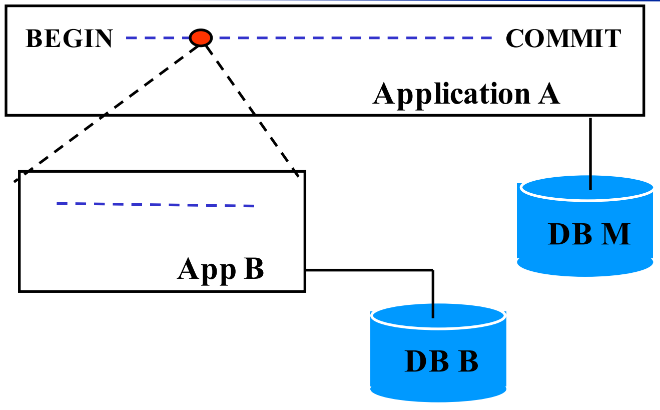
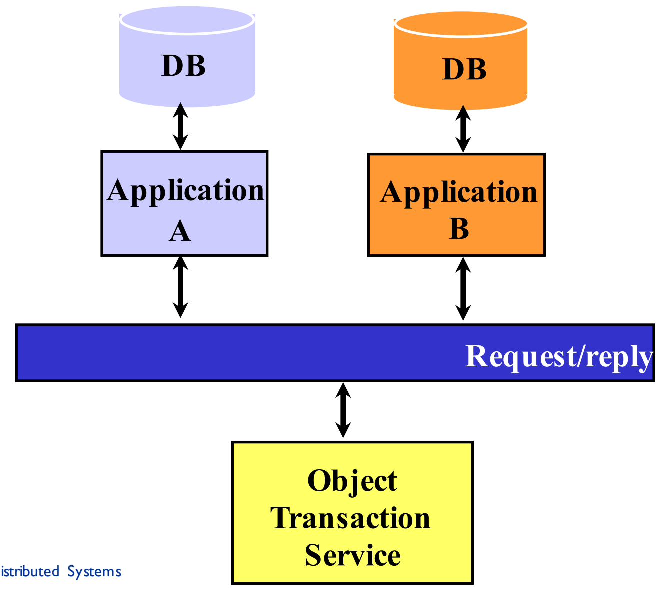
Example (CORBA/JTS)

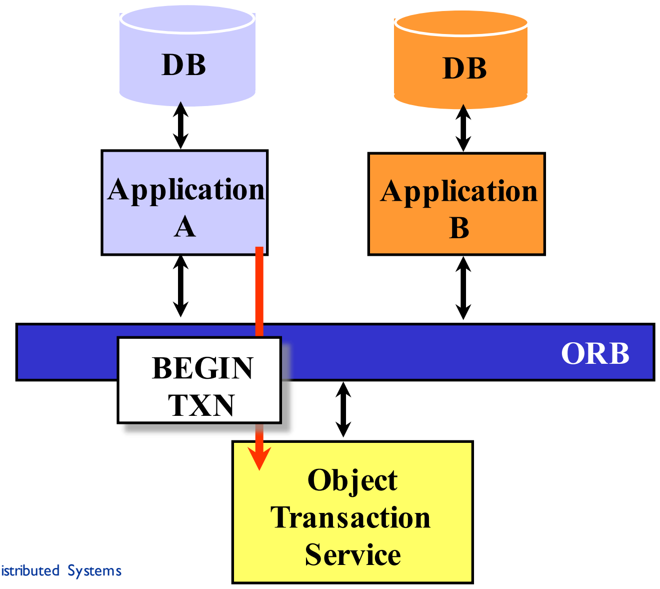


- Project:
  - ☆ Replace Application A with Middleware (middleware has own DB with customer information)
  - ☆ Replication App B with Flight
  - ☆ Instead of DB M / DB B we have things so far in main memory

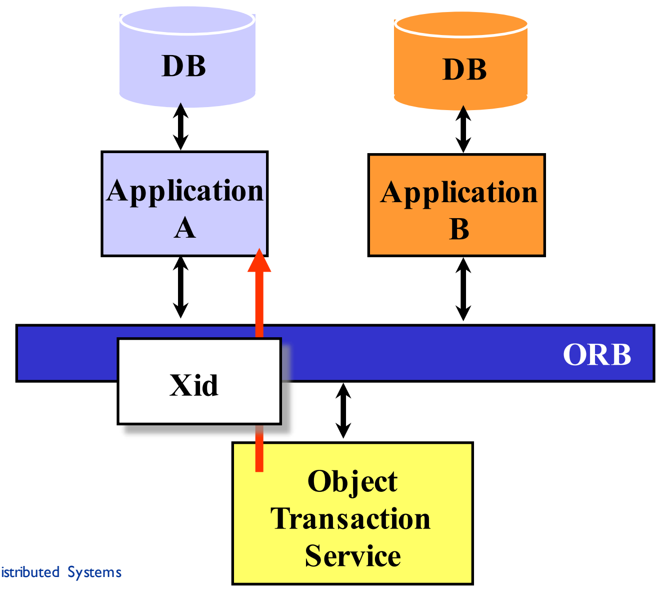
Example I



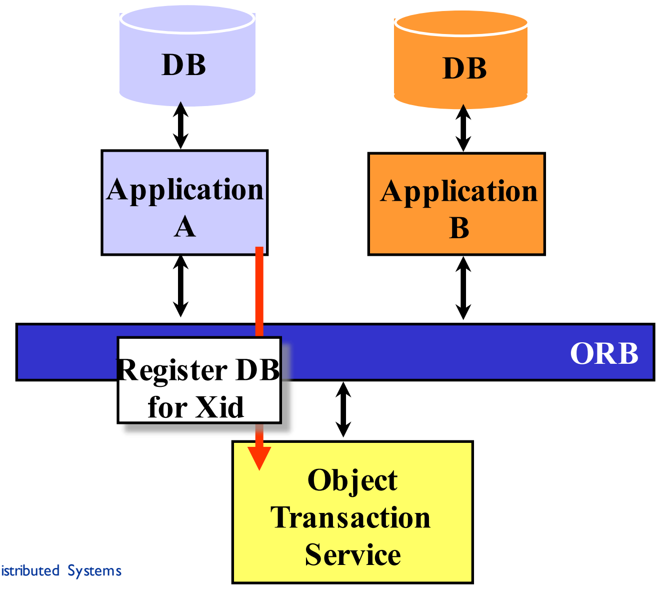
Example 2



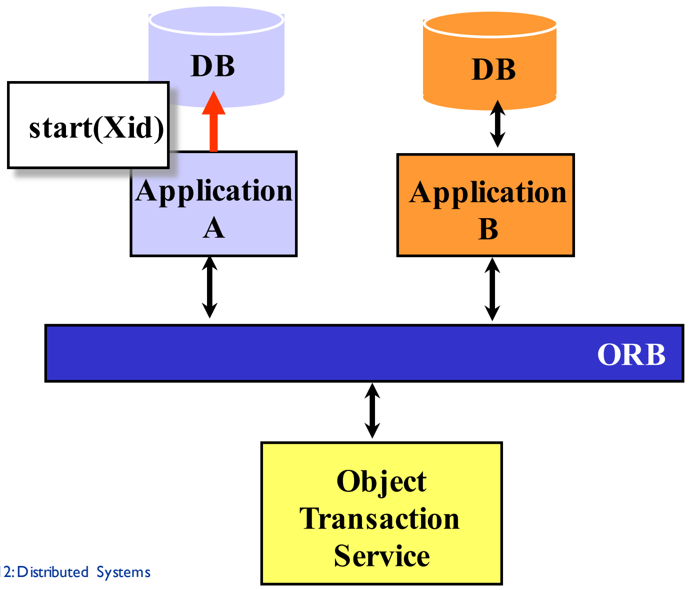
Example 3



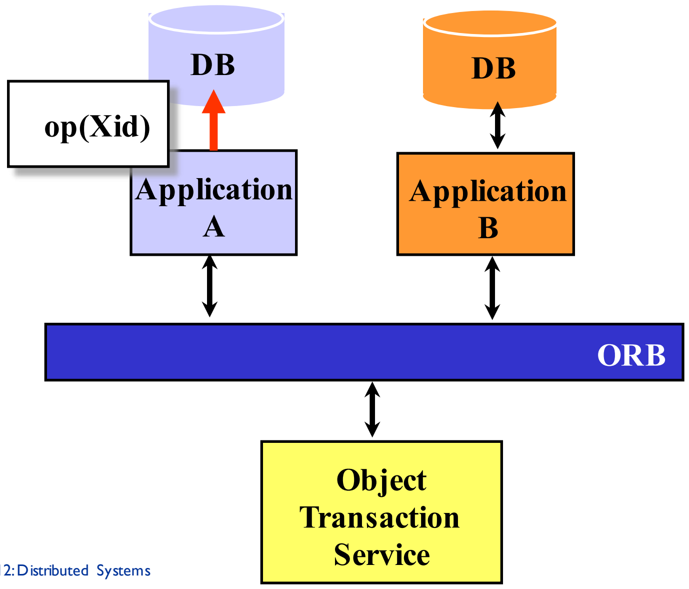
Example 4



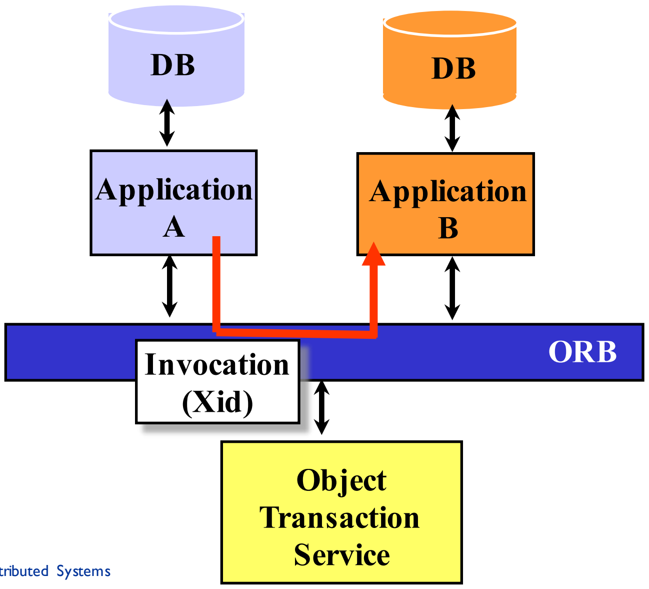
Example 5



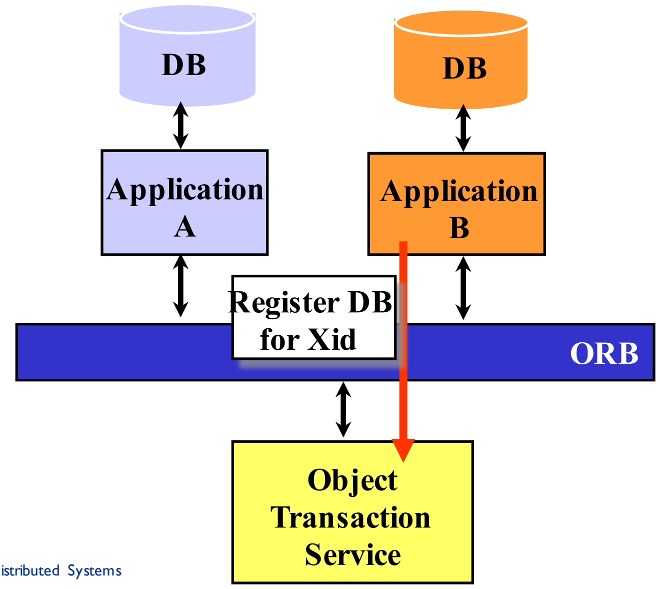
Example 6



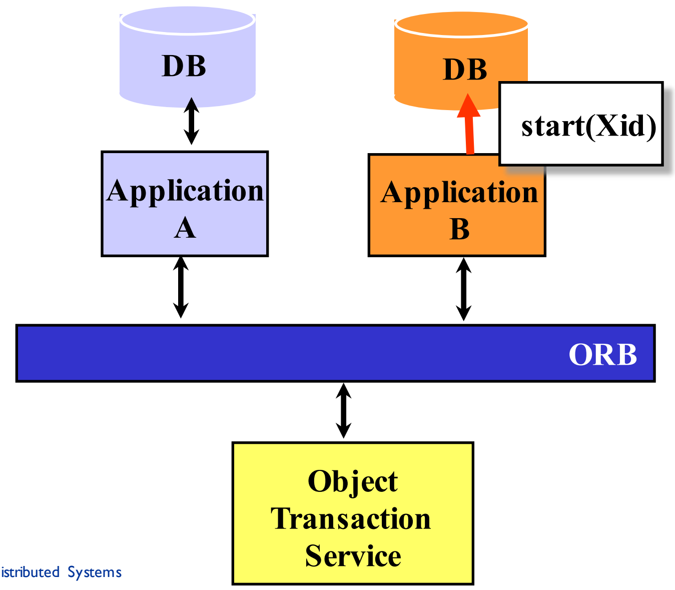
Example 7



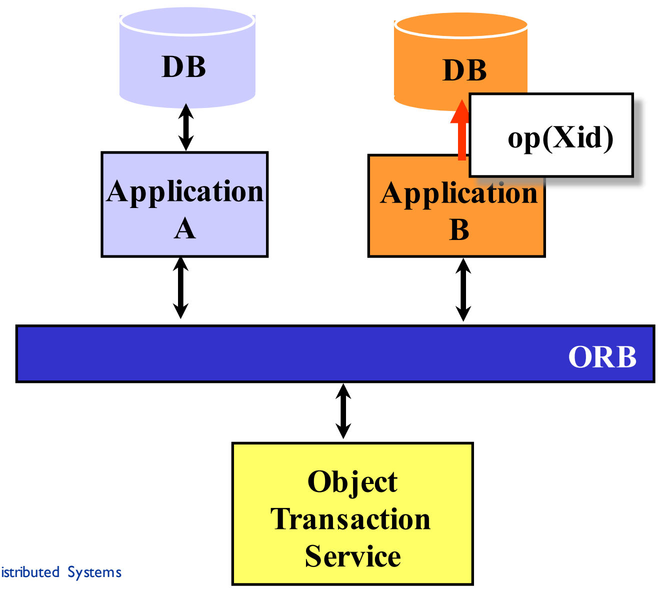
Example 8



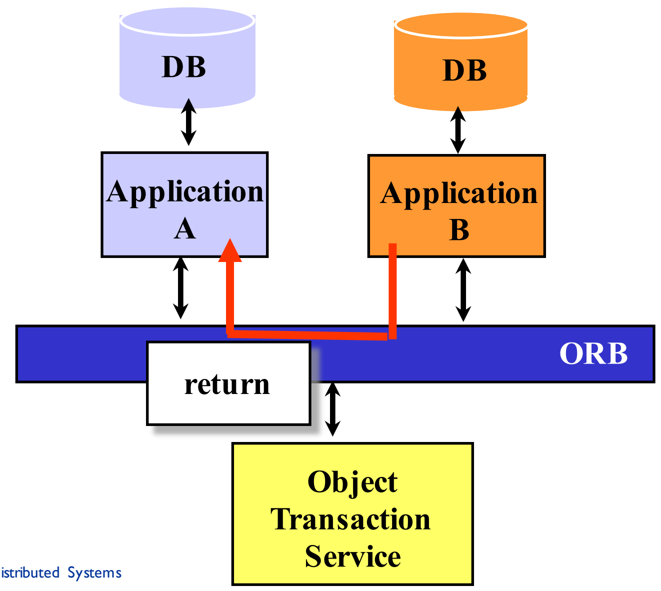
Example 9



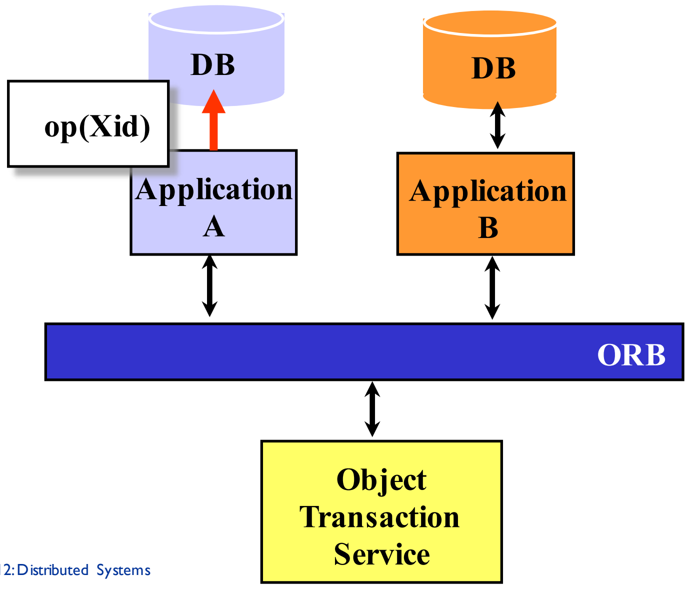
Example 10



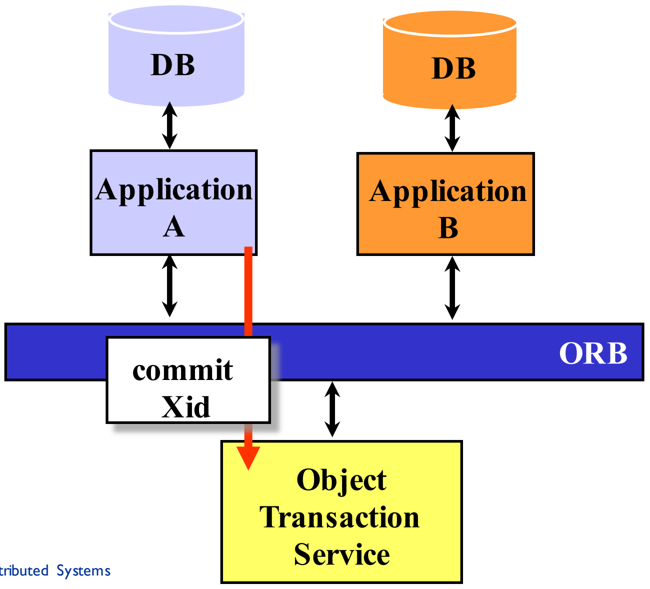
Example 11



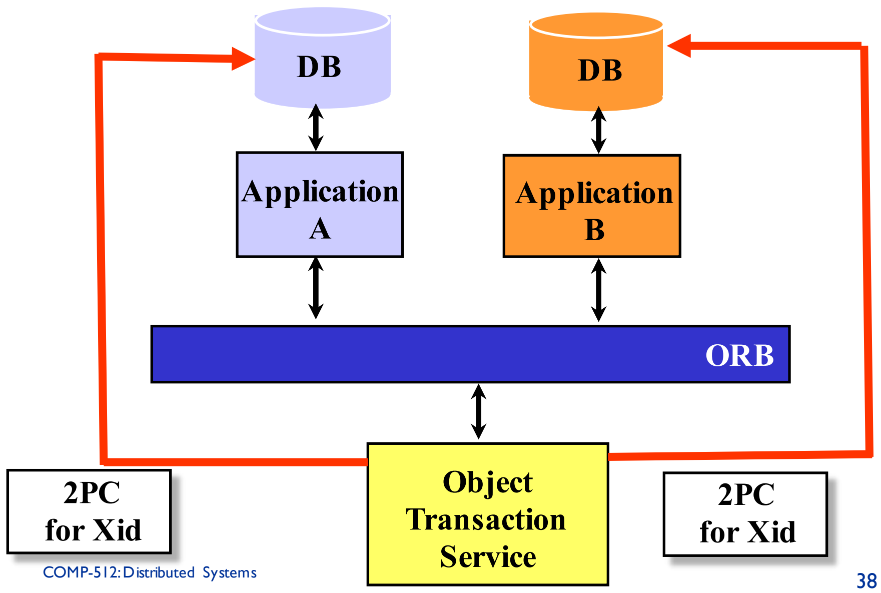
Example 12



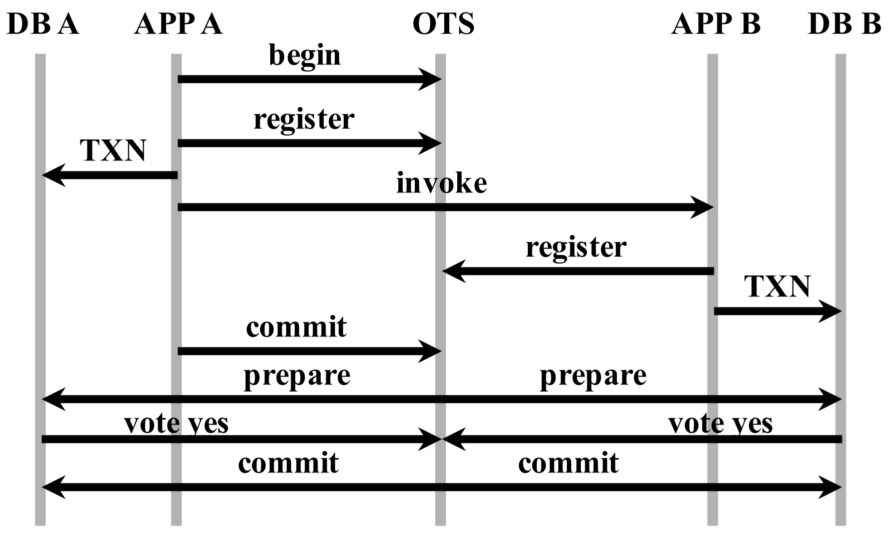
Example 13



Example 14



OTS Sequence of Messages

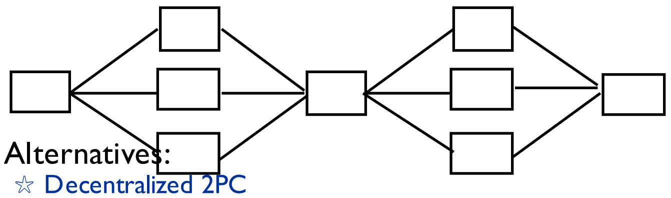


Resource Registration

- ❑ Registration is necessary in order to tell the OTS who will participate in the 2PC protocol and what type of interface is supported. Registration can be manual or automatic
- ❑ Example Interface:
  - ☆ XA supported by many DBS
  - ☆ allows exchange of transaction identifier
    - Hence, the OTS can tell the DBS, which transaction it wants to terminate

Protocol Topologies

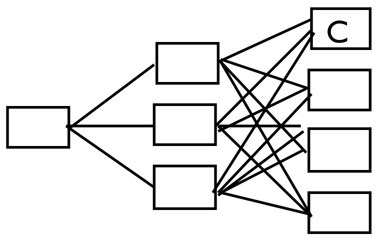
- ❑ Centralized Protocols:
  - ☆ A coordinator manages the protocol flow
  - ☆ Communication is only between coordinator and individual participant
  - ☆ Participants do not need to know each other and do not communicate with each other



- ❑ Alternatives:
  - ☆ Decentralized 2PC
  - ☆ Linear 2PC
  - ☆ Hierarchical 2PC

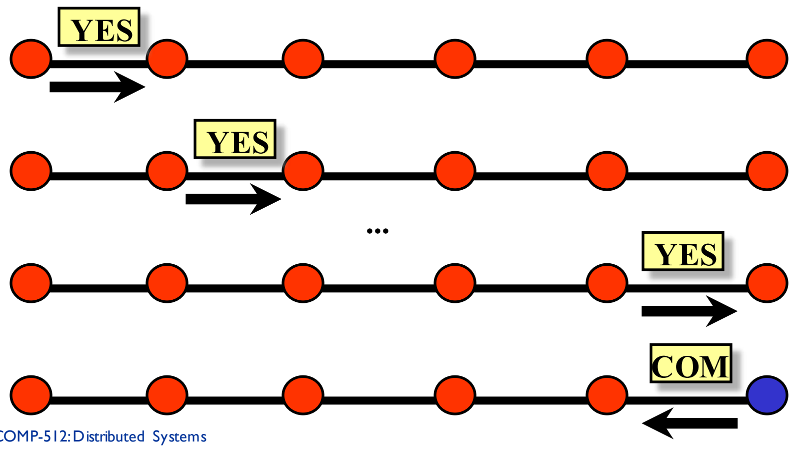
Decentralized 2PC

- ❑ Protocol
  - ☆ Coordinator multicasts YES/NO to all participants (indicates also start of 2PC)
  - ☆ Upon receipt of vote from coordinator, participant multicasts YES/NO vote all other participants and the coordinator
  - ☆ Whenever participant or coordinator has votes from all other sites, terminate transaction accordingly
- ❑ Complexity
  - ☆ Rounds: 2 (coordinator's vote + votes of the others)
  - ☆ Number of messages (n+1 sites):
    - Point-to-point: n * (n+1)
    - Broadcast: n+1



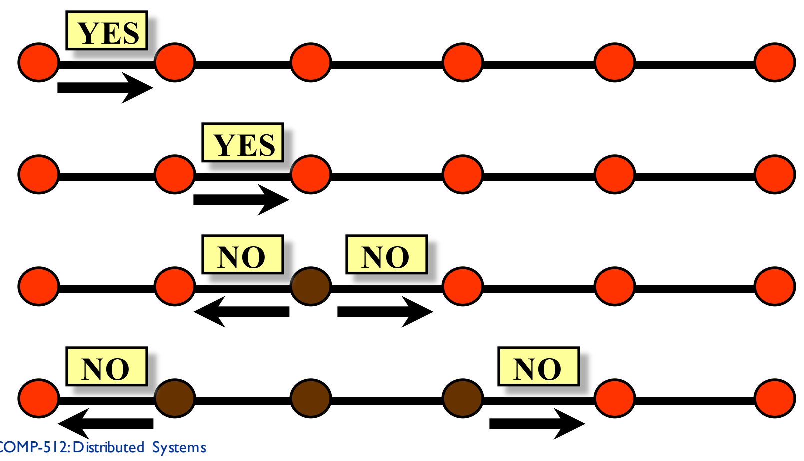
Linear 2PC

- ❑ Linear 2PC commit exploits a particular network configuration to minimize the number of messages:



Linear 2PC

❑ Abort Case



Linear 2PC

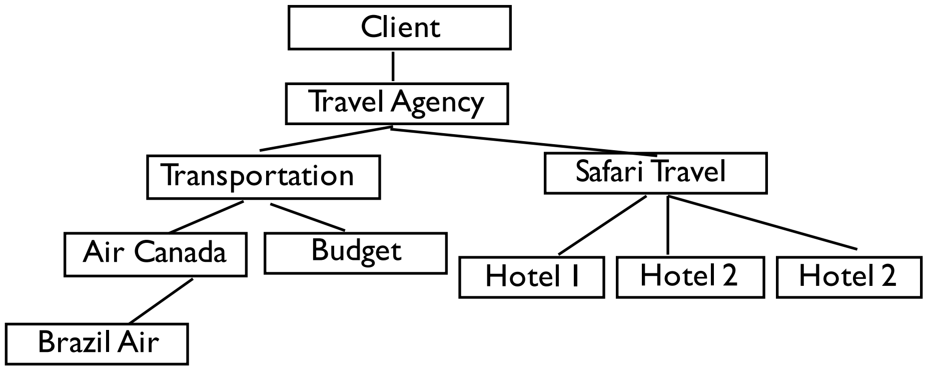
❑ Complexity
  - ☆ Message rounds: 2n-2
  - ☆ Number of messages: 2n-2 (broadcast cannot be exploited)

❑ Often implemented when only two sites

❑ Coordinator delegation: last site takes over to be the coordinator for decision phase
  - ☆ other coordinator delegation protocol implemented in Oracle

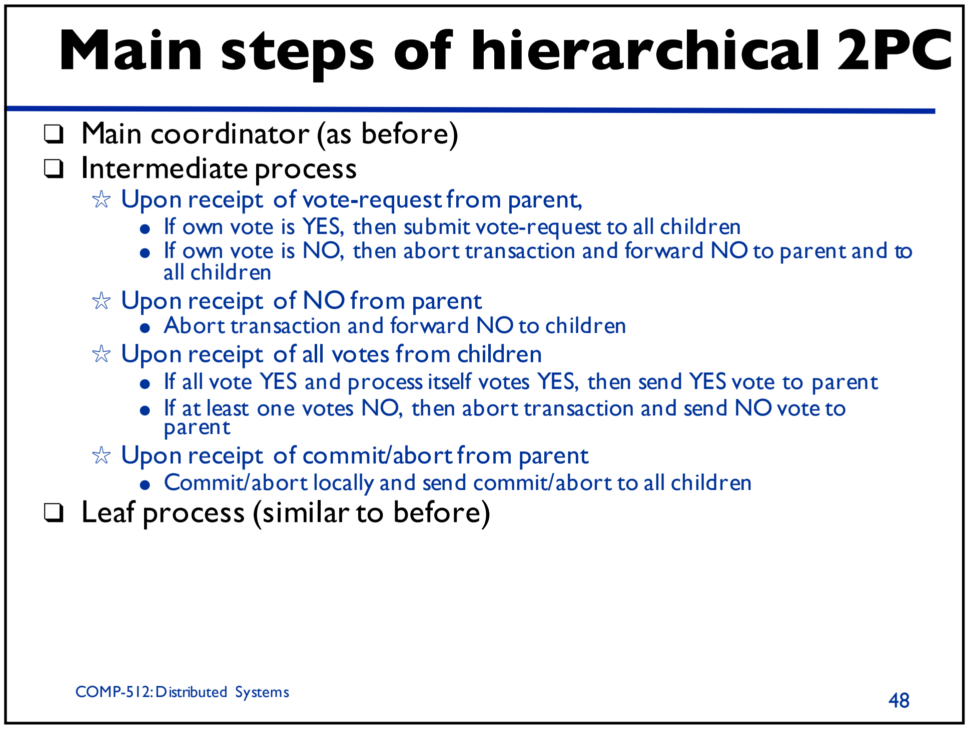
Hierarchical 2PC

- ❑ Truly distributed systems can have hierarchical calling histories
  - ☆ An instance can be server and client at the same time
  - ☆ E-Commerce Applications J2 Enterprise Edition



Hierarchical execution

- ❑ During normal execution
  - ☆ Processes dynamically form tree
  - ☆ New edges are added whenever one processes calls another process to perform a subtransaction
  - ☆ Once a link is created it can be reused for subsequent requests
- ❑ Atomic Commit Protocol
  - ☆ Flattened:
    - Propagation of one global transaction identifier
    - all resources register with the same transaction service
  - ☆ Hierarchical:
    - Root is main coordinator
    - Intermediate nodes are coordinator for children and participant to parent process



Main steps of hierarchical 2PC

- ❑ Main coordinator (as before)
- ❑ Intermediate process
  - ☆ Upon receipt of vote-request from parent,
    - If own vote is YES, then submit vote-request to all children
    - If own vote is NO, then abort transaction and forward NO to parent and to all children
  - ☆ Upon receipt of NO from parent
    - Abort transaction and forward NO to children
  - ☆ Upon receipt of all votes from children
    - If all vote YES and process itself votes YES, then send YES vote to parent
    - If at least one votes NO, then abort transaction and send NO vote to parent
  - ☆ Upon receipt of commit/abort from parent
    - Commit/abort locally and send commit/abort to all children
- ❑ Leaf process (similar to before)