Base Language - Bug #6570

stale procedure leak when appservers are used

06/30/2022 11:02 AM - Constantin Asofiei

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Alexandru Lungu	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#2 - 06/30/2022 11:16 AM - Constantin Asofiei

DeferredDeletablesManager.registerAt is called with blockDepth set to 0 in some cases. For a normal GUI/ChUI client, this registers the DeferredDeletable with the global scope, which gets iterated when the FWD client is terminated.

The appservers are a special type of client - they are very long running, and the TransactionManager.globalScope gets iterated only when the agent gets terminated. So DeferredDeletable instances can accumulate in this global block (in the DeferredDeletablesManager finalizable instance), and never get cleaned up.

In OE, if the Agent OOME's, then it just gets abended and another is started. In FWD, this is not the case, as it runs in the FWD server.

There is also another problem with the DeferredDeletablesManager.registerAt code - if blockDepth == 0, then a new DeferredDeletablesManager is created and registered in the TransactionManager.globalBlock finalizables, instead of calling DeferredDeletablesManager with the TransactionManager.globalBlock instance.

I would be inclined to use (in case of appserver) the agent's "startup" block; for an Agent there is:

- the "appserver-agent" block which is pushed first (when the agent is started) - this block is iterated only when the agent terminates, or if its state gets reset, this scope is pushed again
- the "startup" block which is pushed when a request is processed, and popped/processed when the request ends

Alexandru: please remind me the rules about the stale/deferred-delete procedures. Are the stale/deferred delete procedures really 'leaked' for the duration of a GUI/ChUI client? What test can I use to see this behavior?

#3 - 07/01/2022 03:04 AM - Alexandru Lungu

The implementation of stale procedures was done in #4187.

The issue: Multiple deletes on the same procedure were not showing the "Invalid or inappropriate handle" error. Basically, the logical delete and physical delete are happening in different blocks. One can explicitly delete a handle but only logically. The physical delete is done only once (when

the procedure is cleared from the memory), but at an "unknown" point of time.

The investigation: The tests (#4187-35, #4187-58, etc.) shows that only persistent procedures using global buffers are subject of becoming "stale" (logically deleted, but not also physically). The physical delete seems to always be done at the block that caused (maybe after several run) both the creation and deletion of the persistent procedure. However, this is more of an empirical observation after testing scenarios with: multiple procedure files, transactions, buffers, OO etc.

The solution: The delete of a persistent procedure with a global buffer makes it "stale", such that any further delete doesn't show any error. A DeferredDeletablesManager was implemented to clean up the stale procedures at the right block.

> DeferredDeletablesManager.registerAt is called with blockDepth set to 0 in some cases. For a normal GUI/ChUI client, this registers the DeferredDeletable with the global scope, which gets iterated when the FWD client is terminated.

I see that there are many stale procedures registered with a small blockDepth, which is not ideal but expected, as the early blocks usually cause both creation and deletion of a persistent procedure with buffers.

> There is also another problem with the DeferredDeletablesManager.registerAt code - if blockDepth == 0, then a new DeferredDeletablesManager is created and registered in the TransactionManager.globalBlock finalizables, instead of calling DeferredDeletablesManager with the TransactionManager.globalBlock instance.

I recall that a single DeferredDeletablesManager can be registered at a certain block, but I may be wrong. This needs to be investigated.

> I would be inclined to use (in case of appserver) the agent's "startup" block; for an Agent there is:

Right. It makes sense that a stale procedure lifecycle is managed in one single request and thus the clean up should be done when a "startup" block is popped. Conceptually, this makes sense and should reflect the behavior in FWD. Right now, (without proper investigation) I tend to think that the only way to leak onto the "appserver-agent" is when the creation and deletion of a persistent procedure with buffer(s) are done in different requests. Is this possible?

#4 - 10/31/2023 04:02 AM - Alexandru Lungu

- Assignee set to Alexandru Donica

- Status changed from New to WIP

#9 - 03/21/2024 09:29 AM - Alexandru Lungu

- Assignee changed from Alexandru Donica to Alexandru Lungu