Change Log for OSS Common API version 1.1

OSS through Java™ Initiative

Vincent Perrot, Sun Microsystems, Inc.

COM-API-SPEC_change_log.1.1.1.doc

Executive Summary

This document summarizes the changes to the OSS Common API (JSR 144) specification Version 1.1. The main purpose of this version was to add the Core Business Entities (CBE) interfaces.
However, since maintenance release to the specification was taking place, additional modifications to the previously existing Java Value Type interface were also incorporated.

There are two lists of changes:

- "proposed" changes are those modifications that are included in OSS Common API version 1.1.
- "deferred" changes are those modifications that are not included in OSS Common API version 1.1, whether for time reasons or because it was considered that the changes were too significant.

The detailed description of changes in this document is principally of interest to people implementing the OSS Common API specification.

Table of Contents

1 Preface

1.1 Objectives

This document lists all the changes that have been requested for the maintenance release v1.1 version of the OSS Common API, JSR 144.

The changes have been collected through:

- Bug parade: Bug and Request For Evolution (RFE) submitted by Java developers
- OSS/J specification leaders: evolution necessary to incorporate new common objects and to improve the common interfaces and Reference Implementation that will be "inherited" by all maintenance releases of the existing OSS APIs
- OSS/J Architectural Board: The common API needs to reflect the necessary new architectural recommendation (new CBE, etc)

1.2 Audience

This document is used to start a Maintenance Release of the OSS Common API JSR 144.

According to the JCP^{SM}:

The Maintenance Lead (ML) will arrange to have all change items placed into the PROPOSED section of the Change Log (this document) and then send a request to the PMO to initiate a Maintenance Review. The PMO will make a public announcement and begin the review.

1.3 Approval and Distribution

The ML may choose to modify one or more of the proposed changes based on comments received during review.

1.4 Related Information

oss_commmon-1_0-fr-spec.zip: contains the Version 1.0 of the OSS common API, JSR 144, http://java.sun.com/products/oss/start_download.html

1.5 Revision History

Date	Version	Author	State	Comments
Aug 11th, 200	0.1	Vincent Perrot	Initial	Include change requests from: • Nortel • Sqm (MBT) • Inventory (Metasolv) • SA (Nokia) • IPB (NEC) • Bug/RFE logged
Aug 12th, 200	0.1_metas	Pierre Gauthier	Metasolv Review	Pierre's Comment
Aug 12th, 200	0.2	Vincent Perrot	Common review	Remove rejected items and update the accepted
Aug 20th, 200	1.1	Vincent Perrot	final	Edit and format text for JCP submissi
Oct 1th, 2004	1.1.1	Vincent Perrot	Final after Vote and comment	• Move ProductOffering CBE to productoffering package • Add ServiceKeyResult • Add Spec doc reformat

2 Summary of proposed changes

- Specification Document format
- Add CBE Common interfaces
- Improved XML Integration Profile
- Fix Exception thrown by AttributeAccess
- Add Attribute Name to UnsupportedAttributeException message
- Improve Documentation around QueryValue Pattern
- Improve equals() by adding the check of the object type
- Common API compliant to DG 1.2
- Consistent Exception Handling
- Relocation of the (JSR 130 IPB) *activity* package
- Local JVT Session
- Improve Weakly Typed Arguments

3 Proposed changes

3.1 Specification Document format

Reformat the Specification Document using the latest OSS/J template.

3.2 Add CBE Common interfaces

Core Business Entities (CBE) describes the foundation for a data model by defining, modeling and implementing core concepts including products, services and resources, etc.

The JSR 144 has the opportunity to improve the efficiency of API developers and maintain consistency by defining, modeling and implementing these core concepts. This work effort leverages work already in progress being carried out by the TeleManagement Forum’s New Generation OSS (NGOSS) Shared Information/Data (SID) Model team.

The CBE white paper available at http://ossj.org/downloads/cbe.shtml contains all the detailed information.

The following java packages and interfaces are added to the OSS Common API to include all the interfaces common to OSS/J interfaces:

The **CBE Core package** “`javax.oss.cbe`” basically contains the following CBE Entities and their keys:

- `javax.oss.cbe.AssociationKey`
- `javax.oss.cbe.AssociationKeyResult`
- `javax.oss.cbe.AssociationKeyResultIterator`
- `javax.oss.cbe.AssociationRule`
- `javax.oss.cbe.AssociationRuleViolationException`
- `javax.oss.cbe.AssociationValue`
- `javax.oss.cbe.AssociationValueIterator`
- `javax.oss.cbe.Cardinality`
- `javax.oss.cbe.CBECoreType`
- `javax.oss.cbe.CBEManagedEntityValue`
- `javax.oss.cbe.CBEManagedEntityValueIterator`
- `javax.oss.cbe.EntityKey`
- `javax.oss.cbe.EntityKeyResult`
- `javax.oss.cbe.EntityKeyResultIterator`
- `javax.oss.cbe.EntitySpecificationKey`
- `javax.oss.cbe.EntitySpecificationKeyResult`
- `javax.oss.cbe.EntitySpecificationKeyResultIterator`
- `javax.oss.cbe.EntitySpecificationValue`

- `javax.oss.cbe.EntitySpecificationValueIterator`
- `javax.oss.cbe.EntityValue`
- `javax.oss.cbe.EntityValueIterator`
- `javax.oss.cbe.HierarchicalPrimaryKey`

The **CBE Service, Resource and Product packages** defines a basic Service, Resource and Product Model:

- `javax.oss.cbe.service.ServiceAggregatesServiceAssocValue`
- `javax.oss.cbe.service.ServiceAssociationKey`
- `javax.oss.cbe.service.ServiceAssociationValue`
- `javax.oss.cbe.service.ServiceKey`
- `javax.oss.cbe.service. ServiceSpecificationAggregatesServiceSpecificationAssocValue`
- `javax.oss.cbe.service.ServiceSpecificationKey`
- `javax.oss.cbe.service.ServiceSpecificationValue`
- `javax.oss.cbe.service.ServiceState`
- `javax.oss.cbe.service.ServiceValue`
- `javax.oss.cbe.service.ServiceKeyResult`
- `javax.oss.cbe.service.StartMode`
- `javax.oss.cbe.resource.ResourceAggregatesResourceAssocValue`
- `javax.oss.cbe.resource.ResourceAssociationKey`
- `javax.oss.cbe.resource.ResourceAssociationValue`
- `javax.oss.cbe.resource.ResourceKey`
- `javax.oss.cbe.resource.ResourceSpecificationKey`
- `javax.oss.cbe.resource.ResourceSpecificationValue`
- `javax.oss.cbe.resource.ResourceValue`
- `javax.oss.cbe.product.ProductAggregatesProductAssocValue`
- `javax.oss.cbe.product.ProductAssociationKey`
- `javax.oss.cbe.product.ProductAssociationValue`
- `javax.oss.cbe.product.ProductKey`
- `javax.oss.cbe.product. ProductSpecificationAggregatesProductSpecificationAssocValue`
- `javax.oss.cbe.product.ProductSpecificationKey`
- `javax.oss.cbe.product.ProductSpecificationValue`
- `javax.oss.cbe.product.ProductStatus`
- `javax.oss.cbe.product.ProductValue`
- `javax.oss.cbe.product.productoffering.ProductOfferingKey`
- `javax.oss.cbe.product.productoffering.ProductOfferingValue`

The **CBE Location package** “`javax.oss.cbe.location`” contains the following interfaces:

- `javax.oss.cbe.location.FormattedAddress`
- `javax.oss.cbe.location.GeographicAddressValue`
- `javax.oss.cbe.location.GeographicLocationValue`
- `javax.oss.cbe.location.LocationKey`

- `javax.oss.cbe.location.LocationValue`
- `javax.oss.cbe.location.MSAGAddress`
- `javax.oss.cbe.location.UrbanPropertyAddress`

The **CBE Party** package “`javax.oss.cbe.party`” contains the following CBE Managed Entity Values and their keys:

- `javax.oss.cbe.party.PartyKey`
- `javax.oss.cbe.party.PartyRoleKey`
- javax.oss.cbe.party.PartyRoleValue
- javax.oss.cbe.party.PartyValue

The **CBE DataTypes** package “`javax.oss.cbe.datatypes`” contains common data type interfaces:

- `javax.oss.cbe.datatypes.Identity`
- `javax.oss.cbe.datatypes.IndividualName`
- `javax.oss.cbe.datatypes.LifeCycleStatus`
- `javax.oss.cbe.datatypes.OrganizationName`
- `javax.oss.cbe.datatypes.Quantity`
- `javax.oss.cbe.datatypes.TimePeriod`

The **CBE Cross** package “`javax.oss.cbe.cross`” contains common data type interfaces:

- `javax.oss.cbe.cross.CrossAssociationKey`
- `javax.oss.cbe.cross.CrossAssociationValue`

The classes listed in this section are subject to change to follow the latest CBE model definition from the OSS/J CBE team.

3.3 Improved XML Integration Profile

Since the initial version of the XML schema used to implement the XML/JMS integration profile, the OSS/J XML generation tools have missing common object definitions and extensibility issues in the *XmlCommonSchema.xsd* file.

Some of those issue have been recorded in the bug ID 4925232 : Current XmlCommonSchema.xsd does not allow for valid XML.

The new XML schema will also include a versioning mechanism. This may affect the schema file name and the schema file content. These modifications will be aligned with the J2EE recommendation regarding XML schemas.

3.4 Fix Exception thrown by AttributeAccess

The javax.oss.UnsupportedAttributeException is added to the isPopulated() and unpopulateAttribute() methods of the javax.oss.AttributeAccess interface to be consistent with the definition of the rest of the methods of this interface.

```
   /** Check if a specific attribute is populated.
     * @param attributeName the name of the attribute which is to be checked for
population.
     * @exception java.lang.IllegalArgumentException
     * (the attribute name is null or is not recognized, or the value is bad)
     * @exception javax.oss.UnsupportedAttributeException
     * (the attribute is optional and is not supported)
     * @return true, if this attribute is populated, false otherwise.
     * @see #isFullyPopulated()
     */
    public boolean isPopulated( String attributeName)
    throws java.lang.IllegalArgumentException,
              javax.oss.UnsupportedAttributeException;



    /**
     * Mark a single attribute as unpopulated.
     * After this call getAttribute( String attributeName ) must
     * raise the IllegalStateException.
     *
     * @param attributeName name of the attribute to be unpopulated.
     * @exception java.lang.IllegalArgumentException
     * (the attribute name is null or is not recognized, or the value is bad)
     * @exception javax.oss.UnsupportedAttributeException
     * (the attribute is optional and is not supported)
     * @see #unpopulateAllAttributes()
     */
    public void unpopulateAttribute( String attributeName )
    throws java.lang.IllegalArgumentException,
           javax.oss.UnsupportedAttributeException;
```

Bug 4970356: Wrong exceptions in interface javax.oss.AttributeAccess

3.5 Add Attribute Name to UnsupportedAttributeException message

The UnsupportedAttributeException Exception implements a constructor with the specified detail message.

The constructor that includes the attribute name and message build the message string using explicitly the unsupported attribute name.

The default value for the unsupported attribute name is the string “UNKNOWN_ATTRIBUTE_NAME”.

```
package javax.oss;

/**
```

```
 * This exception is thrown for operations affecting
 * unsupported attributes. It is unchecked.
 *
 * @ossj:exception
 */

public class UnsupportedAttributeException extends RuntimeException
{
    private String  attributeName = "UNKNOWN_ATTRIBUTE_NAME";

    /**
     * Returns the unsupported attribute name that caused the exception.
     * @return the unsupported attribute name as string.
     */
    public  String getAttributeName()
    {
       return attributeName;
    }

    /**
     * Constructs an instance of UnsupportedAttributeException with no message.
     */
    public UnsupportedAttributeException ()
    {
       super();
    }

    /**
     * Constructs an instance of UnsupportedAttributeException with the specified detail
message.
     * A detail message is an instance of String that describes this particular exception.
     * @param message a string representation of the additional detailed message
     */
    public UnsupportedAttributeException (String message)
    {
       super(message);
    }

    /**
     * Constructs an instance of UnsupportedAttributeException with the specified
     * unsupported attribute name and additional detail message.
     * A detail message is an instance of String that describes this particular exception.
     * @param unsupportedAttributeName a string representation of the unsupported
attribute name
     * @param message a string representation of the additional detailed message
     */
    public UnsupportedAttributeException ( String unsupportedAttributeName,
                                           String message )
    {
        //Bug Id 5037431
         //super(message);
        super("Unsupported Attribute Name ["+unsupportedAttributeName+"],"+message);
       attributeName = unsupportedAttributeName;
    }
}
```

Rfe 5037431: UnsupportedAttributeException should mention attribute in toString

3.6 Improve Documentation around QueryValue Pattern

The Specification document and the Javadoc include examples detailing how the named query pattern can be used to implement an implementation-specific query such as “JDO query value”.

3.7 Improve equals() by adding the check of the object type

The Specification document shall explicitly mentioned, for equals method of ManagedAntityValue and QueryValue, that the type of object shall also be equal.

In chapter 3.4.2:

Two managed entity values are equals:

- If their managed entity keys are equal
- If they contain the same populated attribute names
- If each populated attribute value are equal
- If they are of the same type
  following a rule like for example:
  ```
  if((obj == null) || (obj.getClass() !=
  this.getClass())) return false;
  ```

In chapter 3.12.1

Two query values are equals:

- if they contain the same populated attribute names
- if each populated attribute value are equal
- If they are of the same type
  following a rule like for example:
  ```
  if((obj == null) || (obj.getClass() !=
  this.getClass())) return false;
  ```

Rfe 4815764: Java Value Type equal

3.8 Common API compliant to DG 1.2

Eou 4753620: Apply OSS/J Design Guidelines v1.2 to the OSS Common API

The purpose of this change is to allow the OSS Common API to follow the OSSJ Design Guidelines (DG) version 1.2.

3.9 Consistent Exception Handling

The common design guidelines currently suggest reusing exceptions that are already defined in Java as much as possible. This dual meaning has lead to confusion regarding the meaning of the thrown exception, particularly when it comes to using existing EJB exceptions for operations on JVT operations (e.g. javax.ejb.CreateException, javax.ejb.DuplicateKeyException, javax.ejb.FinderException, javax.ejb.ObjectNotFoundException, javax.ejb.RemoveException).

The OSS common API package shall be independent and self-contained. This implies the definition for each needed exception.

3.10 Relocation of the (JSR 130 IPB) *activity* package

In order to share the activity concept with other OSSJ JSRs, the classes in the *activity* package from the JSR 130 specification shall move to the JSR 144. The location of the *activity* package is *javax.oss.cfi*. The following classes are impacted:

javax.oss.cfi.activity.ActivityCapability
javax.oss.cfi.activity.ActivityControlException
javax.oss.cfi.activity.ActivityControlParams
javax.oss.cfi.activity.ActivityController
javax.oss.cfi.activity.ActivityCreationEvent
javax.oss.cfi.activity.ActivityCreationEventPropertyDescriptor
javax.oss.cfi.activity.ActivityEvent
javax.oss.cfi.activity.ActivityEventPropertyDescriptor
javax.oss.cfi.activity.ActivityExecParams
javax.oss.cfi.activity.ActivityKey
javax.oss.cfi.activity.ActivityKeyResult
javax.oss.cfi.activity.ActivityKeyResultIterator
javax.oss.cfi.activity.ActivityPrimaryKey
javax.oss.cfi.activity.ActivityRemovalEvent
javax.oss.cfi.activity.ActivityRemovalEventPropertyDescriptor
javax.oss.cfi.activity.ActivityReportAvailableEvent
javax.oss.cfi.activity.ActivityReportAvailableEventPropertyDescriptor
javax.oss.cfi.activity.ActivityReportDataEvent
javax.oss.cfi.activity.ActivityReportDataEventPropertyDescriptor
javax.oss.cfi.activity.ActivityReportParams
javax.oss.cfi.activity.ActivityResumeEvent
javax.oss.cfi.activity.ActivityResumeEventPropertyDescriptor
javax.oss.cfi.activity.ActivityState
javax.oss.cfi.activity.ActivitySuspendEvent
javax.oss.cfi.activity.ActivitySuspendEventPropertyDescriptor
javax.oss.cfi.activity.ActivityValue
javax.oss.cfi.activity.ActivityValueIterator

javax.oss.cfi.activity.AttributeDescriptor
javax.oss.cfi.activity.DailyScheduleInfo
javax.oss.cfi.activity.QueryActivityReportData
javax.oss.cfi.activity.QueryActivityValue
javax.oss.cfi.activity.RecordDescriptor
javax.oss.cfi.activity.ReportFormat
javax.oss.cfi.activity.ReportInfo
javax.oss.cfi.activity.ReportInfoIterator
javax.oss.cfi.activity.ReportIterator
javax.oss.cfi.activity.ReportMode
javax.oss.cfi.activity.ReportRecord
javax.oss.cfi.activity.Schedule
javax.oss.cfi.activity.SubscriptionFilter
javax.oss.cfi.activity.SubscriptionParams
javax.oss.cfi.activity.WeeklyScheduleInfo

3.11 Local JVT Session

For co-resident component collaboration, the J2EE specification has the concept of Local Session Beans. The OSS Common API is updated to define a local session object name JVTLocalSession for that purpose. This object is identical to the JVTSession, but extends the javax.ejb.EJBLocalObject.

```
package javax.oss;

/**
 * Every JVT&lt;ApplicationType&gt;Session Bean interface must derive from
 * the following base session interface. The base JVTLocalSession interface
 * provides a common set of operations shared by all the JVT Session Beans
 * for local interaction.
 * The implementation of the <CODE>JVTLocalSession</CODE> operations are mandatory.
 *
 * @ossj:jvtlocalsession
 */
public interface JVTLocalSession extends javax.ejb.EJBLocalObject
{


    /**
     * Get the names of the optional operations supported by this JVT
     * Session Bean. The names of the optional operations are defined
     * in the <CODE>JVT&lt;ApplicationType&gt;SessionOptionalOps</CODE> interface as
     * defined by the API.
     *
     *
     * @return String array which contains the names of all the optional
     * operations supported by an implementation of a JVT Session Bean.
     *
     * @exception java.rmi.RemoteException
     */
    String[] getSupportedOptionalOperations( )
    throws java.rmi.RemoteException;

    /**
     * Get the Managed Entity types supported by a JVT Session Bean.
     *
     * @return String array which contains the fully qualified names of the leaf
     * node interfaces representing the supported managed entity types.
     * Note that it is not the implementation class name that is returned.
     * @exception java.rmi.RemoteException
     */

    String[] getManagedEntityTypes()
    throws java.rmi.RemoteException;

    /**
     * Get the Query type names supported by a JVT Session Bean
     *
     * @return String array which contains the fully qualified names of the leaf
     * node interfaces representing the supported query value types,
     * i.e., interfaces which extend QueryValue.
     * @exception java.rmi.RemoteException
     */

    String[] getQueryTypes()
    throws java.rmi.RemoteException;

    /**
     * Get the Event Type names supported by the JVT Session Bean
     *
     * @return String array which contains the fully qualified names of the
     * leaf node interfaces representing the supported event types.
     * @exception java.rmi.RemoteException
     */
    //Get the Event Type Names supported by the Session Component
    String[] getEventTypes()
    throws java.rmi.RemoteException;

     /**
      * Create a QueryValue Instance matching a Query type name.
      * The Session Bean is used as a factory for the creation of
      * query values.
      *
      * @param type fully qualified name of the leaf node QueryValue interface.
      * @return query value object of the specified type.
      * @exception javax.oss.IllegalArgumentException unknown or unsupported
      * query type.
      * @exception java.rmi.RemoteException
      */

    QueryValue makeQueryValue(String type)
    throws javax.oss.IllegalArgumentException ,java.rmi.RemoteException;

    /**
     * Get the EventPropertyDescriptor associated with an event type name.
     *
     * @param eventType fully qualified name of the leaf node Event interface.
     * @return EventPropertyDescriptor which can be used to discover the
     * filterable properties of the specified event type.
     * @exception javax.oss.IllegalArgumentException unknown or unsupported
     * event type.
     * @exception java.rmi.RemoteException
     */

    EventPropertyDescriptor getEventDescriptor(String eventType)
    throws javax.oss.IllegalArgumentException , java.rmi.RemoteException;



    /**
     * Create a Value Type object for a specific Managed Entity type.
     * Not to be confused with the creation of a Managed Entity.
     * The Session Bean is used as a factory for the creation of
     * Value Types.
     *
     * @param valueType fully qualified name of the leaf managed entity value
     * interface.
     * @return managed entity value object of the specified type.
     * @exception javax.oss.IllegalArgumentException unknown or unsupported
     * managed entity value type.
     * @exception java.rmi.RemoteException
     */

    ManagedEntityValue makeManagedEntityValue( String valueType)
    throws javax.oss.IllegalArgumentException, java.rmi.RemoteException;

    /**
     * Query multiple Managed Entities using a QueryValue.
     *
     * @param query a QueryValue object representing the query.
     * @param attrNames names of attributes which should be populated in the
     * ManagedEntityValue objects returned by the ManagedEntityValueIterator.
     * @return a ManagedEnityValueIterator used to extract the results of the query.
     * @exception javax.oss.IllegalArgumentException unsupported query value type.
     * (Note: if an attribute specified in the attributeName array is not
     * supported, it is simply ignored and and does not result in an exception.)
     * @exception java.rmi.RemoteException
     */

    ManagedEntityValueIterator
    queryManagedEntities( QueryValue query , String[] attributeNames)
    throws javax.oss.IllegalArgumentException ,java.rmi.RemoteException;

}
```

3.12 Improve Weakly Typed Arguments

Replace static final constants with J2SE 1.5 enums. Update the design guidelines and Interfaces definitions accordingly. Update all OSS/J APIs in the following releases.

```
public enum OrderState { OPEN, OPEN.NOT_RUNNING, OPEN.NOT_RUNNING.NOT_STARTED,
OPEN.NOT_RUNNING.SUSPENDED, RUNNING, CLOSED, COMPLETED, ABORTED, ABORTED_BYCLIENT,
ABORTED_BYSERVER };
```