1999 Distributed Systems (Paper 8) solution

p8q1
JMB

a) (i) 2PL: phase of acquiring object locks (no release) then (for strict 2PL) hold until tx committed.
2PL is subject to deadlock so we have to deal
4 with distributed deadlock detection. Algorithms are complex with high overhead - following distributed cycles - so a timeout approach is likely in practice.
(Centralised resource management would be bad - bottleneck/failure

(ii) TSO - distributed ~~d~~ naturally. Each object decides whether to allow or reject an invocation depending
4 on the tx timestamp (ID). Strict TSO - hold up subsequent invocations until commit/abort.

(iii) OCC - take shadow copies of objects - may be literal copies in distributed/user space or versions at objects. Client invokes shadow copies then requests commit. Validation
4 must ensure shadow copies were consistent and that no conflicting updates have been committed at the object since the shadows were taken.

either b) Pessimistic approach - (i) and (ii). The objects are locked awaiting commit. The coordinator carries out a two-phase (or three-phase) commit protocol.
8 (bookwork - expand on:-
1. secure old & new versions in persistent store - coordinator collects commit vote & decide
2. send commit to all participants.).
3. handle any failure.

or b) Optimistic approach. (iii) - expand on:-
objects are not locked.
a central validator checks consistency of
or 8 shadows & conflict of update - run a 2-phase protocol with the objects.
If OK - assignes a timestamp & updates are applied at object in timestamp order. See bookwork for detail

$\boxed{20}$