Orleans Transactions

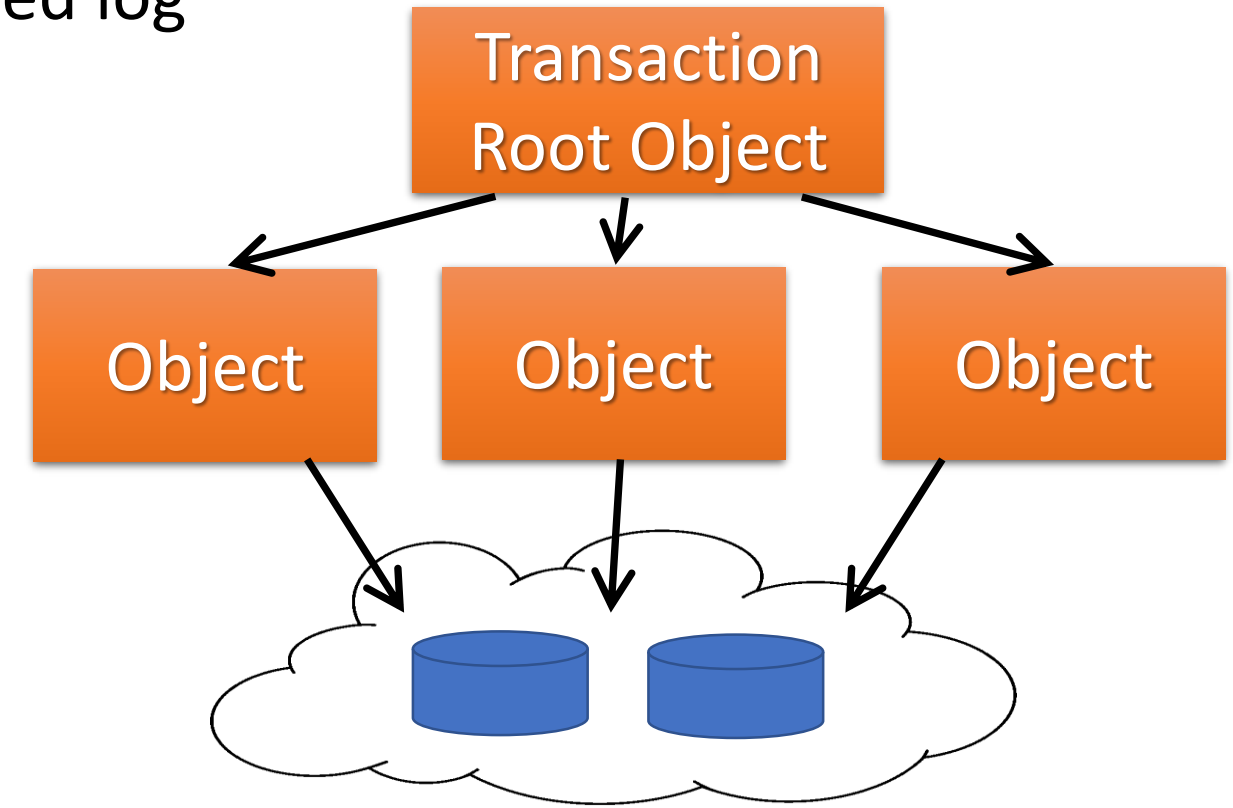
Phil Bernstein, Microsoft

Sebastian Burckhardt, Tamer Eldeeb, Christopher Meicklejohn,
Alejandro Tomsic, Orleans developer team

- Orleans is an actor-oriented programming framework that makes it easy to develop distributed stateful apps that scale out fault-tolerantly

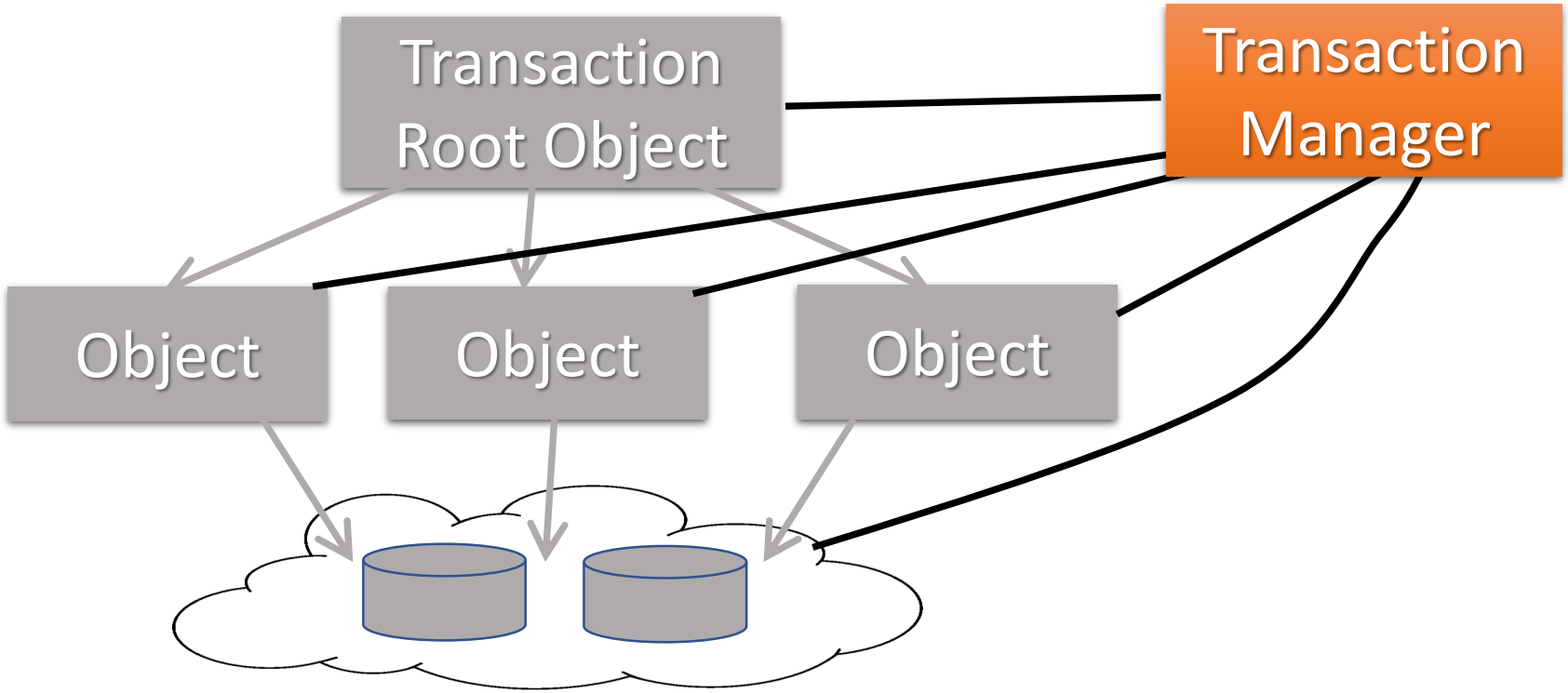
Execution Model

- Transaction executes over distributed stateful objects
  - Objects save state in cloud storage, e.g., key-value store
  - No shared log



Transaction Execution Model

- Object-granularity 2PL
- A centralized transaction manager (TM) runs 2PC
  - One TM per-deployment, with a private log in the cloud



Early Lock Release

- When object *o* receives Prepare, it releases T_1's lock
- If T_2 reads/writes *o*, it takes a "commit dependency" on T_1
  - After T_2 terminates, TM sends Prepare to *o*, which releases lock
  - Now T_3 can read/write *o*, and so on
  - When T_1 finishes Prepare, dependent txns can persist state
  - When T_1 commits, prepared txns that depend on T_1 can commit
- TM commits txns in dependency order
- Benefits:
  - Conflicting transactions execute in parallel with 2PC
  - Enables group commit without a shared log
  - Up to 20x throughput improvement
- But single-object txn must ask TM to validate its dependency

Solution: Per-object TM

- Single-object txns resolve dependencies locally

- Other benefits
  - No central TM bottleneck or point-of-failure
  - TM's are naturally geo-distributed, with the objects
  - Less configuration complexity

Status

- α-release of centralized TM with early-lock-release
- Per-object TM coming soon
- http://dotnet.github.io/orleans