

OSS through Java™ Technology Design Guidelines 1.2 Overview

Pierre Gauthier
Vincent Perrot
Gabriela Chiribau
Andreas Ebbert

Metasolv LLC
Sun Microsystems, Inc.
Covad Communications
Nokia Networks

OSS/J Design Guidelines 1.2

Industry proven EJB, XML and Webservices pattern

Learn about the latest development of the
OSS through Java Design Guidelines

Agenda

Design Guidelines 1.1 Patterns

What's New In Version 1.2

Plans For The Future

Q&A

Building on top of DG 1.1 Core Patterns

- **Session Façade Pattern**
- **Value Object Pattern, Data Transfer Object**
- **Factory Pattern, Data Transfer Object Factory**
- **Generic Attribute Access**
- **Version Number**
- **Iterators for Bulk Transfer (Value List Handler)**
- **Template-based Queries**
- **Generic Named Queries**
- **Bulk Operation with Best effort and Atomic Semantics**
- **Meta-Data Discovery**

Agenda

Design Guidelines 1.1 Patterns

What's New In Version 1.2

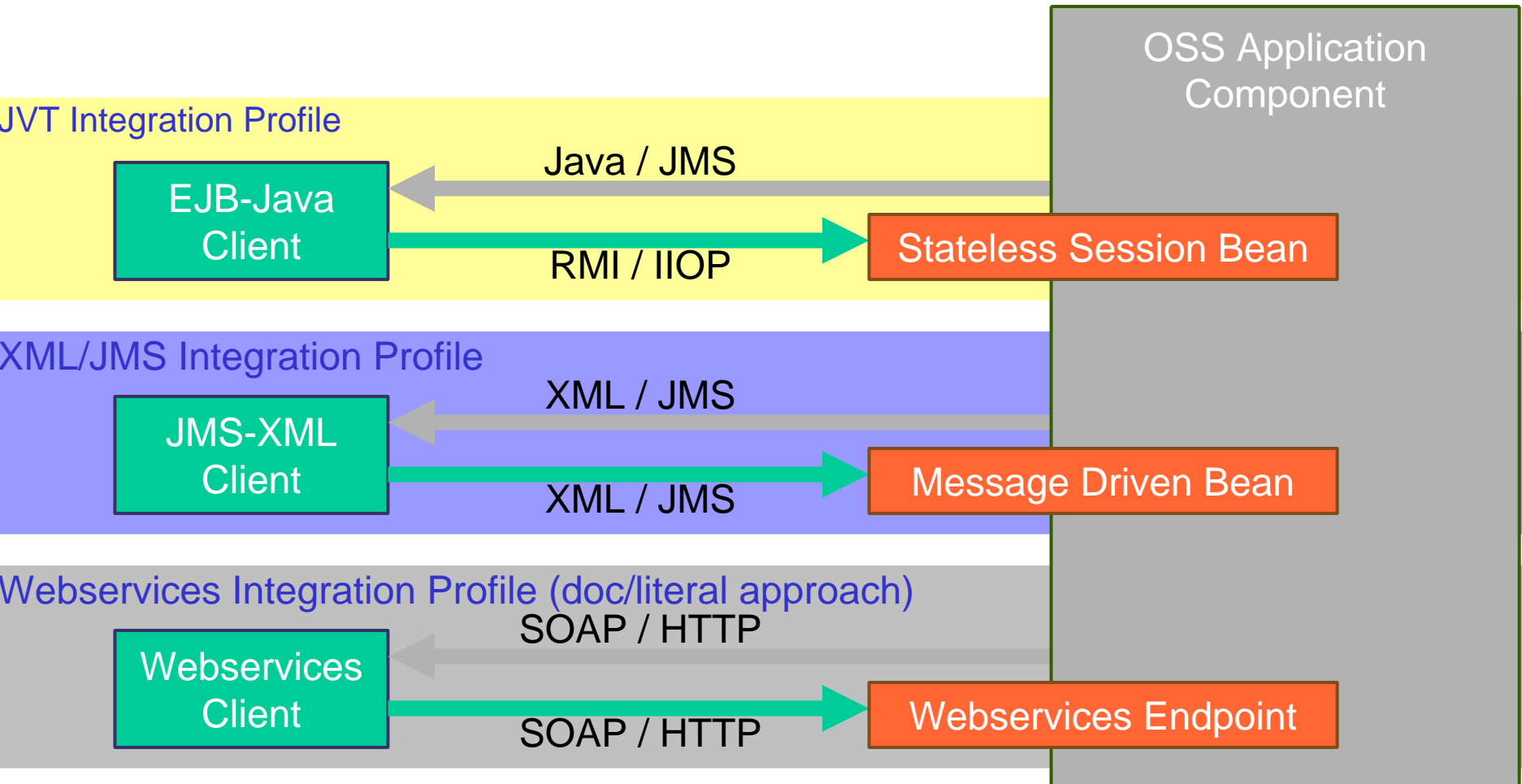
Plans For The Future

Q&A

What's New In Version 1.2

- **New integration profile Web Services:**
 - **Web Service Notification Management Model**
 - **Web Service Location Pattern based on UDDI**
 - **Web Service WSDL literal approach**
- **Enhancements for XML/JMS integration profile**
 - **XML Naming and Schema Versioning Guidelines**
- **Changes in the JVT integration profile:**
 - **Serializer Interface Deprecated**
- **Additions to all integration profiles:**
 - **Shared Information Model Pattern**
 - **OSS/J Exceptions**
 - **Modification to the Named Query Pattern**
 - **UpdateValue Procedure Pattern**

All Integration Profiles



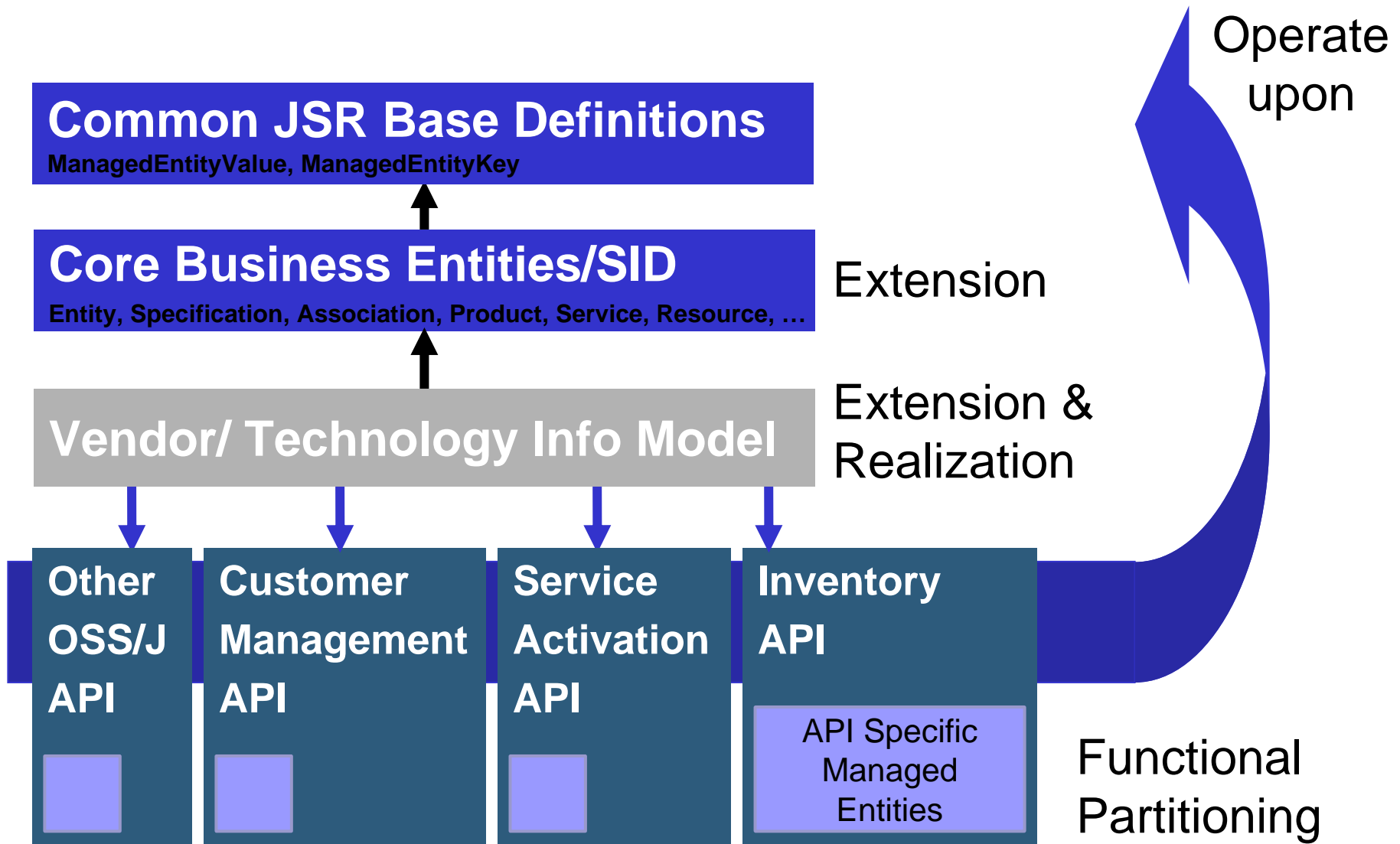
Next version of OSS/J APIs (1.2)

- **Inherit from OSS Common JSR (based on 1.2 DG)**
- **Start JSRs Maintenance Release**
- **All API's CBE based**
- **All API's WS, XML and JVT profiles**
- **Create/Improve TCKs for each Profile**
- **Improve and deprecate of some 1.1 interfaces**

OSS Common API, JSR 144 version 1.2

- **Improved CBE components,**
- **Integration Profiles (JVT, XML, WS)**
 - **Functional equivalence pattern**
 - **Use generation tools to create profiles**
 - **Certification of profile independently using the appropriated TCK**

Shared Info Model Pattern: Core Business Entities



Info Model Extensions

- **Model extensions in java.net as contribution**
<http://sbm-ossj.dev.java.net/>
- **API based certification established**
 - **TT API**
 - **Test CBE TT State Machine**
- **Model based certification in progress**
 - **X.790 Trouble Ticket**
 - **Use JCPsm to standardize information model using [javax.oss.infomodel](http://javax.oss.infomodel.org). ***

Agenda

Design Guidelines 1.1 Patterns

What's New In Version 1.2

Plans For The Future

Q&A

DG Evolution

- **J2SE 5.0 (Enumeration, Annotations, EJB 3.0)**
- **BPEL using OSS/J WS Endpoints**
- **Loosely Coupled Transaction Management Patterns (based on WS Transaction and/or Coordination)**
 - **Apply Conclusions to the JMS Profile**
- **Guidelines for Business Process Definitions using OSS/J as WS Endpoints**
- **WS Endpoint as Ports for Business Process (JBI, BPEL)**
- **Conversational Web Services and Iterators**
- **Information Model JSR**
- **Service Data Object investigation**

For More Information

- **OSS through Java™**
 - **Homepage:**
<http://ossj.org>
 - **Open source project:**
<http://sbm-OSSJ.dev.java.net>
- **EJB Design Pattern**
 - **Floyd Marinescu, "EJB Design Patterns"**
<http://www.theserverside.com/books/wiley/EJBDesignPatterns/>
 - **Core J2EE patterns**
<http://java.sun.com/blueprints/corej2eepatterns/Patterns/index.html>

Q&A

OSS/J

OSS through Java™ Technology Design Guidelines 1.2 Overview

Pierre Gauthier
Vincent Perrot
Gabriela Chiribau
Andreas Ebbert

Metasolv LLC
Sun Microsystems, Inc.
Covad Communications
Nokia Networks

Design Guidelines 1.1 Review

Part of JSR 144, the OSS Common API

- **Foundation of all OSS/J APIs**
- **Collection and Implementation of Design Pattern**
- **Defines two integration profiles:**
 - **JVT (RMI/IIOP): synchronous interface**
 - **XML/JMS: asynchronous interface**
- **Is implemented as standard interface design procedure in many OSS/J member companies**

Session Façade Pattern

From: Floyd Marinescu, "EJB Design Patterns", page 3

- **Description:**

- The most widely used of all EJB design patterns, the Session Façade shows how to properly partition the business logic in your system to help minimize dependencies between client and server, while forcing use cases to execute in one network call and in one transaction.

- **Application:**

- The Design Guidelines define a `JVTSession` interface, which all OSS/J APIs extend.

Value Object Pattern Data Transfer Object

From: Floyd Marinescu, "EJB Design Patterns", page 45

- **Description:**

- The essential design pattern. Discusses why, how, and when to marshal data across the network in bulk bundles called data transfer objects (DTOs).

- **Application:**

- The ManagedEntityValue (MEV) is the base class for all objects passed between client and server.

Factory Pattern, Data Transfer Object Factory

From: Floyd Marinescu, "EJB Design Patterns", page 3

- **Description:**

- Debunks the old practice of placing DTO creation/consumption logic on the entity bean itself and prescribes centralizing data transfer object creation and consumption logic into a single layer (implemented as session beans or plain java factories).

- **Application:**

- The JVTSession serves as a factory for the MEVs, which in turn serve as factory for any objects needed to populate the attributes in the MEV.

Footnote: Add sources or other footnotes here.

Generic Attribute Access

From: Floyd Marinescu, "EJB Design Patterns", page 4

- **Description:**
 - This pattern discusses when and how to provide a domain-generic interface to the attributes of an entity bean for maintainability and performance purposes.
- **Application:**
 - The MEV extends the `AttributeAccess` interface that contains means to read and write all attributes.
- **Extension:**
 - In addition, the `AttributeAccess` interface allows for MEV only to carry a subset of the specified attributes (called populated) to save bandwidth.

Footnote: Add sources or other footnotes here.

Version Number

From: Floyd Marinescu, "EJB Design Patterns", page 69

- **Description:**
 - **Used to program your entity beans with optimistic concurrency checks that can protect the consistency of your database, when dealing with use cases that span transactions and user think time.**
- **Application:**
 - **Not applied in entity beans, but in the MEV. It has a field called lastUpdateVersionNumber.**

Footnote: Add sources or other footnotes here.

Iterators for Bulk Data Transfer Value List Handler

From: Sun, Core J2EE patterns

- **Description:**

- Use a Value List Handler to control the lookup, cache the results, and provide the results to the client in a result set whose size and traversal meets the client's requirements.

- **Application:**

- The Design Guidelines define a `ManagedEntityValueIterator`, which shall be further specified in every API domain to handle the domain type. Every query to the `JVTSession` returns such an iterator.

Footnote: Add sources or other footnotes here.

Template-based Queries

- **Description:**
 - Use this pattern to query for all managed entities which match an example in all provided attributes.
- **Application:**
 - The JVTSession includes query methods, which take a MEV as input and returns an iterator. The query result contains all MEVs, that match the template in all populated attributes.

Generic Named Queries

- **Description:**
 - An implementation wants to introduce new query capabilities, which a standard interface definition cannot foresee. The generic query interface provides this functionality.
- **Application:**
 - The JVTSession has query methods that take an abstract QueryValue interface as input. The concrete occurrence of query values and its parameters are provided by the implementation.

Footnote: Add sources or other footnotes here.

Bulk Operations with Best Effort and Atomic Semantics

- **Description:**
 - Beside single parameter methods, a session façade should also accept a list of input parameters in bulk methods to save remote calls.
- **Application:**
 - The JVTSession provides those bulk methods by default, and differentiates between best effort and bulk semantics, by prefixing best effort methods with the keyword "try".

Footnote: Add sources or other footnotes here.