Change Log for OSS Common API version 1.3

OSS through Java™ Initiative

Vincent Perrot, Sun Microsystems, Inc.

COM-API-SPEC_change_log.1.3.11.doc

Executive Summary

This document summarizes the changes to the OSS Common API (JSR 144) specification Version 1.3. The main purpose of this version is

- Update and fix issues related to the Core Business Entities (CBE),
- Include feedback from other OSS/J API extending JSR 144

However, since maintenance release to the specification was taking place, additional modifications to the previously existing Java Value Type interface were also incorporated. All these modifications are coming from the Web Bug tracking system at:

- https://jsr144-public.dev.java.net
- https://jsr144-private.dev.java.net (dedicated to OSS/J Members)

There are two lists of changes:

- "proposed" changes are those modifications that are included in OSS Common API version 1.2.
- "deferred" changes are those modifications that are not included in OSS Common API version 1.2, 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 v1.3 version of the OSS Common API, JSR 144.

The changes have been collected through:

- Java.net Issue Tracker: 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_common-1_3-mr-spec.zip: contains the Version 1.3 of the OSS common API, JSR 144, http://java.sun.com/products/oss/start_download.html

The backward compatibility may not be guarantied. The deprecation mechanism of java will be applied every time possible.

“*Deprecated* means this method is still usable, but you should not use it. It will gradually be phased out. There is a new method to do the same thing. Deprecated methods are marked with a special JavaDoc comment.”

See addition information about java deprecation at:

http://java.sun.com/j2se/1.3/docs/guide/misc/deprecation/deprecation.html

1.5 Revision History

Date	Version	Author	State	Comments
Nov 2005	1.3.1	Vincent Perrot, Sun Microsystems	Initial Draft	• Add change request from issue tracker
Dec 7th 2005	1.3.3	Vincent Perrot, Sun Microsystems	Spec leader Revie Draft	• Add the latest CR from issue tracker
Dec 13th 2005	1.3.4	Vincent Perrot, Sun Microsystems	Draft review for approval	• For final OSSJ approval
Dec 16th 2005	1.3.5	Vincent Perrot, Sun Microsystems	Submitted for MREL	• Add some issues
Jan 9th 2006	1.3.7-9	Vincent Perrot, Sun Microsystems	Spec Leaders Review	• Add some issue in list •
Jan 9th 2006	1.3.11	Vincent Perrot, Sun Microsystems	Reviewed	•

2 Summary of changes

3 Accepted changes

Details on the fix implementations are available in Issue tracker.

3.1 Public domain

3.1.1 Issue #2, add troubleDescription to TroubleTicketValue

https://jsr144-public.dev.java.net/issues/show_bug.cgi?id=2

Please add an attribute 'troubleDescription' (getter, setter, constant) to javax.oss.cbe.trouble.TroubleTicketValue

This would be the javadoc for that attribute:

```
/* The trouble description is a plain text description of the occurred trouble.
Confinement: So troubleDescription is not a description of the trouble ticket
(or BusinessInteraction), but of the trouble which the ticket deals with.
Example: "DSL is slow for our customers in Chelsea."
*/
```

Note: introducing "Description" attribute at the BusinessInteraction level have been evaluated. There is currently no request for this, so only the troubleDescription will be added to the TroubleTicketValue definition.

3.1.2 Issue #3, ... role assignments for trouble ticket

https://jsr144-public.dev.java.net/issues/show_bug.cgi?id=3

We would like to add an attribute to javax.oss.cbe.trouble.TroubleTicketValue that allows clients to attach a list of role assignments to a trouble ticket. A role assignment is a combination of role name and the person that has that role for that ticket.

We would also like to avoid too complex structures, e.g. involving associations or other independently managed entities.

So our change request is (please review and check if this makes sense):

Please add a datatype javax.oss.cbe.trouble.TroubleTicketRoleAssignment with two attributes

- roleName (String)

- assignedParty (PartyValue)

So this datatype has to have these methods as well:

- String[] getSupportedPartyTypes()

- PartyValue makePartyValue(String partyType)

Please also add an attribute roleAssignments javax.oss.cbe.trouble.TroubleTicketRoleAssignment[] to javax.oss.cbe.trouble.TroubleTicketValue

So there needs to be a factory method as well:

- TroubleTicketRoleAssignment makeTroubleTicketRoleAssignment()

Some Javadocs for the new definitions:

TroubleTicketRoleAssignment: "An assignment of a party to a role that it has during the trouble resolution process."

roleName: "Name of the role. Can be freely defined. Please also see TroubleTicketRole."

assignedParty: "Party that is assigned that role."

roleAssignments: "List of role assignments for the trouble resolution process."

Note: as only array of roleAssignment are used in TroubleTiocketValue, the factory method will be:

```
        /**
         * Returns the list of empty role assignments
         *
         * @param howMany the number of element in the array
         * @return the new array of role assignements
         */
        public TroubleTicketRoleAssignment[] makeTroubleTicketRoleAssignments(int
howMany);
```

3.1.3 Issue #4, BusinessInteraction shall include a BIItemkeys attribute management

https://jsr144-public.dev.java.net/issues/show_bug.cgi?id=4

In SID there is an association 1-n between BI and BIItem.

This association shall be implemented a and Item[] attribute management in the BusinessInteractionvalue definition as is done in javax/oss/cbe/sla/ServiceLevelAgreementValue

```
        /**
         * Create a array of BusinessInteractionItemKey identifying instance of the 
Business Interaction Items.
         * @param n number of element in the array
         * @return The array of new instance of keys
         */
        public BusinessInteractionItemKey[] makeBusinessInteractionItemKeys(int n);

        /**
         * Gets the list of keys for all the Business Interaction Items composing
         * this Business Interaction.
         * 
         * 
         * @return The keys to the Business Interaction Items composing this
         *         Business Interaction.
         * @exception IllegalStateException -
         *                if the Attribute is not populated.
         */
        public BusinessInteractionItemKey[] getBusinessInteractionItemKeys() throws 
IllegalStateException;

        /**
         * Sets the list of Business Interaction Items to be composed by this
         * Business Interaction.
         * 
         * 
         * @param pKeys
         *            The keys of the Business Interaction Items to be used to
         *            define this Business Interaction.
         * @exception IllegalArgumentException -
         *                if the argument is invalid.
         */
        public void setBusinessInteractionItemKeys(BusinessInteractionItemKey[] pKeys)
                       throws IllegalArgumentException;
```

All interfaces that inherit from this one shall also includes the same kind of itemKeys management.

For example, trouble ticket package:

```
interface TroubleTicketValue {

        String TROUBLE_TICKET_ITEM_KEYS = "troubleTicketItems";

        TroubleTicketItemKey[] getTroubleTicketItems();

        void setTroubleTicketItems(TroubleTicketItemKey[] items);

}



Example for a TroubleTicketItemValue type to be used:

interface ResourceTroubleRicketItemValue extends TroubleTicketItemValue {

   String RESOURCE_KEY = "resourceKey";

   ResourceKey getResourceKey();
   void setResourceKey(ResourceKey  key);

   ResourceKey makeResourceKey();

}
```

Note: Using this basic definition will need to add method definitions in the functional API definitions to handle the Values and the ValueItems.

AI Axel: Synchronize with Thierry Supplisson (Spec lead of SQM API) to define a consistent way to handle Value and ValueItem pair.

In BusinessIteractionValue and all extensions value (AgreementValue, ServiceLevelAgreementValue, TroubleTicketValue) the management and factory method have been added to handle xxxxItemKeys.

3.1.4 Issue #5, use java 1.4 for compiling specs

https://jsr144-public.dev.java.net/issues/show_bug.cgi?id=5

use java 1.4 for compiling specs

Common 1.2 specs are compiled with Java 1.5, but OSS/J is still comitted to using Java 1.4.

This leads to difficulties when compiling 1.4 stuff (TT API spec, TT RI, ...).

Note: will not fixed, but a build script have been provided to allow JVM 1.4 recompilation when needed. Visit http://jsr144-public.dev.java.net

3.2 Private domain

3.2.1 Issue #6, Remove obsolete javax.oss.cbe.alarm.AlarmConfig definition

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=6

AlarmConfig -- as far as I can tell, this is used only by the SQM / SLA stuff. I'm not sure on what exactly its used for or why we really need it.

It seems to be an abridged version of AlarmValue. IMHO, if we keep it, there needs to be an inheritance relationship established between AlarmValue and AlarmConfig. Two classes containing two views of the same basic information sort of bothers me. I did not update the comments; I don't remember the motivation behind the SQM API need for this puppy.

Note: AlarmConfig interface definition will noted "@deprecated"

3.2.2 Issue #8, State in cbe.datatypes

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=8

in cbe.datatypes:

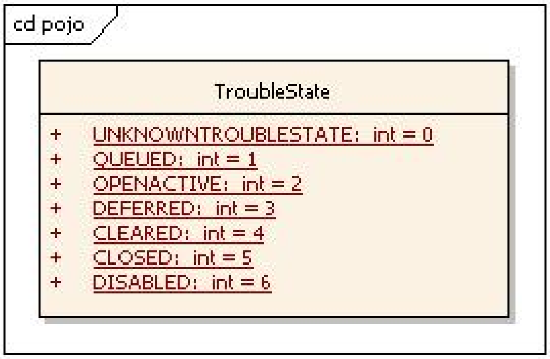
1. add State interface

2. change the LifeCycleStatus to LifeCycleState and extend it from the State

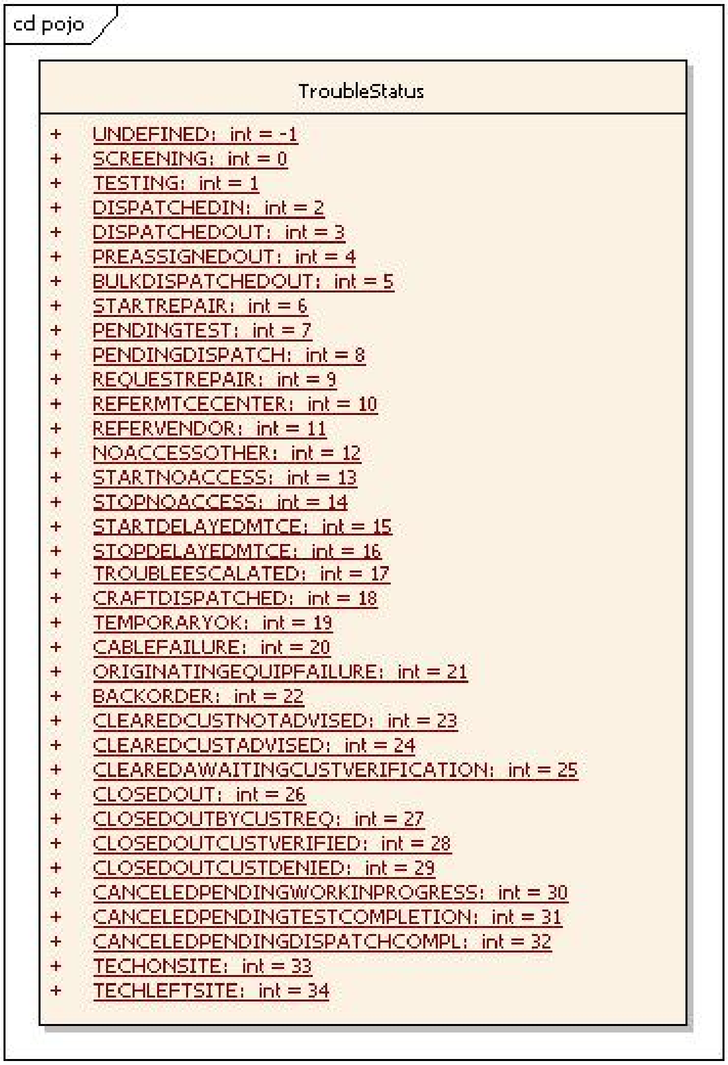
State is one of the core concepts used in all OSS applications. It is beneficial to share the interface definition among all APIs. The proposed State interface support hierarchical nature of the state by using java.lang.String based 'dotted' format x.y.z. instead of a regular enumerated value. State major purpose is to reflect a process/activity execution state.

Note:

state and status are not related entities. we will use state for managing e.g. orders and e.g. tt need statuses.



here still with integer ...



```
+ UNDEFINED: int = -1
+ SCREENING: int = 0
+ TESTING: int = 1
+ DISPATCHEDIN: int = 2
+ DISPATCHEDOUT: int = 3
+ PREASSIGNEDOUT: int = 4
+ BULKDISPATCHEDOUT: int = 5
+ STARTREPAIR: int = 6
+ PENDINGTEST: int = 7
+ PENDINGDISPATCH: int = 8
+ REQUESTREPAIR: int = 9
+ REFERMTCECENTER: int = 10
+ REFERVENDOR: int = 11
+ NOACCESSOTHER: int = 12
+ STARTNOACCESS: int = 13
+ STOPNOACCESS: int = 14
+ STARTDELAYEDMTCE: int = 15
+ STOPDELAYEDMTCE: int = 16
+ TROUBLEESCALATED: int = 17
+ CRAFTDISPATCHED: int = 18
+ TEMPORARYOK: int = 19
+ CABLEFAILURE: int = 20
+ ORIGINATINGEQUIPFAILURE: int = 21
+ BACKORDER: int = 22
+ CLEAREDCUSTNOTADVISED: int = 23
+ CLEAREDCUSTADVISED: int = 24
+ CLEAREDAWAITINGCUSTVERIFICATION: int = 25
+ CLOSEDOUT: int = 26
+ CLOSEDOUTBYCUSTREQ: int = 27
+ CLOSEDOUTCUSTVERIFIED: int = 28
+ CLOSEDOUTCUSTDENIED: int = 29
+ CANCELEDPENDINGWORKINPROGRESS: int = 30
+ CANCELEDPENDINGTESTCOMPLETION: int = 31
+ CANCELEDPENDINGDISPATCHCOMPL: int = 32
+ TECHONSITE: int = 33
+ TECHLEFTSITE: int = 34
```

states need hierarchy (string) while statuses are flat (no hierarchy) and can be managed by integers

purpose of string operations in integer status is to provide more user friendly interface and it applies to status only (not state).

yes, the string operations can be dropped or move to util.

Note: State and Status interface definitions need to be sent for review.

Axel volunteer to review them before the of the week.

AI Vince: send Axel the proposel State and status definitions and an improved the proposal.

Impacts

TroubleTicketState extends State

AlarmAckState deprecated replaced by AlarmAckStatus

PerformanceMonitorState change definition to

```
/*
Copyright 2002-2006 The Members of the OSS through Java(TM) Initiative.
All rights reserved. Use is subject to license terms.
*/
package javax.oss.cbe.measurement;


/**
 * Defines the state constants for performance monitor.
 *
 * <p>
 * The measurement job can have several states: Active and on duty, Active but of duty and
 * suspended.
 *
 *@see PerformanceMonitorValue
 *
 * @author OSS through Java Initiative, Vincent Perrot Sun Microsystems Inc.
 * @version 1.3
 * @since March 2005
 *
 */
public interface PerformanceMonitorState extends javax.oss.cbe.datatypes.State
{
    /**
     * Constant for a measurement job that is active and on duty.
     * @deprecated replaced by ACTIVE.ONDUTY
     */
    public static final int ACTIVE_ON_DUTY = 1;

    /**
     * Constant for a measurement job that is active and off duty.
     * @deprecated replaced by ACTIVE.OFFDUTY
     */
    public static final int ACTIVE_OFF_DUTY = 2;

    /**
     * Constant for a measurement job that is suspended.
     * @deprecated replaced by SUSPENDED
     */
    //public static final int SUSPENDED = 3;

    /**
     * Constant for a measurement job that is active.
     */
    public static final java.lang.String ACTIVE = "active";
    /**
     * Constant for a measurement job that is active and on duty.
     */
    public static final java.lang.String ONDUTY = "onduty";
    public static final java.lang.String ACTIVEONDUTY =
ACTIVE+javax.oss.cbe.datatypes.State.SEPARATOR+ONDUTY;
    /**
     * Constant for a measurement job that is active and off duty.
     */
    public static final java.lang.String OFFDUTY = "offduty";
    public static final java.lang.String ACTIVEOFFDUTY =
ACTIVE+javax.oss.cbe.datatypes.State.SEPARATOR+OFFDUTY;
    /**
     * Constant for a measurement job that is suspended.
     */
    public static final java.lang.String SUSPENDED = "suspended";

}
```

*ProductStatus, ServiceStatus, ResourceStatus: will be deprecated, then be renamed using "*State" extend datatypes.LifeCycleState definition.*

3.2.3 Issue #9, Status in cbe.datatypes

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=9

in cbe.datatypes:

1. add Status interface

optional:

a) create LifeCycleStatus interface, extending Status

Status is independent of the State and is used to reflect on the business status of a process/activity. Implementation wise a plain enumerated value is sufficient.

The state of the activity may change through the execution while the status is expected to remain the same. Example: a TT Status is ESCALATED and the process handling the TT may go from "open.notrunning" through "open.running.xxx" to "closed.completed"

Note: same as Issue #8

The optional section related to LifeCycleStatus" of this issue will not be implemented as the LifeCycleStatus definition have been deprecated after the implementation of the fix regarding Issue #8.

And agree on the statement "Implementation wise a plain enumarated value is sufficient."

3.2.4 Issue #10, RequestValue in cbe.bi

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=10

to align CBEs with SID introduce RequestValue, orders are Requests extension.

I am not sure the purpose of Request in SID, in ossj it could be used to break up businessInteractions into internal requests/activities.

proposed changes:

in cbe.bi:

1. add RequestValue interface, extending cbe.bi.CustomerBusinessInteractionValue

2. add RequestState, extending cbe.datatypes.State

3. replace State enum attribute in BI with cbe.datatypes.State

optional:

a) add RequestStatusValue, extending cbe.datatypes.Status

b) replace Status enum attribute in BI with cbe.datatypes.Status

Note: this new CBE definitions will need improved definitions.

The Javadoc shall at least allow the implementor to clearly make the distinction between the BusinessInteraction/CustomerBusinessInteraction and Request definition and usage. The SID definitions will be the starting point, but additional documentation (like example) may be requested.

AI OM expert Group: provide this informations to the JSR 144 Spec leader.

Done: the RequestValue, RequestKey and BusinessInteractionState have been added. The BusinessInteractionValue deprecate the interactionStatus attribute and replaced by interactionState (see also Issue #13). The Description attribute have also been added to the BusinessInteractionValue.

3.2.5 Issue #13, update all cbe to align with State, Status additions

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=13

Review all cbe entities and align them with the newly introduced State and Status where necessary.

Note: See issue #8 and #9

3.2.6 Issue #14, Improve ReportMode content by adding report mode from JSR 130 activity package

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=14

Improve ReportMode content by adding report mode from JSR 130 activity package

The JSR 130 activity package includes the ReportMode interface definition.

As the report package is already part of the common CBE the JSR 130 will have to use instead of redefining it.

But they are some definition and information that need to increment the current v 1.2 CBE Definition.

The comparison shall also apply for the ReportFormat, ReportData, ReportInfo, ReportInfoIterator interface definitions.

3.2.7 Issue #15, Improve Report package interface definitions

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=15

Improve Report package interface definitions

The report package needs to be improved to better follow the DG:

- complete javadoc remove measurement related comments)

- add static definition for attributes

- add get AND set when needed or update the comment to ask the constructor with arguments.

- Issue #14...

3.2.8 Issue #17, Array of Date in common schema is not accurate

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=17

Array of Date in common schema is not accurate

I've found these items in http://ossj.org/xml/Common/v1-2/OSSJ-Common-v1-2.xsd.

The first one here has the type of byte for the items in an array of Date. From the DG I think the mapping should be from Java Date type to dateTime.

```
<complexType name="ArrayOfDate">
           <sequence>
<element name="item" type="byte" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</complexType>
```

This one has dateTime items for Calendar array. I can see that items of java type Calendar have to become of type dateTime in XML, so should the name of the array change in line with that? I think that it will make sense for the XML to be self-consistent, since it may be used in situations where Java is not part of the picture.

If indeed the name is changed, then the same is true for ArrayOfDate => ArrayOfDateTime and there is no need for both of these.

```
<complexType name="ArrayOfCalendar">

            <sequence>

<element name="item" type="dateTime" nillable="true" minOccurs="0" maxOccurs="unbounded"/>

</sequence>

</complexType>
```

Note: The array definition for Calendar is removed.

The following array is added:

```
       <!--============date array=============-->
       <complexType name="ArrayOfDateTime">
              <sequence>
                     <element name="item" type="dateTime" nillable="true" minOccurs="0"
maxOccurs="unbounded"/>
              </sequence>
       </complexType>
```

The doclet.properties shall add the following mapping when array of calendar is declared:

```
[Ljava.util.Calendar;.type=co-v1-3:ArrayOfDateTime
```

The Schema has also been improved to remove the unnecessary arrays of exceptions.

3.2.9 Issue #18, added isWithin to the TimePeriod

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=18

add operation to javax.oss.cbe.datatypes.TimePeriod

```
/*
   * true if current Time within the TimePeriod
   * @return boolean
   * @exception java.lang.IllegalStateException
   */
  public boolean isWithin(java.util.Date currentDate)
```

```
    throws java.lang.IllegalStateException;
```

VP: done

3.2.10 Issue #19, Version data type

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=19

add Version data type in javax.oss.cbe.datatypes. Version is required in a number of entities and processes. As minimum, Integer version is supported yet it would be beneficial to support M.m.P.B notation as well (explained below).

```
package javax.oss.cbe.datatypes;
import java.util.Date;
/**
 * Version type
 * @version 0.1
 * @created 09-Dec-2005 6:32:11 AM
 * @since December 2005
 * @ossj:complexdata
 * @author Artur Uzieblo CommsArt Technologies
 *
 * attributes
 * -------------------------------
 * versionType          String
 * versionNumber      Integer
 * versionDate          Date
 * versionDescription String
 * versionValidFor    TimePeriod
 *
 * potentially (Ma.Mi.P.B) :
 * versionMajor          Integer
 * versionMinor          Integer
 * versionPatch          Integer
 * versionBuild          Integer
 * // return "Ma.Mi.P.B"
 * public String getVersionMMPB();
 * // return sub-versions
 * public Integer getVersionMajor();
 * public Integer getVersionMinor();
 * public Integer getVersionPatch();
 * public Integer getVersionBuild();
 * // set "Ma.Mi.P.B"
 * public boolean setVersionMMPB(String);
 * // set sub-versions
 * public void setVersionMajor(Integer);
 * public void setVersionMinor(Integer);
 * public void setVersionPatch(Integer);
 * public void setVersionBuild(Integer);
 *
 */
public interface Version {

  // returns true if the current date falls within the ValidFor and/or other
conditions. Implementation may 'build' in margins on both sides (start, end),
e.g. if the order takes three days to complete, it may not allow new orders 4
days before the version expiry date.
  public boolean isValid();

  // return versionNumber
  public int getVersion();
  // versionNumber
  public void setVersion();
  // return Type
  public String getType();
  // Type
  public void setType(String type);

  // return Date
  public java.util.Date getDate();
  // Date
  public boolean setDate(java.util.Date date);

  // return description
  public String getDescription();
  // description
  public void setDescription(String description);

  // return TimePeriod
  public javax.oss.cbe.datatypes.TimePeriod getValidFor();
  // TimePeriod
  public void setValidFor(javax.oss.cbe.datatypes.TimePeriod timeperiod);

}
```

Note: This definition is too stick with a model. The basic Version definition shall be model agnostic as possible. The new proposed interface definition will extend Comparable and define only the description and validFor attributes.

The proposed interface is:

```
/*
 Copyright 2002-2006 The Members of the OSS through Java(TM) Initiative.
 All rights reserved. Use is subject to license terms.
 */
package javax.oss.cbe.datatypes;

/**
 * Version type
 *
 * @author OSS through Java Initiative, Artur Uzieblo CommsArt Technologies, Vincent
Perrot Sun Microsystems
 * @version 1.3
 * @since Jan 2006
 * @ossj:complexdata
 *
 */
public interface Version extends java.lang.Comparable {

        /**
         * Returns a textual description of this Version
         *
         *
         * @return A textual description of the version.
         */
        public String getDescription();

        /**
         * Sets the textual description of this Version
         *
         * @param description - A textual description of the version
         */
        public void setDescription(String description);

        /**
         * Builds and returns the default timePeriod for this place
         *
         * @return a new instance of TimePeriod
         */
        public TimePeriod makeTimePeriod();

        /**
         * Returns the time period that the place is valid for
         *
         * @return time period
         *
         * @throws java.lang.IllegalStateException when the attribute can not be returned
         */
        public javax.oss.cbe.datatypes.TimePeriod getValidFor();

        /**
         * Sets the time period that the place is valid for
         *
         * @param validFor time period
         *
         * @throws java.lang.IllegalArgumentException if the attribute is null or invalid
         */
        public void setValidFor(javax.oss.cbe.datatypes.TimePeriod validFor)
                       throws java.lang.IllegalArgumentException;

}
```

VP: done

3.2.11 Issue #20, improve generic features

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=20

This functionality would ease the creation of generic oss/j clients:

JVT Session

1) add functionality to explore the return types of named queries and named updates (maybe like event descriptors)

Attribute Access

1) add a method to request the data type of an attribute

2) add a method to request the valid values for enumeration based attributes

3) add a generic factory (make method) for all data types

Data types

1) add an interface almost like AttributeAccess that allows exploring the sub attributes of complex attributes

So a generic client could understand the structure of Managed Entity Values down into the complex attribute types.

Note: This topic needs some concrete proposal in order to progress.

“Utilities” methods have been added to the base AttributeAccess definition.

3.2.12 Issue #22, [RI & Spec] RI Managed Entity Value should support NULL values.

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=22

The Common Specification (User's Guide) Chapter 3.1.1.4 Null Value for return and input parameters:

"All values that are returned or are an input parameter are not normally expected to be "null". This is even if an array is empty. It must be explicitly stated in the description operation if "null" is allowed."

shall be updated using the material provided below from the Design Guidelines:

“The <ManagedEntity>Value Java interfaces must provide public static final string field definition for the name of each attribute it support. For example if the managed entity support an attribute called CustomerID a string CUSTOMER_ID ="customerID" must be defined.

The name of the attribute should start with a lower case.   This attribute name constant should be used to fill the array of requested attribute names in the JVT Beans. It is also used when calling the Managed Entity Value getAttribute(String name) method. The same constants are used to identify the populated attributes of a value object. The implementation of a Managed Entity Value interface should add the name of the attribute to the list of populated attributes each time an attribute is set on a value type. The getPopulatedAttributeNames() can then be used to retrieve the names of all the attributes for which a value was provided.

In order to provide a null value for an attribute it is mandatory to explicitly call the setXXX(...) method for this attribute with a null value. An attribute "XXX" is null if both the getAttributeValue("XXX") returns a null object and if the isPopulated("XXX") returns true. A value of false for isPopulated (...) implies that no value is provided for that attribute.

Note: Axel agrees. This confirms that the RI common 1.3 will be “regenerated” with the support of the null value for attributes.

The specification document has been updated in this sense and the RI supports now null attribute values.

3.2.13 Issue #26, Use of isXXX() for boolean is inconsistent

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=26

In some cases boolean attributes are accessed using isXX() and in other cases using getXXX().

eg : ServiceValue uses getMandatory()

WeeklySchedule uses isActive()

See also issue #33.

It impacts mainly the "getMandatory" method in ServiceValue.

3.2.14 Issue #27, Method exceptions are inconsistent

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=27

The exceptions thrown by many getters and setters are incorrect with regard to the Design Guidelines - example AlarmValue : getAcknowledgementInteractionRecord does not throw any exceptions.

See also issue #33.

Ok.

3.2.15 Issue #28, KeyREsult files include setXXXKey methods

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=28

AssociationKeyResult, EntityKeyResult & EntitySpecificationKeyResult have setXXKey methods in the spec.

These should not be included

See also issue #33.

Ok.

3.2.16 Issue #29, CBEManagedEntityValue has missing methods

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=29

CBEManagedEntityValue has no get/set methods for its associated Key.

See also issue #33

Ok.

3.2.17 Issue #30, DG is unclear about inclusion of Constants

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=30

The list of constants for the attribute declarations is included in all of the Value files in the CBE, but there are a few included in *some* of the datatype files as well. eg location/FormattedAddress has things like:

public static final String ADDR_LN1 = “addrLn1”

The Specification should state the precise rules.

See also Issue #33.

Ok.

3.2.18 Issue #31, In ServiceValue change the getMandatory to isMandatory

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=31

In order to follow the recommended java rule regarding how to managed boolean attribute, the method getMandatory shall be deprecated to be replaced by the isMandatory().

Removed: redundant with Issue #26

3.2.19 Issue #32, In AlarmValue change the att name BackedUpStatus to BackedUp

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=32

In AlarmValue, the boolean "BackedUpStatus" attribute name shall change to "BackedUp" in order to use the is/setBackedUp instead of get/setBackedUpStatus

So isBackedUp makes more sense than isBackedUpStatus.

See also Issue #26.

Ok.

3.2.20 Issue #33, Align exception/factory method/key management to the Common/DG rules

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=33

In the attached XLS sheet you will get all the changes that will be needed to applied to the interface definition in order to get consistent CBE definitions.

The document is available on the jsr144-private project on java.net at:

https://jsr144-private.dev.java.net/files/documents/3868/27031/COM-API-Consistency_change_list.0.3.xls

Ok.

3.2.21 Issue #34, Association Value missing AEndKey and ZEndKey mutators and accessors

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=34

Association Value is missing AEndKey and ZEndKey mutators and accessors.

The following static fields and methods should be added to the AssociationValue interface.

```
…

       public static final String A_END_KEY = "aEndKey";
       public static final String Z_END_KEY = "zEndKey";

 …


    /**
     * Sets the key for the A end of the Association.
     *
     * @param key for the A end
     * @exception java.lang.IllegalArgumentException thrown
     * to report that a bad argument was provided to the method.
     */

       public void setAEndKey( EntityKey key)
       throws java.lang.IllegalArgumentException;

    /**
     * Gets the key for the A end of the Association.
     *
     * @return the A end key
     * @exception java.lang.IllegalStateException in case
     * the key attribute is not populated
     * @ossj:ignore
     */
       public EntityKey getAEndKey()
       throws java.lang.IllegalStateException ;

    /**
     * Sets the key for the Z end of the Association.
     *
     * @param key for the Z end
     * @exception java.lang.IllegalArgumentException thrown
     * to report that a bad argument was provided to the method.
     */
       public void setZEndKey( EntityKey key)
       throws java.lang.IllegalArgumentException;

    /**
     * Gets the key for the Z end of the Association.
     *
     * @return the Z end key
     * @exception java.lang.IllegalStateException in case
     * the key attribute is not populated
     * @ossj:ignore
     */
```

```
        public EntityKey getZEndKey()
        throws java.lang.IllegalStateException ;
```

Ok.

3.2.22 Issue #35, Fix chapter 3.1.1.1

https://jsr144-private.dev.java.net/issues/show_bug.cgi?id=35

The last sentence:

“If an attribute is not populated and the corresponding get the **javax.oss.OssIllegalStateException** is raised.”

Shall be

“If an attribute is not populated and the corresponding get the **java.lang.IllegalStateException** is raised.”

Ok.

3.2.23 Remove Deprecated interfaces/Methods

Interface or method definitions noted “deprecated” in the Common v 1.2 version will be removed from this version (1.3) of the Common API.

Following the detailed list (as displayed in the javadoc):

Deprecated Interfaces
javax.oss.util.IRPEvent *CR#6293854*
javax.oss.util.IRPEventPropertyDescriptor *CR#6293854*
javax.oss.QueryValue *replaced by NamedQueryValue*
javax.oss.Serializer *CR#4753620*
javax.oss.SerializerFactory *CR#4753620*
javax.oss.XmlSerializer *CR#4753620*
javax.oss.XmlSerializerEncodingStyles *CR#4753620*

Deprecated Fields

javax.oss.cbe.alarm.PerceivedSeverity.CLEARED

Deprecated Methods

javax.oss.JVTSession.getQueryTypes()
replaced by getNamedQueryTypes

javax.oss.JVTLocalSession.getQueryTypes()
replaced by getNamedQueryTypes

javax.oss.cbe.schedule.WeeklySchedule.isFridayActive()
replaced isActiveDay

javax.oss.cbe.schedule.WeeklySchedule.isMondayActive()
replaced isActiveDay

javax.oss.cbe.schedule.WeeklySchedule.isSaturdayActive()
replaced isActiveDay

javax.oss.cbe.schedule.WeeklySchedule.isSundayActive()
replaced isActiveDay

javax.oss.cbe.schedule.WeeklySchedule.isThursdayActive()
replaced isActiveDay

javax.oss.cbe.schedule.WeeklySchedule.isTuesdayActive()
replaced isActiveDay

javax.oss.cbe.schedule.WeeklySchedule.isWednesdayActive()
replaced isActiveDay

javax.oss.JVTSession.makeQueryValue(String)
replaced by makeNamedQueryValue

javax.oss.JVTLocalSession.makeQueryValue(String)
replaced by makeNamedQueryValue

javax.oss.JVTSession.queryManagedEntities(QueryValue, String[])
replaced by query

javax.oss.JVTLocalSession.queryManagedEntities(QueryValue, String[])
replaced by query

javax.oss.cbe.schedule.WeeklySchedule.setFridayActive(boolean)
replaced setActiveDay

javax.oss.cbe.schedule.WeeklySchedule.setMondayActive(boolean)
replaced setActiveDay

javax.oss.cbe.schedule.WeeklySchedule.setSaturdayActive(boolean)
replaced setActiveDay

javax.oss.cbe.schedule.WeeklySchedule.setSundayActive(boolean) *replaced setActiveDay*
javax.oss.cbe.schedule.WeeklySchedule.setThursdayActive(boolean) *replaced setActiveDay*
javax.oss.cbe.schedule.WeeklySchedule.setTuesdayActive(boolean) *replaced setActiveDay*
javax.oss.cbe.schedule.WeeklySchedule.setWednesdayActive(boolean) *replaced setActiveDay*

Ok.