**Disclaimer:**

I have neither given nor received unauthorized assistance on this work.

Signed:

Hardik Jain (ID: 1001954448) Date: 12/05/2021

Anuhya Patibanda (ID: 1001969235)

**Two Phase Commit Protocol**

**Concept:**

1. The coordinator sends a VOTE REQUEST message to all participants.
2. When a participant receives a VOTE REQUEST message, it returns either a VOTE COMMIT message to the coordinator telling the coordinator that it is prepared to locally commit its part of the transaction, or otherwise, a VOTE-ABORT message.
3. The coordinator collects all votes from the participants. If all participants have voted to commit the transaction, then so will the coordinator. In that case, it sends a GLOBAL\_COMMIT message to all participants. However, if one participant had voted to abort the transaction, the coordinator will also decide to abort the transaction and multicasts a GLOBAL ABORT message.
4. Each participant that voted for a commit waits for the final reaction by the coordinator. If a participant receives a GLOBAL\_COMMIT message, it locally commits the transaction. Otherwise, when receiving a GLOBAL ABORT message, the transaction is locally aborted as well.

Coordinator Participant

QUERY TO COMMIT

-------------------------------->

VOTE YES/NO prepare\*/abort\*

<-------------------------------

commit\*/abort\* COMMIT/ROLLBACK

-------------------------------->

ACKNOWLEDGMENT commit\*/abort\*

<--------------------------------

end

**Instructions:**

* Unzip the folder
* The folder contains 2 subfolders – Case1 and Case2
* Case1 contains files which showcase how a coordinator deals with votes if all processes vote “commit” or if any of the process votes “abort”
* Case2 contains file which showcase the scenario of a failure using sleep function. How does processes react when a coordinator fails or how does a coordinator react when any of the process fails.

**How to run the code:**

* Open Lab3 > Case1 / Case2 >
* Run coordinator.py in 1st terminal
* Run process.py in 3 separate terminals to create 3 processes.

**Libraries used:**

- Socket

- time

- thread

**Note -** Minimum Python version required - 3.9.7

**Screenshots:**

1. After running coordinator.py

Text

Description automatically generated

1. After running process.py in 3 different terminals coordinator initiates voting

Graphical user interface, text, application

Description automatically generated

Scenario 1: When all processes vote “Commit”, Global commit message is released.

Graphical user interface, text, application

Description automatically generated

Scenario 2: When any of the processor votes “Abort”, Global Abort message is released.

Graphical user interface, text, application

Description automatically generated

Scenario 3: If coordinator experiences a failure (Failure has been showcased using a delay at coordinator’s end which is much longer than the process’s delay)

Graphical user interface, text, application

Description automatically generated

**References:**

* Distributed Systems: Principles and Paradigms

(Author: Andrew S. Tanenbaum, Maarten Van Steen)