# Database - Feature #1967

## rework DMO class/interface hierarchy

01/20/2013 11:30 PM - Eric Faulhaber

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 01/18/2013 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Eric Faulhaber | **% Done:** | 100% |
| **Category:** | | **Estimated time:** | 20.00 hours |
| **Target version:** | Conversion Support for Server Features | | |
| **billable:** | No | **vendor_id:** | GCD |

**Description**

### History

#### #1 - 01/21/2013 12:18 AM - Eric Faulhaber

The idea is to support buffer handles more naturally and to support converted, buffer-related methods, attributes, and language statements as direct method calls on DMO proxy objects.

Today, converted business logic references DMOs by their schema interfaces, which include only getter and setter methods for their database fields. This means that any buffer-related 4GL methods and attributes must be invoked as second-class citizens, typically by invoking a static method call and passing the proxy object into that call (e.g., RecordBuffer.recordID(myDMO)).

Going forward, business logic will reference DMOs by an interface which extends both the schema interface as well as a new Buffer interface, which includes APIs for all converted 4GL methods and attributes. That interface will extend WrappedResource, so that instances of these objects can be used within instances of the handle class. This means that business logic can invoke both schema-specific methods, and 4GL-style methods and attributes, as direct method calls on a DMO proxy object.

Each converted DMO interface will include a new, public, static, inner interface which extends both the enclosing interface, as well as the new Buffer interface:

```
public interface SomeDMO
{
   // getter and setter methods
   ...

   public static interface Buf
   extends SomeDMO, Buffer
   {
   }
}
```

This grouping interface does not define any methods of its own; its only purpose is to group the enclosing interface and the Buffer interface, so that DMO proxies in converted business logic can access the methods of both, without requiring any casting.

DMO implementation classes will continue to implement only the schema-specific, DMO interface (i.e., not the one that extends Buffer).

BufferImpl is the concrete implementation of the Buffer interface. It also implements BufferReference, which has one method, buffer, used to access the RecordBuffer instance associated with a DMO proxy object. Internally, BufferImpl uses the buffer method to delegate much of its work to the RecordBuffer class. The implementation of the buffer method in this class throws an exception; at runtime, we rely on a proxy implementation of the buffer method to override this implementation to get us an actual RecordBuffer instance. However, it needs to be present in BufferImpl to satisfy the Java compiler.

This new implementation requires a number of conversion changes in addition to the schema conversion change described above. Everywhere a DMO is defined or declared today, the data type must change from {DMO interface} to {DMO interface}.Buf. This affects all calls to:

- RecordBuffer.define
- TemporaryBuffer.define (overloaded)
- TemporaryBuffer.useShared
- SharedVariableManager.lookupBuffer

In addition, a number of existing constructs must change:

```
RecordBuffer.recordID(dmo)       --> dmo.recordID()
RecordBuffer.rowID(dmo)          --> dmo.rowID()
RecordBuffer.isAvailable(dmo)    --> dmo.available()
RecordBuffer._isAvailable(dmo)   --> dmo._available()
RecordBuffer.wasLocked(dmo)      --> dmo.locked()
RecordBuffer._wasLocked(dmo)     --> dmo._locked()
RecordBuffer.wasAmbiguous(dmo)   --> dmo.ambiguous()
RecordBuffer._wasAmbiguous(dmo)  --> dmo._ambiguous()
RecordBuffer.isNew(dmo)          --> dmo.newlyCreated()
RecordBuffer._isNew(dmo)         --> dmo._newlyCreated()
TemporaryBuffer.clear(dmo)       --> dmo.deleteAll()
```

Note that since there is now the opportunity to encounter naming collisions with DMO-specific method names, we have to ensure that no method defined by the Buffer interface begins with is, get, or set. These prefixes are reserved for DMO-specific interface method names, and in fact, such methods will not use any other prefixes, so as long as we follow this rule, we will avoid collisions.

This change presents some problems for hand-written code. Specifically, there is now a body of hand-written Java code which relies on the old APIs, most notably RecordBuffer.isAvailable/_isAvailable, RecordBuffer.recordID, and a handful of others. For the time being, these methods are marked deprecated so that this code will still compile, though it requires testing to make sure it still works. Ultimately, this code should be edited to comply with the new architecture.

I have gotten very far with this change over the weekend, but I'm still working out issues getting converted Majic to compile. I hope to finish tomorrow and get it into testing very soon.

For those of you currently working on issues that will be affected by this change, please ask any questions you may have.

#### #2 - 01/21/2013 12:19 AM - Eric Faulhaber

*- Status changed from New to WIP*

*- Assignee set to Eric Faulhaber*

#### #3 - 01/21/2013 05:19 AM - Stanislav Lomany

Will buffer parameters change to {DMO interface}.Buf?

```
public class ExternalProc
{
   SomeDMO.Buf buf;

   public void execute(final SomeDMO.Buf buf)
   {
      ...
```

#### #4 - 01/21/2013 10:01 AM - Eric Faulhaber

Stanislav Lomany wrote:

> Will buffer parameters change to {DMO interface}.Buf?

Yes, any access to a DMO from business logic will be via this new interface.

#### #5 - 01/21/2013 05:24 PM - Eric Faulhaber

Updated the Buffer interface to accommodate the following, additional changes:

```
RecordBuffer.create(dmo)   --> dmo.create();
RecordBuffer.delete(dmo)   --> dmo.delete();
RecordBuffer.release(dmo)  --> dmo.release();
RecordBuffer.validate(dmo) --> dmo.validate();
```

In addition, I've added the following at Stanislav's request for trigger support, though currently the runtime backing is only stubbed out:

```
   public void openReadOnlyScope();
```

I am running conversion now, cleaning up, adding javadoc and comments;  hope to be able to go into regression testing tomorrow.

#### #6 - 01/23/2013 05:48 AM - Eric Faulhaber

*- File ecf_upd20130123a.zip added*

Here is a drop of the code which is about to undergo regression testing. Note that additional, customer-specific files are necessary for regression testing, which are not attached here.

Stanislav, Ovidiu, Vadim:  please base your conversion work on this model. Add new methods, attributes, functions to the Buffer interface and BufferImpl class, observing the naming restriction noted above.

#### #7 - 01/23/2013 04:49 PM - Vadim Nebogatov

Methods  RecordBuffer.copy() and  RecordBuffer.openScope() are not deprecated, but compiler makes warnings

```
compile:
    [javac] Compiling 1006 source files to /home/vmn/p2j/build/classes
    [javac] /home/vmn/p2j/src/com/goldencode/p2j/persist/BufferImpl.java:354: warning: [deprecation] copy(com.
goldencode.p2j.persist.DataModelObject,com.goldencode.p2j.persist.DataModelObject,boolean) in com.goldencode.p
2j.persist.RecordBuffer has been deprecated
    [javac]       RecordBuffer.copy((BufferImpl) bufHandle.get(), this, true);
    [javac]                    ^
    [javac] /home/vmn/p2j/src/com/goldencode/testcases/Vadim22.java:52: warning: [deprecation] openScope(com.g
oldencode.p2j.persist.DataModelObject...) in com.goldencode.p2j.persist.RecordBuffer has been deprecated
    [javac]             RecordBuffer.openScope(btest, hbuffer);
    [javac]                          ^
    [javac] Note: Some input files use unchecked or unsafe operations.
    [javac] Note: Recompile with -Xlint:unchecked for details.
    [javac] 2 warnings
```

Possible it takes “deprecated” keyword from some commented lines above these methods.

#### #8 - 01/23/2013 05:18 PM - Eric Faulhaber

Yes, I should have removed the commented methods once I got everything compiling.  I didn't notice this when I built the regression testing application, as I just saw that the build was successful, but missed the summary lines:

```
    [javac] Note: Some input files use or override a deprecated API.
    [javac] Note: Recompile with -Xlint:deprecation for details.
```

Thanks for pointing this out.  This means the javadoc will likely be wrong, too. I'll remove these commented, deprecated methods in the final update.

#### #9 - 01/25/2013 02:12 PM - Eric Faulhaber

*- Status changed from WIP to Closed*

*- % Done changed from 0 to 100*

Update has passed regression testing;  committed to bzr (rev 10154).

#### #10 - 01/28/2013 03:42 AM - Stanislav Lomany

Eric, please be aware that define parameter buffer is not converted properly:

```
Book.Buf buf1;

public void execute(final Book buf1)
{
   externalProcedure(new Block()
   {
      public void body()
      {
         RecordBuffer.openScope(buf1);
         ExternalProc.this.buf1 = buf1;  /* different types */
       }
   });
}
```

#### #11 - 01/28/2013 04:02 AM - Constantin Asofiei

I confirm for temp-tables too, please check the buffer_test2a.p and buffer_test2b.p (they are in bazaar).

#### #12 - 01/28/2013 02:24 PM - Eric Faulhaber

Stas, Constantin, thanks for letting me know about this. I have an update which fixes this for me locally, but it needs to go through testing still. I'll post it here for your temporary use, once I've cleaned it up.

#### #13 - 01/28/2013 09:31 PM - Eric Faulhaber

*- File ecf_upd20130128a.zip added*

Note this fix has not yet been regression tested. Though I couldn't find any unanticipated conversion differences in Majic due to this fix, a comparison with an older drop of converted code was difficult, because there were many unrelated changes due to other updates that have since been merged into my local code base.

#### #14 - 03/04/2013 10:33 AM - Ovidiu Maxiniuc

*- File om_upd20130304b.zip added*

This update fixes eventual collisions in dmo interfaces because of WrappedResource interface (isValid() & isUnknown()).

#### #15 - 03/04/2013 10:47 AM - Constantin Asofiei

Unfortunately, at some point we might have to rename the APIs in these interfaces too, which are extended by Buffer interface:

- Errorable
- DatabaseInfo
- ADMData
- UniqueID
- NamespaceURI

#### #16 - 03/05/2013 01:48 PM - Constantin Asofiei

*- File ca_upd20130305d.zip added*

Fix cases of a FOR EACH block converted to a tt.delete call, in case the FOR EACH is enclosed in a statement - in this case, wrap it in a block.

```
def temp-table tt1 field f1 as int.
def var i as int.

if i = 0
then for each tt1:
        delete tt1.
     end.
else message "bla".
```

#### #17 - 03/05/2013 04:18 PM - Constantin Asofiei

I'm putting this through conversion regression testing.

#### #18 - 03/05/2013 05:15 PM - Constantin Asofiei

~~Passed conversion regression testing.~~

#### #19 - 03/05/2013 05:16 PM - Constantin Asofiei

I'm taking this back, wrong task: this has some regressions in terms that is too aggressive when bracketing the deleteAll call, there are other cases when it doesn't need to.

I'll fix this and re-test.

#### #20 - 03/05/2013 09:50 PM - Eric Faulhaber

*- File ecf_upd20130305a.zip added*

Code review ~~20130304a~~ 20130304b:

The code looks fine. The attached update corrects some cut-and-paste errors in the file headers. It should be applied after om_upd20130304b.zip.

I'm going to run this through conversion regression testing with some other updates. I just want to make sure Majic compiles correctly with these changes.

#### #21 - 03/05/2013 11:36 PM - Eric Faulhaber

The om_upd20130304b.zip and ecf_upd20130305a.zip updates have gotten through conversion regression testing. However, Majic requires an update to hand-written Java code which implements the WrappedResource interface. See #2081.

Please commit and distribute the combined update.

#### #22 - 03/06/2013 02:25 AM - Constantin Asofiei

*- File ca_upd20130306b.zip added*

Attached update fixes the problem for my 0305d.zip. It brackets the deleteAll with a block only if it's parent is not a block.

#### #23 - 03/06/2013 04:46 AM - Constantin Asofiei

ca_upd20130306b.zip has passed conversion regression testing (no changes in generated sources).

#### #24 - 03/06/2013 07:14 AM - Constantin Asofiei

0306b.zip was committed to bzr revision 10253.

#### #25 - 11/16/2016 11:07 AM - Greg Shah

*- Target version changed from Milestone 4 to Conversion Support for Server Features*

### Files

| | | | |
|---|---|---|---|
| ecf_upd20130123a.zip | 209 KB | 01/23/2013 | Eric Faulhaber |
| ecf_upd20130128a.zip | 15 KB | 01/29/2013 | Eric Faulhaber |
| om_upd20130304b.zip | 183 KB | 03/04/2013 | Ovidiu Maxiniuc |
| ca_upd20130305d.zip | 6.03 KB | 03/05/2013 | Constantin Asofiei |
| ecf_upd20130305a.zip | 25.5 KB | 03/06/2013 | Eric Faulhaber |
| ca_upd20130306b.zip | 6.03 KB | 03/06/2013 | Constantin Asofiei |