Mulgara - Bug #44

Error during transaction.commit() causes infinite recursion and looping

02/25/2007 09:13 AM - ronald -

Status: Closed Start date: **Priority:** High Due date:

% Done: Assignee: Andrae Muys -0%

Category: **Estimated time:** 0.00 hour Mulgara

Target version:

Resolution: fixed

Description

```
This is svn head (rev <a href="http://mulgara.org/trac/changeset/189">189</a>
), i.e. with the new transaction code.
<br/>
<br/>
Here's a small snippet of the stack when the error is triggered:
<br/>
<br/>
             
at org.objectweb.jotm.TransactionImpl.commit(TransactionImpl.java:222)
<br/>
             
at org.mulgara.resolver.MulgaraTransaction.terminateTransaction(MulgaraTransaction.java:330)
<br/>
             
at org.mulgara.resolver.MulgaraTransaction.deactivate(MulgaraTransaction.java:154)
<br/>
             
at org.mulgara.resolver.MulgaraTransaction.execute(MulgaraTransaction.java:207)
<br/>
<br/>
If commit() throws an exception, implicitRollback() will be invoked at
<br/>
line 333, which in turn call checkActivated() on line 262, which in
<br/>
turn calls implicitRollback() on line 442 because inuse is 0 (from
<br/>
deactivate()).
<br/>
<br/>
At this point things go horribly wrong...
<br/>
<br/>
An easy way to trigger this is run out of disk space where the db
<br/>
resides (e.g. on linux create a fs on a small file and mount using the
<br/>
loop device: 'dd if=/dev/zero of=/tmp/mdb bs=4096 count=1000',
<br/>
'mke2fs /tmp/mdb', 'mount -o loop /tmp/mdb /mnt/disk').
<br/>
<br/>
```

History

04/20/2024 1/3

#1 - 02/27/2007 01:01 AM - Andrae Muys -

I'll address this next week, for now just comment out the call to checkActivated() - it shouldn't ever fail for those times it probably manages to anyway, I'll figure what to do there next week.

#2 - 02/27/2007 01:39 AM - ronald -

Just to confirm: we're talking about commenting out line 263 (as of rev 192), right?

#3 - 02/27/2007 01:53 AM - Andrae Muys -

```
Yes.
<br/>
<br/>
        try {
<br/>
      checkActivated();
//
<br/>
          rollback = IMPLICIT_ROLLBACK;
<br/>
          rollbackCause = cause;
<br/>
            failTransaction();
<br/>
     anbsp;return new [[MulgaraTransactionException]]("Transaction in Rollback
", cause);
<br/>
        } catch (Throwable th) {
<br/>
      abortTransaction("Failed to rollback normally", th);
<br/>
      throw new [[MulgaraTransactionException]]("
Abort failed to throw exception", th);
<br/>
        }
<br/>
<br/>
The check is there as a defensive measure as execution should only reach that point if the transaction is acti
vated. Removing it will ensure rollback get's set, which will short-circuit any future call to implicitRollba
ck().
<br/>
<br/>
I'll take a closer look and determine what the permanent fix should be next week.
```

#4 - 02/27/2007 04:31 AM - ronald -

```
I had to make one small additional modification: line 357 needs to
<br/>
be protect with an if:
<br/>
<br/>
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        &n
<br/>
          if (manager != null)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        &n
manager.transactionComplete(this);
<br/>
              
<br/>
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        &n
         inuse = 0;
<br/>
<br/>
This is because on line 335 terminateTransaction() is called recursively,
<br/>
and the third finally clause sets manager to null, so after the
<br/>
terminateTransaction() returns the first time you get a NPE otherwise.
```

04/20/2024 2/3

#5 - 03/19/2007 07:04 AM - Andrae Muys -

Fixed in revision 207

04/20/2024 3/3