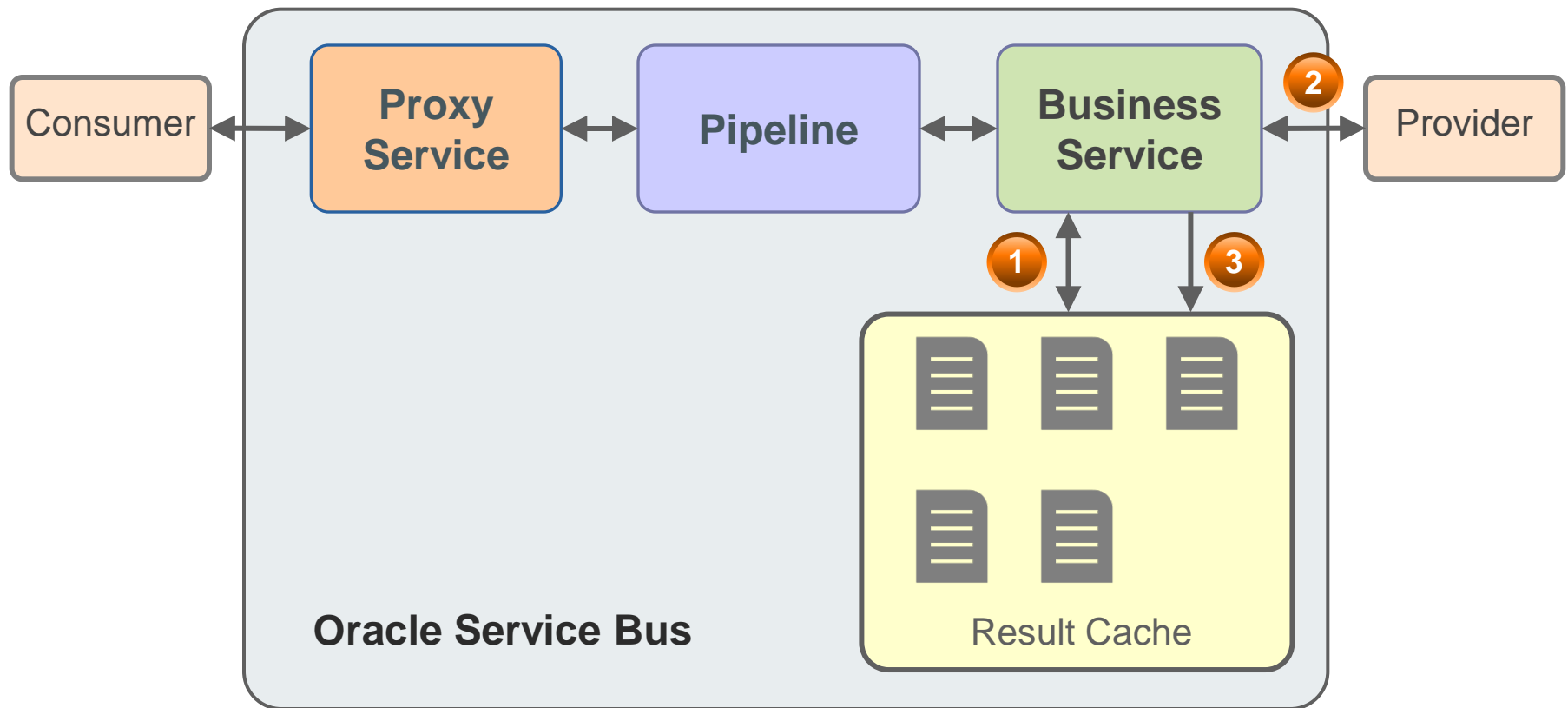


# Agenda

- Best practices
- **Service result caching**
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

# Service Result Caching

Requirement: Handle a lot of read-only requests to a service with limited capability.

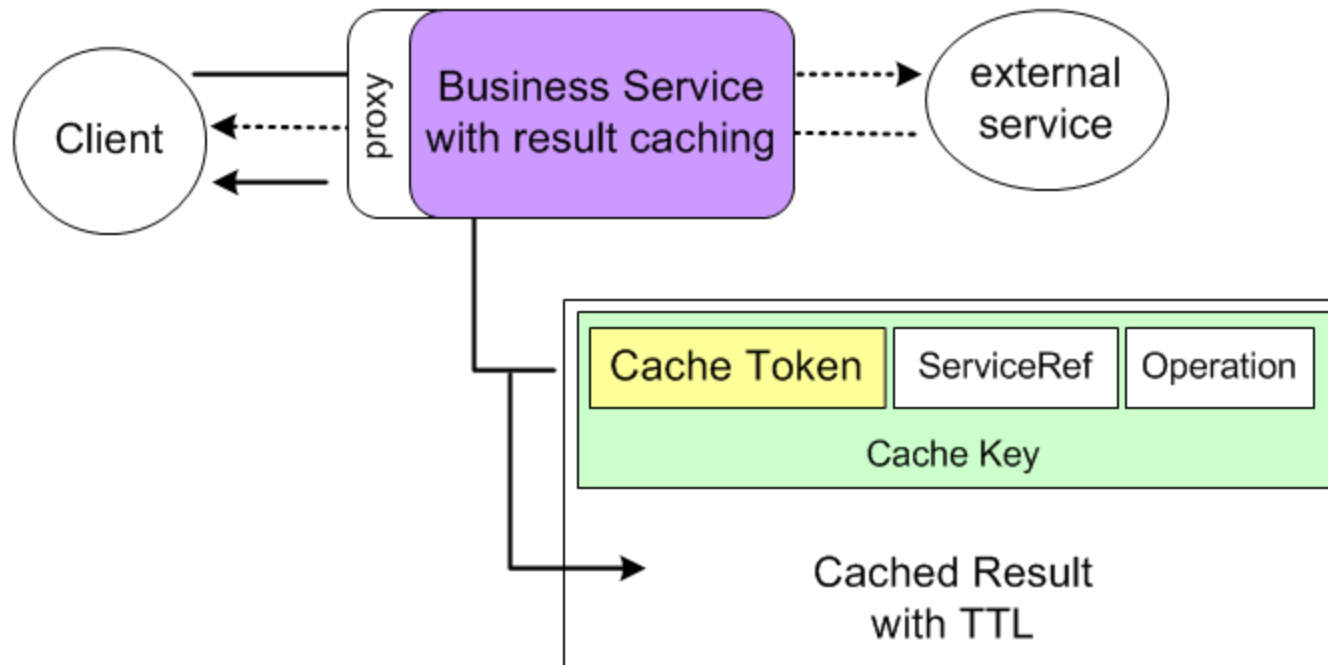




# Use Cases of Result Caching

- Synchronous services that return results that do not change often
- Handle a lot of read-only calls to a system with limited scalability

# How Does Result Catching Work?



# Configuring a Business Service for Result Caching

The screenshot shows the 'Performance Handling Configuration' page in the Oracle Service Bus console. The left sidebar contains a navigation menu with 'General', 'Transport', 'Transport Details', 'Message Handling', 'Performance', and 'Policies'. The 'Performance' tab is selected. The main content area is titled 'Performance Handling Configuration' and includes a sub-section 'Result Caching'. The 'Enable Result Caching' checkbox is checked. The 'Cache Token Expression' field contains the XPath expression '\$body/soas:PaymentInfo/soas:CardNum'. The 'Expiration Time' section has three options: 'Default', 'Duration', and 'Expression'. The 'Duration' option is selected, with a value of 6 Days, 0 Hours, 30 Minutes, and 0 Seconds. A red box highlights the 'Cache Token Expression' and 'Expiration Time' sections. A callout bubble points to the 'Expiration Time' section with the text 'Tune to make sure information is updated from time to time'.

**Performance Handling Configuration**  
Use this page to configure performance handling configuration such as result caching, etc

**Result Caching**

☒ Enable Result Caching

Cache Token Expression  
\$body/soas:PaymentInfo/soas:CardNum

Expiration Time

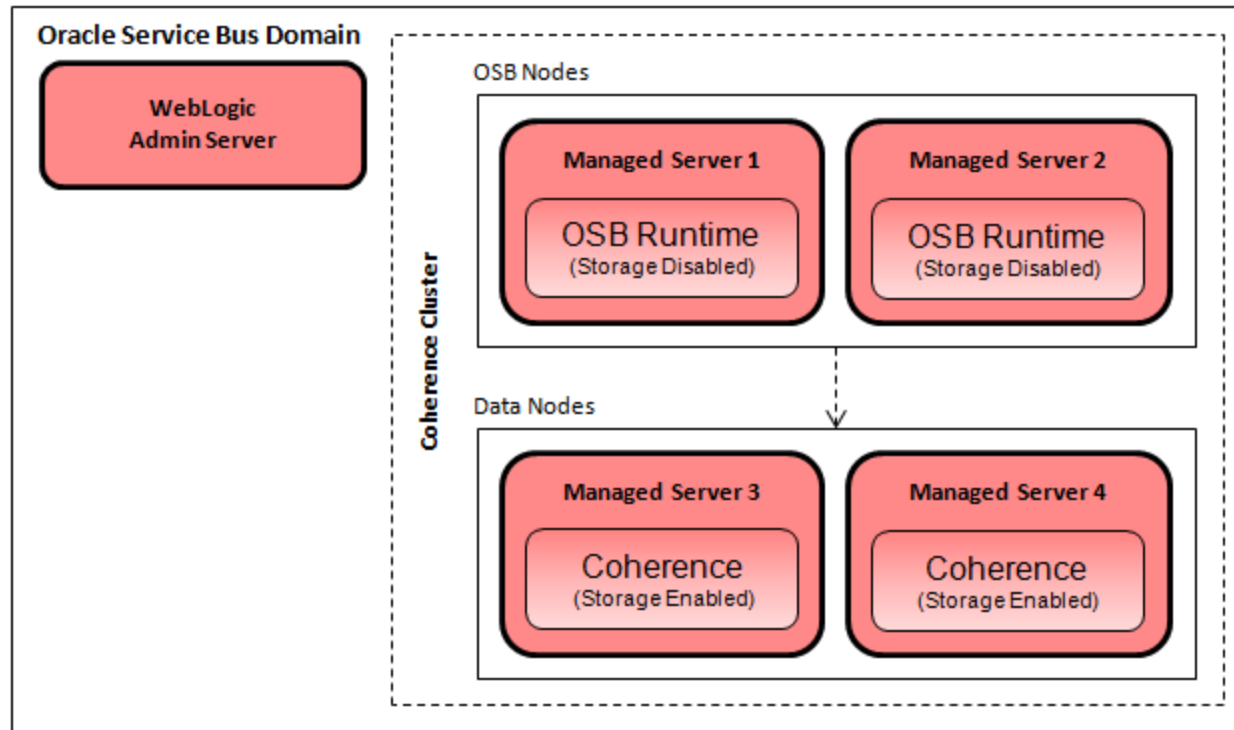
☐ Default

☒ Duration 6 Days 0 : 30 : 0 (hr:min:sec)

☐ Expression Request <Expression>

Tune to make sure information is updated from time to time

# Using an Out-of-Process Coherence Cache Server



# Coherence Configuration Files

- The configuration files are:
  - `osb-coherence-override.xml`:
    - Defines the operational settings for Coherence
    - Enables configuring Coherence clusters
  - `osb-coherence-cache-config.xml`:
    - Defines the cache configuration
- Both files are located in the `DOMAIN_HOME/config/osb/coherence` directory.

# Agenda

- Best practices
- Service result caching
- **SLA alerts**
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

# SLA Alert in OSB

- A service-level agreement (SLA) is a contract between a service provider and a service consumer:
  - Min/max response time
  - Message count
  - Error count
- SLA alert provides insight to health metrics of services, such as response time, errors, or even load.

# SLA Alert Rules

- SLA alerts:
  - Consist of a set of conditions, or “rules”
  - Are automatically evaluated after each service’s aggregation interval
- Define SLA alert rule to measure service health metrics:
  - Min/max response time
  - Message count
  - Error count



# Creating SLA Alert Rules

Create SLA Alert Rule

1

Rule Configuration

Rule Condition

\* Name

Error in PaymentValidation

Rule Description

Raise an SLA Alert if there are any errors in the current interval.

Rule Definition

Rule State

☒ Enabled

Summary

ValidatePayment Composite Failed

Alert Destination

AlertDestination

Path: PaymentValidation

Start Time

End Time

Expiration Date

Severity

Critical

Frequency

Every Time

Process Next Rule

Continue

Create SLA Alert Rule

2

Rule Configuration

Rule Condition

Condition Aggregation Interval

1 Min

Condition Builder

+

×

( )

↓

( )

Condition

☒

Count

Operation.validate.Error Count

>

0

Back

Next

Create

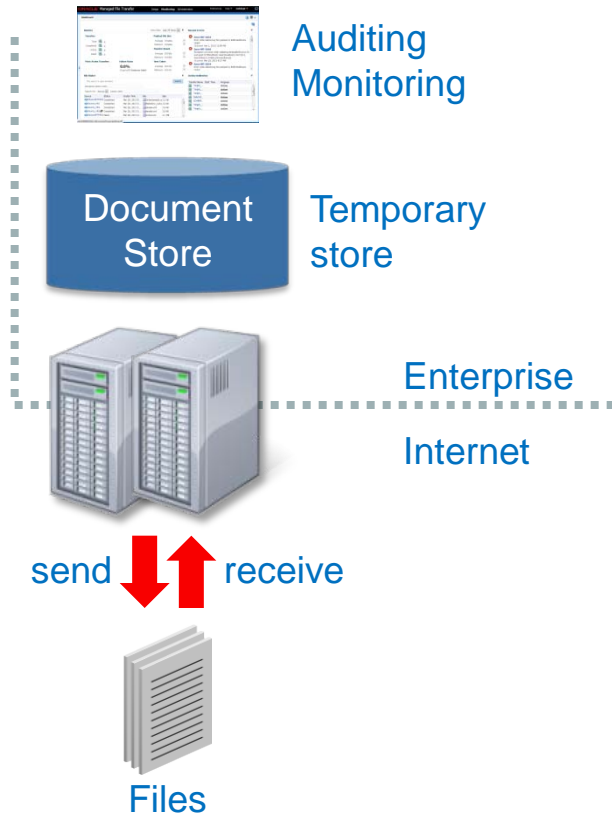
Cancel

# Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

# Managed File Transfer (MFT)

Simple and secure end-to-end managed file gateway

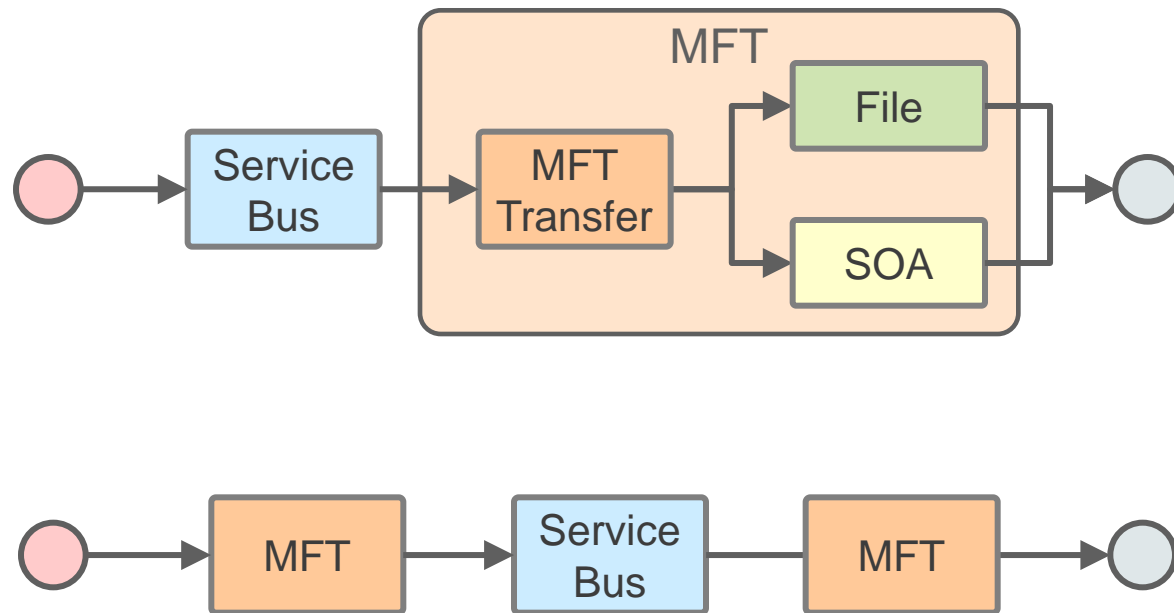


- Centralized file transfer infrastructure
- Supports:
  - Large file transfer
  - Pass-by-reference
  - Encryption
  - Auditing and monitoring
  - Scheduling
- Standard-based integration with SOA, OSB, B2B
- Highly-available/clusterable

# OSB and MFT Integration

Oracle Service Bus interface can be:

- The source or target of a transfer
- The common endpoint for the target of one transfer and the source of another



# Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- **Message resequencing**
- Continuous integration with Maven

# Resequencing Message

- The resequencer in Service Bus rearranges a stream of related but out-of-sequence messages into a sequential order.
- Resequencing is configured inside a pipeline component.
  - Only supports one-way WSDL-based service
- Resequencing order:
  - **Standard:** Is used when incoming message contain numeric identifier
  - **FIFO:** Is based on message arrival time
  - **Best Effort:** Is used for applications that produce a large number of messages in a short period and cannot provide identifier information.

# Configuring the Resequencer

PaymentValidationPP.pipeline

General  
Message Handling  
Operation Selection  
**Resequencer**

**Resequencer Configuration**  
Use this page to configure Resequencer options.

☒ Enable Resequencer

Resequence Level: ☒ Pipeline ☐ Operations

Resequence:

Dispatch Policy:

**Resequencer Options**

Group:

ID:

Start:

Increment:

Timeout:  sec.

Design Configuration

Specifies the Work Manager

# Work Manager in Service Bus

- WebLogic Server uses Work Manager to prioritize work and allocate threads for the applications/components.
- In Service Bus (OSB), several transports for proxy and business services provide a configuration option called *Dispatch Policy* that enables a Work Manager to associate a service to prioritize service work.



# Agenda

- Best practices
- Service result caching
- SLA alerts
- Integrating with MFT
- Message resequencing
- Continuous integration with Maven

# About Maven

Maven is essentially a project management and comprehension tool and provides a way to help with managing:

- Builds
- Documentation
- Reporting
- Dependencies
- SCMs
- Releases
- Distribution

# Concepts and Terminology

- **Build Life Cycle:** Made up of phases, like validate, compile, test, package, and so on
- **Repositories:** Hold build artifacts and dependencies of varying types
- **Plug-in:** Maven is a plug-in execution framework; all work is done by plug-ins
- **Goals:** A specific task, which can be mapped to one or more phases
- **Artifact** (deployment unit): Library or plug-in, identified by groupId + artifact ID + version + signed (signature)

Example:

```
mvn com.oracle.maven(groupId):oracle-maven-  
sync(artifact ID):push(goal)  
-DoracleHome=$ORACLE_HOME
```

# Concepts and Terminology

- POM files: Define Maven projects with a description of the project's artifacts:
  - Plug-ins to use
  - Inheritance
  - Dependencies on other artifacts
- Archetypes: A template for creating a specific type of project

# OSB Maven Support

- OSB provides:
  - A plug-in for pulling/downloading all requires libraries and plug-ins into local repository
  - A plug-in to package and deploy an OSB project
  - An archetype to create an OSB project
- Service Bus Maven plug-in goals:
  - Package
  - Deploy

# Summary

In this lesson, you should have learned how to:

- List best practices of using Service Bus
- Improve performance by caching service results
- Define SLA alert rules
- Explain how Oracle MFT works with OSB to handle large message transfers
- Describe message resequencing
- Build and deploy Service Bus projects with Maven



# Practice 13: Overview

- 13-1: Configuring Result Caching to Improve Service Performance
- 13-2: Adding a Service Level Agreement Alert
- 13-3: Building and Deploying Service Bus Projects with Maven (Optional)

# Building Service Bus Projects with Maven

- Maven 3.0.5 bundled in JDeveloper 12.1.3:  
`$FMW_DEV_HOME/oracle_common/modules/org.apache.maven_3.0.5`
- OSB plug-ins:  
`$FMW_DEV_HOME/oracle_common/plugins/maven/com/oracle/maven/  
oracle-maven-sync/12.1.3`

