Design of the OSS Common API Reference Implementation (JSR 144)

OSS through Java™ Initiative

Vincent Perrot, Sun Microsystems Inc.

COM-API-Ri_Design.1.3.doc

Executive Summary

The Common API offers interfaces and classes, which are common across all OSS API defined under OSS J Initiative. This document describes how the Reference Implementation was designed to implement the Common API Specifications.

Table of Contents

Preface

Objectives

Design description of the OSS/J Common Reference Implementation

Audience

The target audiences are

- Developers who seek information about how the Common API can be implemented
- Developers of other OSS J API Reference Implementers
- Developers who want to make use of these API and extend its implementations

Related Information

Revision History:

Date	Version	Author	State	Comments
03-01-2002	1.0	OSS Common Team	final	First version
March 2005	1.1	Vincent Perrot Sun Microsystems Inc	Maintenance release	Include CBE 1.2.1
March 2005	1.2	Vincent Perrot Sun Microsystems Inc	Maintenance release 2	
February 2006	1.3	Vincent Perrot Sun Microsystems Inc	Maintenance release 3	

1 Introduction

This document describes the design of the OSS through Java™ Initiative, Common API Reference Implementation.

The Reference Implementation can be used either as a proof-of-concept for the Common API specification, showing that it is possible to implement the API or API's can be directly used as a package. The Reference Implementation consists of two parts. The first part, consists of abstract classes and interfaces which can be reused by the target audiences of this API directly and the second part consists of concrete implementation the Interfaces of the API this part cant be reused and this only serves as proof or example how to implement the API.

This document shows how the Reference Implementation is designed. Reference Implementation provides the concrete interfaces and classes as specified in the Common API Specifications. It also provides some abstract classes for certain interfaces so that the developers classes can directly extend these classes and implement only those none generic methods.

In general the concrete Reference Implementation is designed as a set of Enterprise Java Beans. The entire RI is developed using J2EE SDK 1.4 as application server available for download at http://java.sun.com/j2ee.

2 Design Overview of Reusable Part of Reference Implementation

The Reference Implementation has following set of classes apart from the classes and interfaces specified by the specification. Each of these classes is explained in detail in later part of the document.

- Application Context Implementer class
- Attribute Access classes
- Base Java Value Type and CBE implementations

All the reusable classes are archived in the jar named oss_common_ri-1.3.jar. This jar also includes all the CBE implementation classes. It can be included in any application deployable archive.

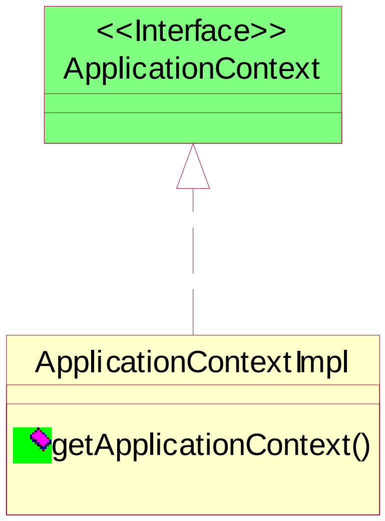
2.1 Application Context Implementer class

The Application Context Implementer class contains the URL and other system properties required to set up an initial connection with the JNDI provider into which the components in charge of that managed entity are registered. This class implements the Application Context Interface defined in the specification apart from this it provides additional static method, which provides the Application context based on the present server configuration.

The figure below shows the relationship between the interface and class

Green color indicates the Interface is part of the specification

Yellow class is additional class defined in Reference Implementation



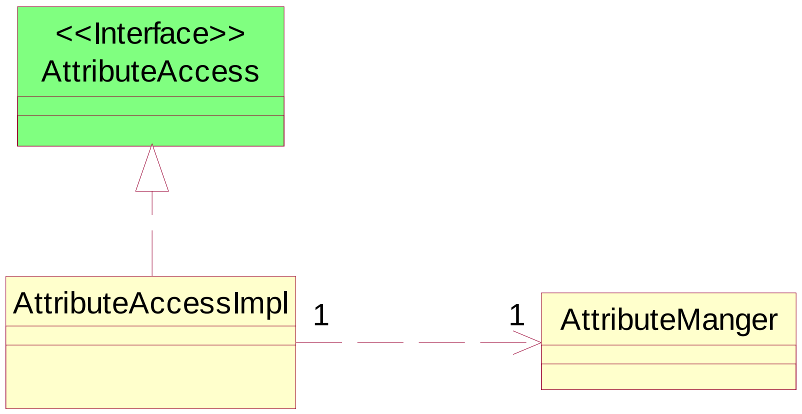
2.2 Attribute Access Classes

Two classes namely AttributeManager and AttributeAccessImpl are the classes, which help in accessing the attributes of value object as specified in the specification. Attribute Manager manages the attributes of the value object and provides easy methods to get the properties of attributes like attribute names, settable attributes etc. AttributeAccessImpl is abstract class implementing the AttributeAcess Interface, which all value objects must implement according to the specification. This abstract class manages the attributes using the AttributeManager class.

The figure below shows the relationship between the interface and classes

Green color indicates the Interface is part of the specification

Yellow classes are addition classes defined in Reference Implementation



2.3 Base Java Value Type and CBE implementations

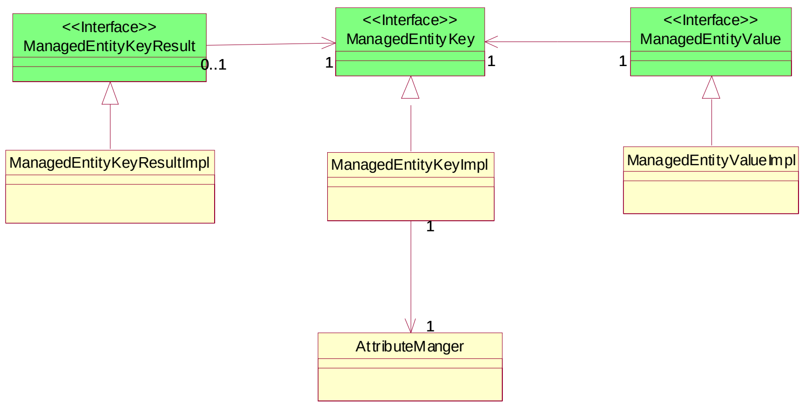
The Java Value Types are the objects, which are exchanged between the client and JVTSessionBean.The following classes are defined in addition to those specified in the specification

- ManagedEntityKeyImpl
- ManagedEntityKeyResultImpl
- ManagedEntityValueImpl

The figure below shows the relationship between the interface and classes

Green color indicates the Interface is part of the specification

Yellow classes are addition classes defined in Reference Implementation



ManagedEntityKeyImpl is an abstract class, which implements the ManagedEntityKey Interface. Since very managed Entity type must have a ManagedEntityKey Interface implementation it can directly extend ManagedEntityKey abstract class and define the methods to make the primary key which will be specific to the entity type, functionalities like checking equality of two key objects as specified in the specification is taken care by the abstract class. ManagedEntityvalueImpl implements the ManagedEntityValue Interface as specified in the specification and it also extends the AttributeAccessImpl class.

Then all the CBE implementation classes are derived from these based definitions.

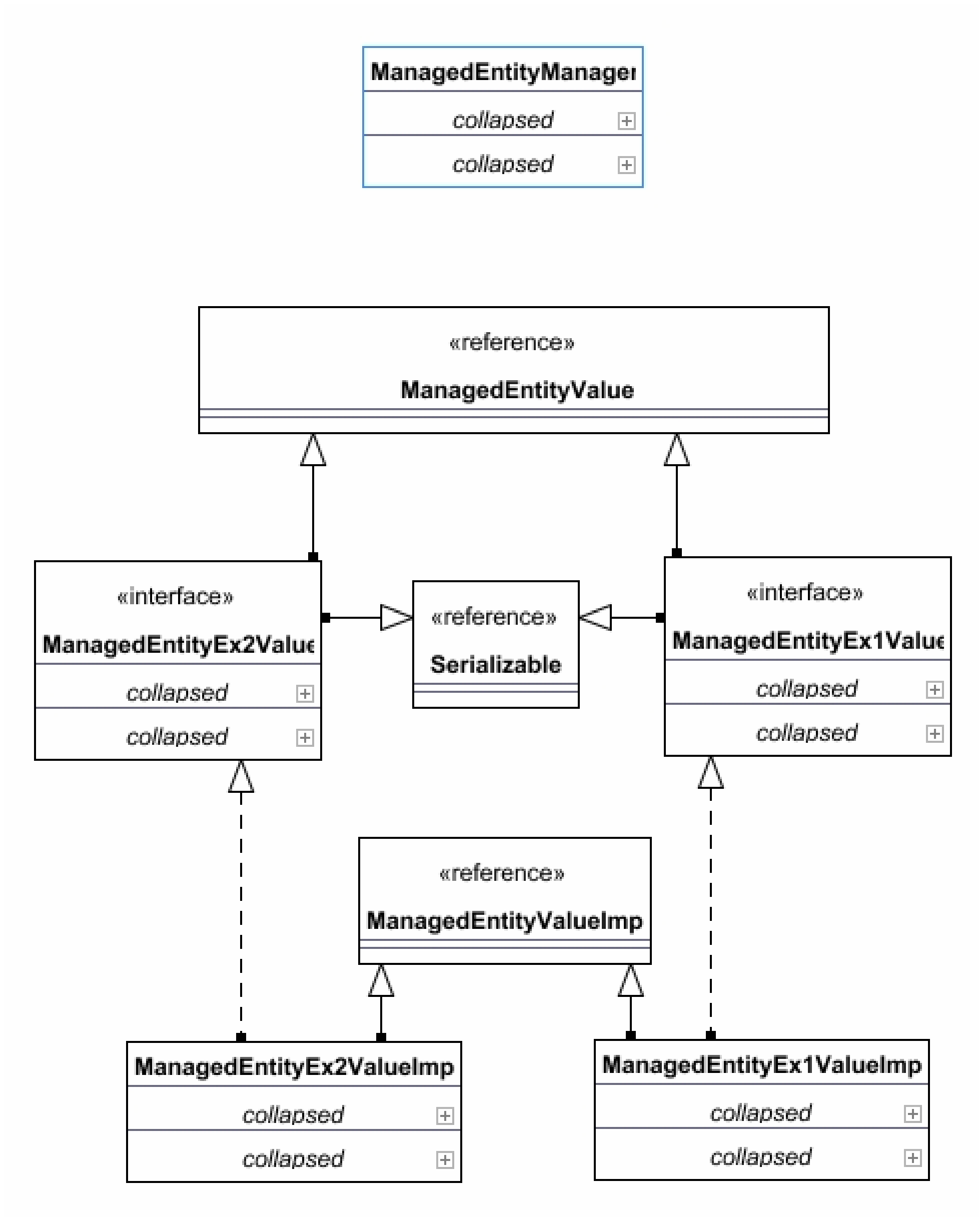
3 Design overview of Non Reusable Part of Reference Implementation

This part of Reference Implementation consists of concrete example Implementation of API. All the classes are archived into the oss_common_ri_ejb-1.1.jar file. Since this Example tries to show simple implementation of overall Common API the Mange Entities created in this example are not pertitient. The emphasis in this example is on to show how to create Managed Entities and how to manage these using corresponding Managed entity values from the Clients.

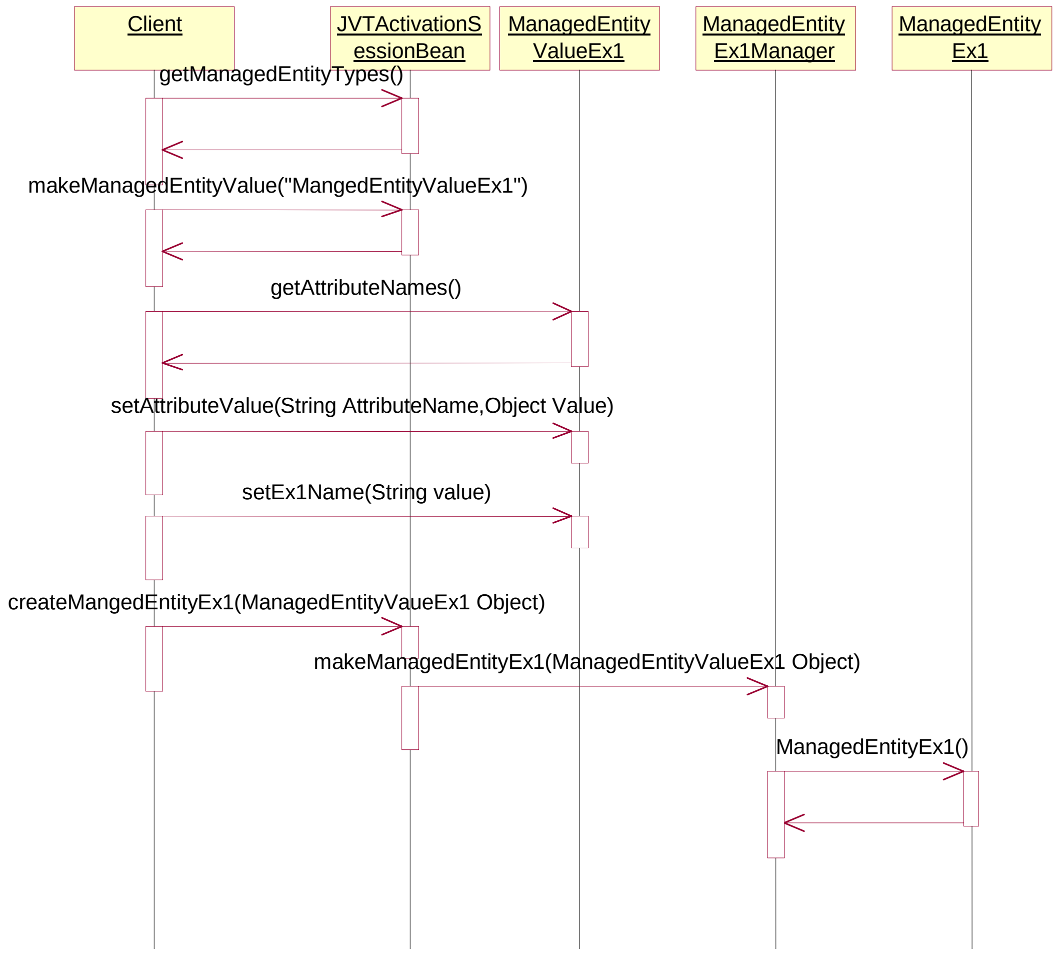
The Reference Implementation example consists of the following main classes

- JVTActivationSessionBean class
- ManagedEntityEx1 and ManagedEntityEx1Value interfaces and classes
- ManagedEntityEx2 and ManagedEntityEx2Value interfaces and classes
- ManagedEntityEx1Manager class
- ManagedEntityEx2Manager class
- QueryAllManagedEntityEx1 class

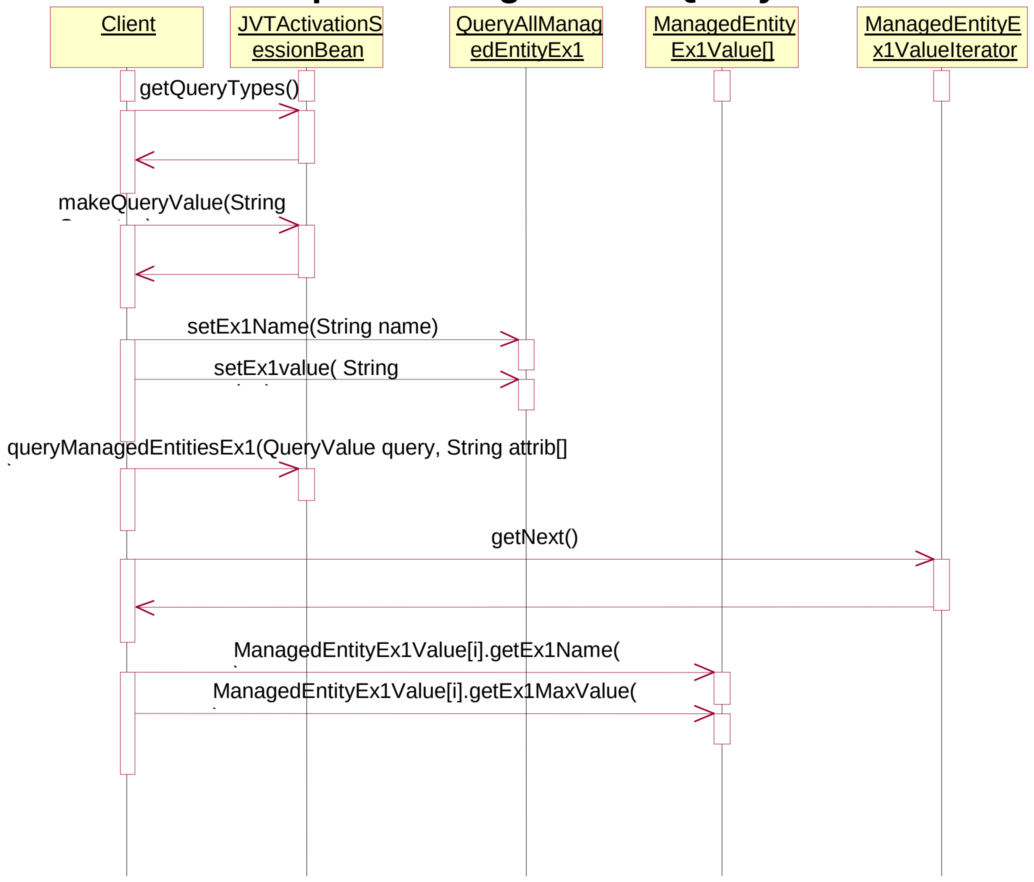
The class and sequence diagram of the Example Implementation is given below


ManagedEntityManager
collapsed
collapsed
«reference»
ManagedEntityValue
«interface»
ManagedEntityEx2Value
collapsed
collapsed
«reference»
Serializable
«interface»
ManagedEntityEx1Value
collapsed
collapsed
«reference»
ManagedEntityValueImp
ManagedEntityEx2ValueImp
collapsed
collapsed
ManagedEntityEx1ValueImp
collapsed
collapsed

Sequence Diagram For Creation of ManagedEntityEx1


Client
JVTActivationSessionBean
ManagedEntityValueEx1
ManagedEntityEx1Manager
ManagedEntityEx1
getManagedEntityTypes()
makeManagedEntityValue("MangedEntityValueEx1")
getAttributeNames()
setAttributeValue(String AttributeName,Object Value)
setEx1Name(String value)
createMangedEntityEx1(ManagedEntityVaueEx1 Object)
makeManagedEntityEx1(ManagedEntityValueEx1 Object)
ManagedEntityEx1()

Sequence Diagram For Query



Appendix A: Glossary and References

References

[1] “Shared Information/Data (SID) Model-Phase VI : Concepts, Principles, and Business Entities” GB922 v3.0., Telemanagement Forum, http://www.tmforum.org

[2] “OSS through Java Design Guidelines” v 1.2, OSS through Java^{TM} Initiative, http://www.ossj.org

[3] “Enterprise JavaBeansTM Specification”,Version 2.1, June 3 2003

, Spec leader: Linda G. DeMichiel, http://java.sun.com/j2ee