POLITECNICO
MILANO 1863

Computer Science and Engineering
Project of Software Engineering 2

Inspection Document

Authors:

Alberto Zeni 813977

Manuel Parenti 876085

Reference Professor: Mottola Luca

Academic Year 2016–2017

Table of Contents

1 Classes that were assigned to the group:

The class that was assigned to our group is: ServiceXaWrapper.java
And its location in the ofbiz directory is the following:

../apache-ofbiz-16.11.01/framework/service/src/main/java/org/apache/ofbiz/service/ServiceXaWrapper.java

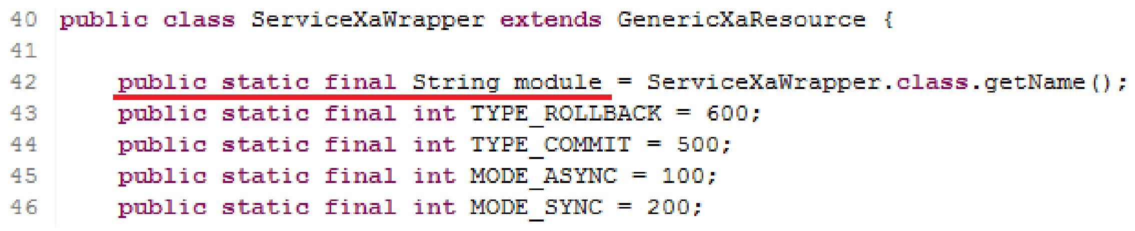
2 Functional role of assigned set of classes

The class we need to analyze is a wrapper for a Xa Resource. This means it acts like a translator for the instructions it receives in order to make them comprehensible for the Xa Resource. In particular, our class focus on running services on commit and rollback methods, defined in the generic class that our class extends.

3 List of issues found by applying the checklist

3.1 Naming conventions issues

The constant “module” is not written in capital letters.



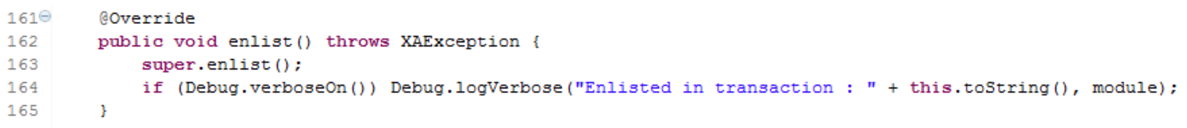
```
40 public class ServiceXaWrapper extends GenericXaResource {
41
42     public static final String module = ServiceXaWrapper.class.getName();
43     public static final int TYPE_ROLLBACK = 600;
44     public static final int TYPE_COMMIT = 500;
45     public static final int MODE_ASYNC = 100;
46     public static final int MODE_SYNC = 200;
```

3.2 Indentation issues

No indentation issues have been found.

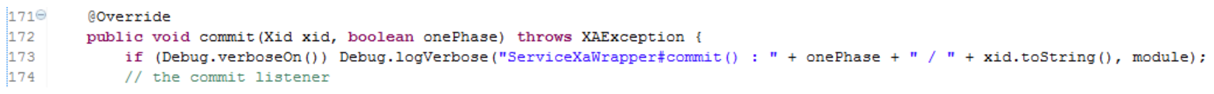
3.3 Braces issues

The statement following the “if” clause at line 164 is not included in curly braces. Also, it could become a shorter and more readable line.



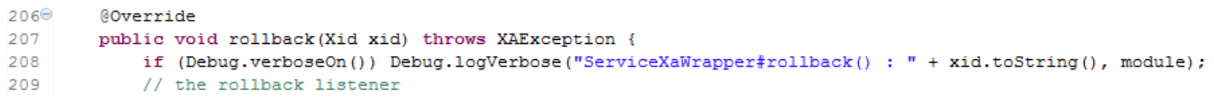
```
161    @Override
162    public void enlist() throws XAException {
163        super.enlist();
164        if (Debug.verboseOn()) Debug.logVerbose("Enlisted in transaction : " + this.toString(), module);
165    }
```

The statement following the “if” clause at line 173 is not included in curly braces. Also, it could become a shorter and more readable line.



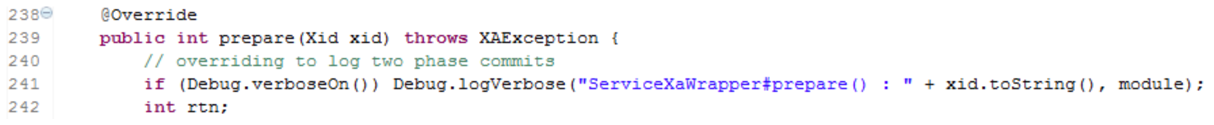
```
171    @Override
172    public void commit(Xid xid, boolean onePhase) throws XAException {
173        if (Debug.verboseOn()) Debug.logVerbose("ServiceXaWrapper#commit() : " + onePhase + " / " + xid.toString(), module);
174        // the commit listener
```

The statement following the “if” clause at line 208 is not included in curly braces. Also, it could become a shorter and more readable line.



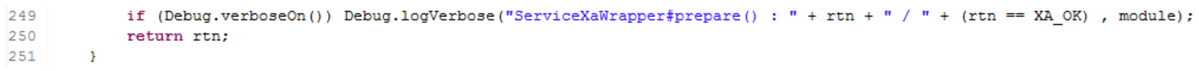
```
206    @Override
207    public void rollback(Xid xid) throws XAException {
208        if (Debug.verboseOn()) Debug.logVerbose("ServiceXaWrapper#rollback() : " + xid.toString(), module);
209        // the rollback listener
```

The statement following the “if” clause at line 241 is not included in curly braces. Also, it could become a shorter and more readable line.



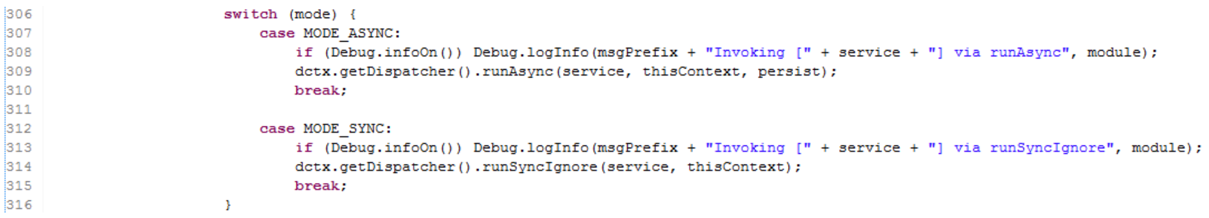
```
238    @Override
239    public int prepare(Xid xid) throws XAException {
240        // overriding to log two phase commits
241        if (Debug.verboseOn()) Debug.logVerbose("ServiceXaWrapper#prepare() : " + xid.toString(), module);
242        int rtn;
```

The statement following the “if” clause at line 249 is not included in curly braces. Also, it could become a shorter and more readable line.



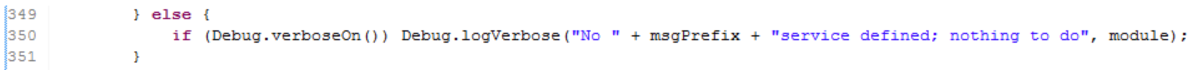
```
249        if (Debug.verboseOn()) Debug.logVerbose("ServiceXaWrapper#prepare() : " + rtn + " / " + (rtn == XA_OK) , module);
250        return rtn;
251    }
```

The statements following the “if” clauses at lines 308 and 313 are not included in curly braces. Also, they could become shorter and more readable lines.



```
306                switch (mode) {
307                    case MODE_ASYNC:
308                        if (Debug.infoOn()) Debug.logInfo(msgPrefix + "Invoking [" + service + "] via runAsync", module);
309                        dctx.getDispatcher().runAsync(service, thisContext, persist);
310                        break;
311
312                    case MODE_SYNC:
313                        if (Debug.infoOn()) Debug.logInfo(msgPrefix + "Invoking [" + service + "] via runSyncIgnore", module);
314                        dctx.getDispatcher().runSyncIgnore(service, thisContext);
315                        break;
316                }
```

The statements following the “if” clause at line 350 is not included in curly braces. Also, it could become a shorter and more readable line.

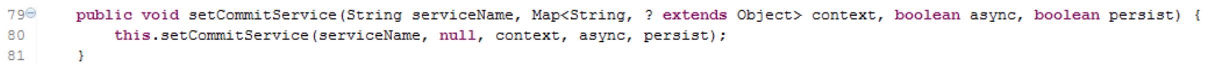


```
349        } else {
350            if (Debug.verboseOn()) Debug.logVerbose("No " + msgPrefix + "service defined; nothing to do", module);
351        }
```

Every other line of code follows the Kernighan & Ritchie style.

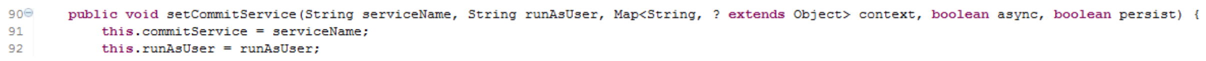
3.4 File organization issues

Line 79 is longer than 120 characters (126).



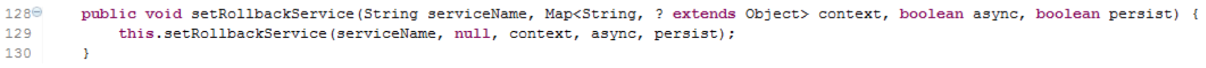
```
79     public void setCommitService(String serviceName, Map<String, ? extends Object> context, boolean async, boolean persist) {
80         this.setCommitService(serviceName, null, context, async, persist);
81     }
```

Line 90 is longer than 120 characters (144).



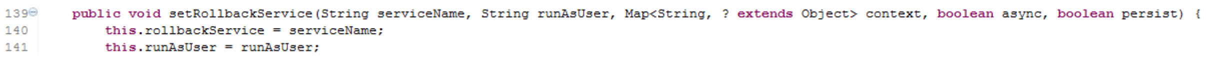
```
90     public void setCommitService(String serviceName, String runAsUser, Map<String, ? extends Object> context, boolean async, boolean persist) {
91         this.commitService = serviceName;
92         this.runAsUser = runAsUser;
```

Line 128 is longer than 120 characters (128).



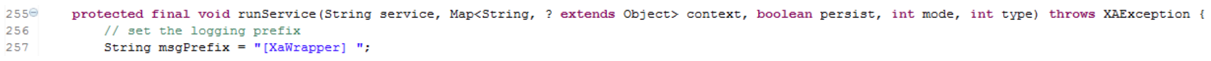
```
128    public void setRollbackService(String serviceName, Map<String, ? extends Object> context, boolean async, boolean persist) {
129        this.setRollbackService(serviceName, null, context, async, persist);
130    }
```

Line 139 is longer than 120 characters (146).



```
139    public void setRollbackService(String serviceName, String runAsUser, Map<String, ? extends Object> context, boolean async, boolean persist) {
140        this.rollbackService = serviceName;
141        this.runAsUser = runAsUser;
```

Line 255 is longer than 120 characters (149).



```
255    protected final void runService(String service, Map<String, ? extends Object> context, boolean persist, int mode, int type) throws XAException {
256        // set the logging prefix
257        String msgPrefix = "[XaWrapper] ";
```

3.5 Wrapping lines issues

No wrapping lines issues have been found.

3.6 Comments issues

No comments issues have been found.

3.7 Java source files issues

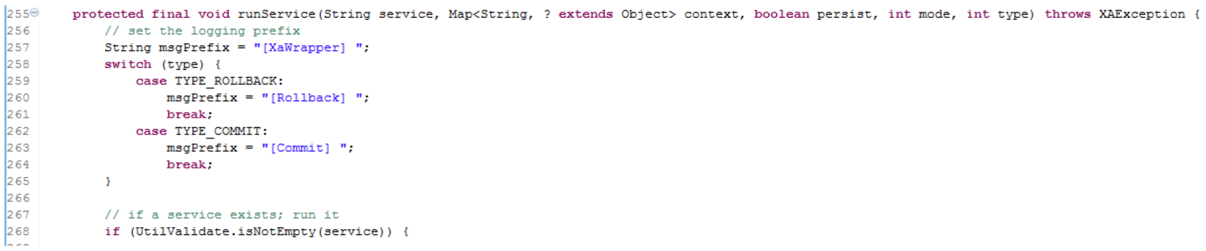
No java source file issues have been found.

3.8 Package and Import Statements issues

No issues concerning package and import statements have been found.

3.9 Class and interface declarations issues

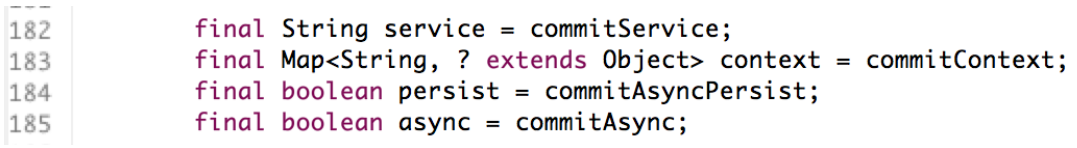
The method runService defined at line 255 is 100 lines long. We think that it should be considerably shorter, this issue might be fixed dividing this method in at least 2-3 submethods.



```
255    protected final void runService(String service, Map<String, ? extends Object> context, boolean persist, int mode, int type) throws XAException {
256        // set the logging prefix
257        String msgPrefix = "[XaWrapper] ";
258        switch (type) {
259            case TYPE_ROLLBACK:
260                msgPrefix = "[Rollback] ";
261                break;
262            case TYPE_COMMIT:
263                msgPrefix = "[Commit] ";
264                break;
265        }
266
267        // if a service exists; run it
268        if (UtilValidate.isNotEmpty(service)) {
```

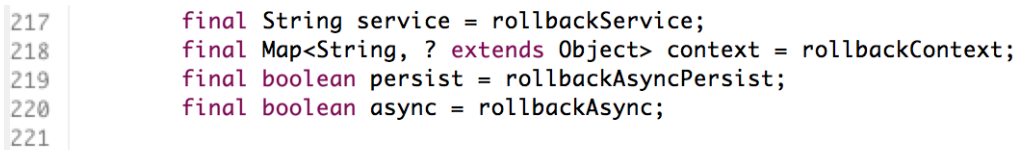
3.10 Initializations and declarations issues

The variables 'service', 'context', 'persist', 'async' declared starting from line 182 are not declared at the beginning of the block.



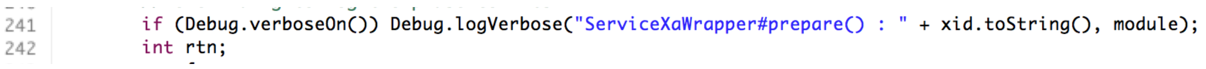
```
182            final String service = commitService;
183            final Map<String, ? extends Object> context = commitContext;
184            final boolean persist = commitAsyncPersist;
185            final boolean async = commitAsync;
```

The variables 'service', 'context', 'persist', 'async' declared starting from line 217 are not declared at the beginning of the block.



```
217            final String service = rollbackService;
218            final Map<String, ? extends Object> context = rollbackContext;
219            final boolean persist = rollbackAsyncPersist;
220            final boolean async = rollbackAsync;
221
```

The variable 'rtn' declared at line 242 is not declared at the beginning of the block.



```
241        if (Debug.verboseOn()) Debug.logVerbose("ServiceXaWrapper#prepare() : " + xid.toString(), module);
242        int rtn;
```

3.11 Method calls issues

No method calls issues have been found.

3.12 Arrays issues:

No array issues have been found.

3.13 Object comparison issues

The xid variable at line 178 is compared using the '==' instead of equal.

```
178         if (this.xid == null || !this.xid.equals(xid)) {
179             throw new XAException(XAException.XAER_NOTA);
```

The xid variable at line 213 is compared using the '==' instead of equal.

```
213         if (this.xid == null || !this.xid.equals(xid)) {
214             throw new XAException(XAException.XAER_NOTA);
```

The rtn variable at line 249 is compared using the '==' instead of equal.

```
    if (Debug.verboseOn()) Debug.logVerbose("ServiceXaWrapper#prepare() : " + rtn + " / " + (rtn == XA_OK) , module);
    return rtn;
```

3.14 Output format issues

In terms of output readability there are no issues, but we think that the way in which the strings are dealt with in the following snapshots of code could be changed in a better implementation. The string we are referring to is "msgPrefix" and it is defined at line 257.

```
257         String msgPrefix = "[XaWrapper] ";
258         switch (type) {
259             case TYPE_ROLLBACK:
260                 msgPrefix = "[Rollback] ";
261                 break;
262             case TYPE_COMMIT:
263                 msgPrefix = "[Commit] ";
264                 break;
265         }
```

It is used in the following outputs:

```
307                     case MODE_ASYNC:
308                         if (Debug.infoOn()) Debug.logInfo(msgPrefix + "Invoking [" + service + "] via runAsync", module);
309                         dctx.getDispatcher().runAsync(service, thisContext, persist);
310                         break;
311
312                     case MODE_SYNC:
313                         if (Debug.infoOn()) Debug.logInfo(msgPrefix + "Invoking [" + service + "] via runSyncIgnore", module);
314                         dctx.getDispatcher().runSyncIgnore(service, thisContext);
315                         break;
```

```
317                 } catch (Throwable t) {
318                     Debug.logError(t, "Problem calling " + msgPrefix + "service : " + service + " / " + context, module);
319                     try {
320                         TransactionUtil.rollback(beganTx, t.getMessage(), t);
```

```
349         } else {
350             if (Debug.verboseOn()) Debug.logVerbose("No " + msgPrefix + "service defined; nothing to do", module);
351         }
```

We think that a more general and safe way to write these outputs is to write a space after the "msgPrefix" variable and to delete the spaces in the "msgPrefix" text.

3.15 Computation, comparisons and assignments issues:

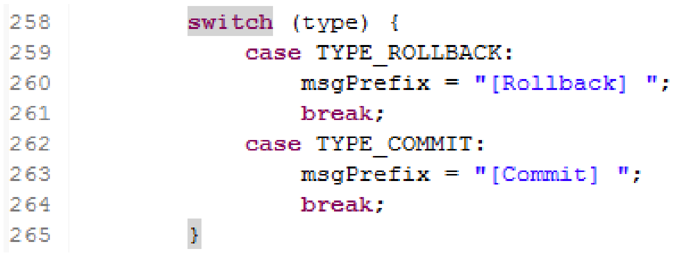
No computation, comparisons or assignments issues have been found.

3.16 Exceptions issues

No exceptions issues have been found.

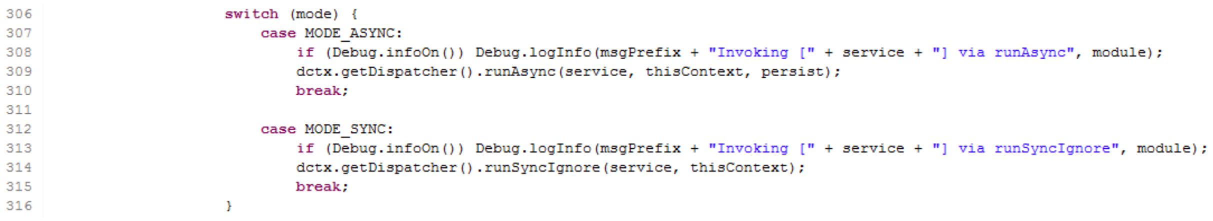
3.17 Flow of control issues

The switch at line 258 hasn't a default branch.



```
258         switch (type) {
259             case TYPE_ROLLBACK:
260                 msgPrefix = "[Rollback] ";
261                 break;
262             case TYPE_COMMIT:
263                 msgPrefix = "[Commit] ";
264                 break;
265         }
```

The switch at line 306 hasn't a default branch.



```
306                     switch (mode) {
307                         case MODE_ASYNC:
308                             if (Debug.infoOn()) Debug.logInfo(msgPrefix + "Invoking [" + service + "] via runAsync", module);
309                             dctx.getDispatcher().runAsync(service, thisContext, persist);
310                             break;
311 
312                         case MODE_SYNC:
313                             if (Debug.infoOn()) Debug.logInfo(msgPrefix + "Invoking [" + service + "] via runSyncIgnore", module);
314                             dctx.getDispatcher().runSyncIgnore(service, thisContext);
315                             break;
316                     }
```

3.18 Files issues

No files are opened by the methods of the class, so no issues have been found.

4 Additional Problems

Many references about the xid and the active objects, that are presumably part of the GenericXaResource class, are in the code, but since this is an extended class we don't know what they are. Because of this they could be sources for problems during the implementation of our class.

5 Additional Information

5.1 Hours of Work

Time spent to write this document:

Manuel Parenti: 7 hours

Alberto Zeni: 7 hours

5.2 Used Tools

The tools that supported us are:

MS Office Word 2016: to write and redact this document;

Eclipse: to read the java code of the class.