A Scalable Architecture for Maximizing Concurrency

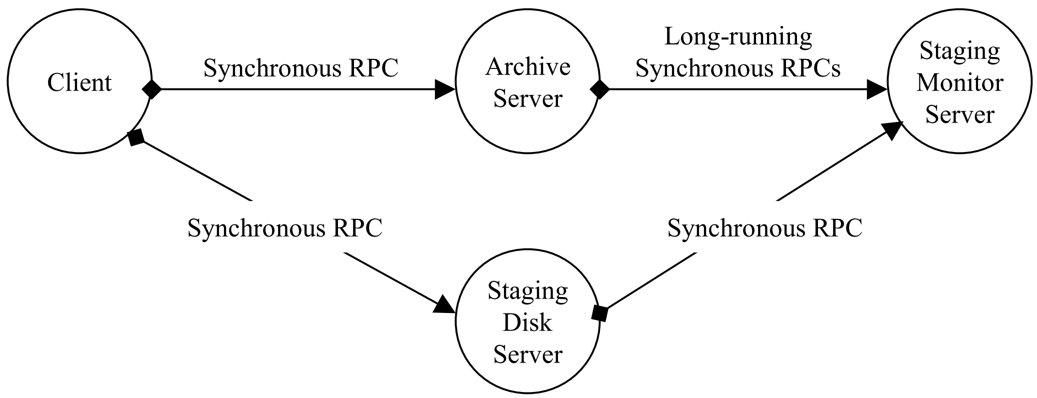
Jonathan Crawford

Lockheed Martin Corporation

IEEE Mass Storage & Technology Conference

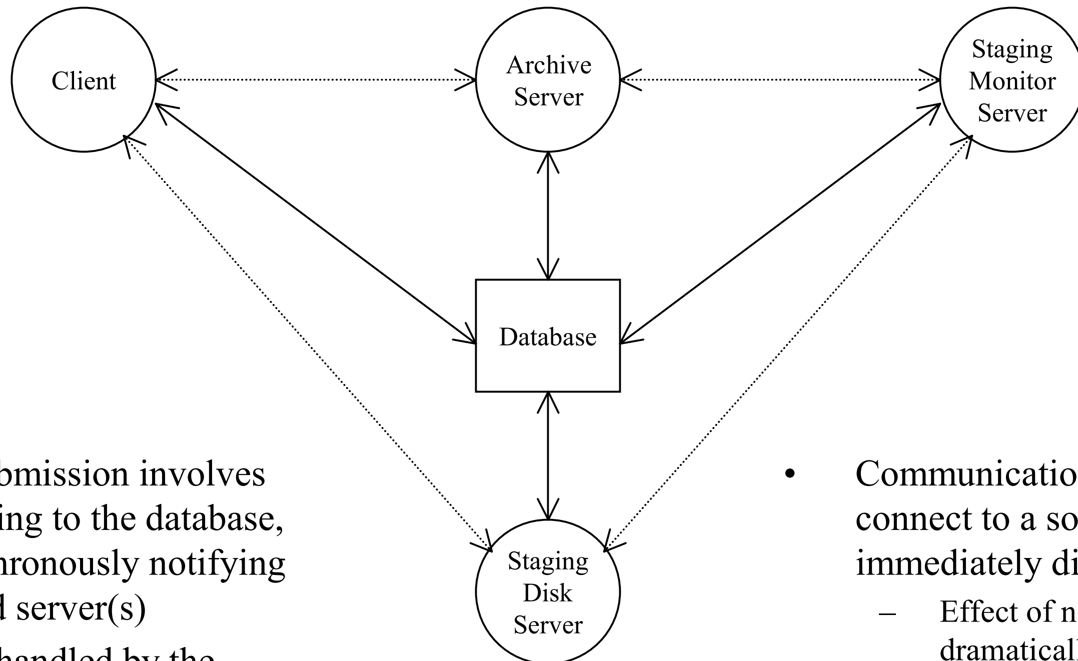
March 28, 2000

DCE-Based Client Interface



- Chained RPCs destabilize all participants
  - A single crash or network disruption triggers recovery in many processes
- Long-running RPCs increase risk and exposure to transient network disruptions
- Architecture heavily reliant on DCE -- not a stable platform under load
- Client communicates with many servers, including multiple instances of the same server type
  - Increases burden of recovery on client
  - Client must notify all servers in event of crash recovery or risk leaking resources

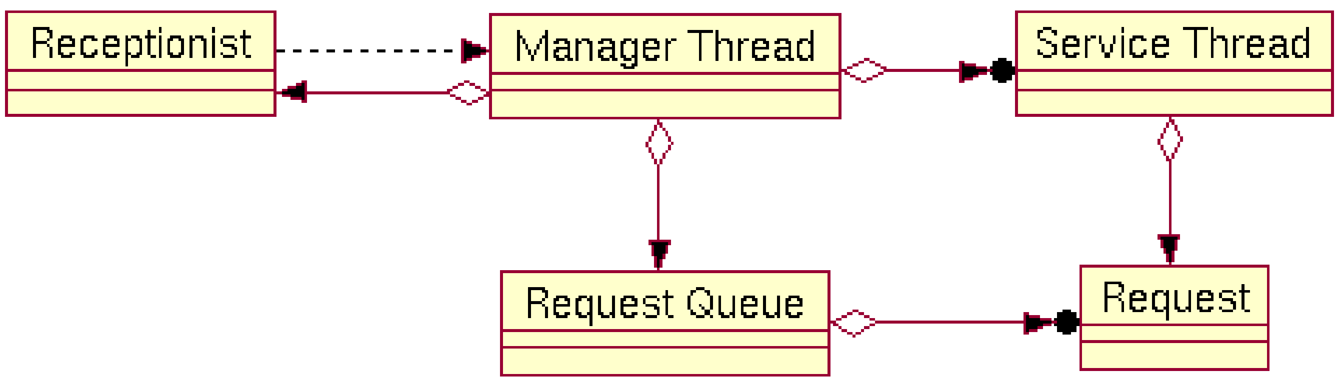
Asynchronous Client Interface



- Request submission involves checkpointing to the database, then asynchronously notifying the affected server(s)
- Routing is handled by the database and base class implementation for asynchronous communications
  - Client does not need to know which servers are contacted

- Communications are trivial -- connect to a socket and immediately disconnect
  - Effect of network disruption is dramatically reduced
- Crashes are isolated
- Architecture is based upon stable, proven RDBMS technology

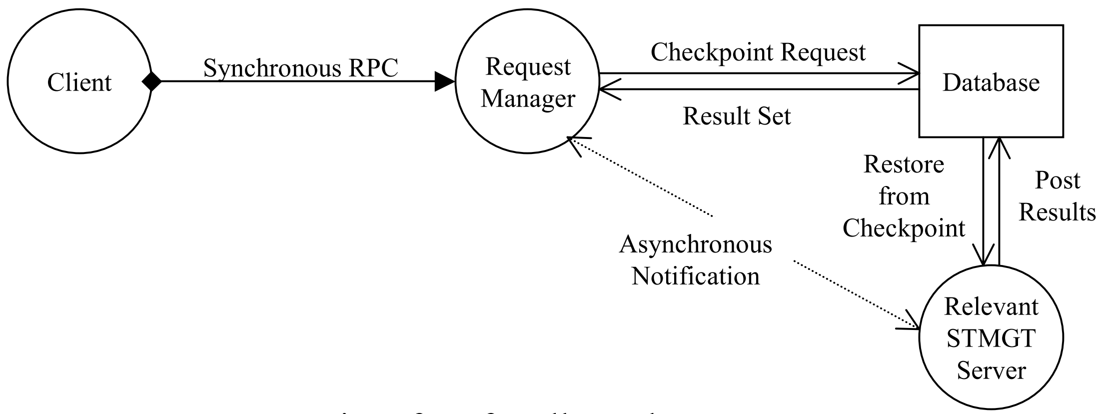
Thread Manager Design Pattern



DCE Limitations

- RPC-based requests arrive on “listen threads”
  - DCE queue is limited to 7 times the number of “listen threads”
  - Queue overflow causes client failures
- DCE has stability issues with heavy bursts of requests to the same server
  - Common in recovery situations
- Difficult to set up development environments
  - Stubs must be provided to simulate client interfaces
  - CDS entry management requires higher-level access than is customarily granted to developers

Request Manager Adapter



- Request Manager serves as proxy interface for all synchronous communications
- Acts as an asynchronous client on behalf of each synchronous client
- Allows all existing client interfaces to be preserved
- Isolates client communications to a single server
  - Simplifies failover, crash recovery
- Network disruptions limited to the Client/Request Manager interface

Scalability

- Multiple Request Managers may be used, either for "closest access" routing or for automatic failover
- Scalability is limited only by database scalability
  - Configuration parameters and checkpointed request information are only things stored in database, and both are typically small
- Routing is handled internally, so new servers and subsystems can be rapidly added without client changes

Reliability

- Only potential "single point of failure" is the database
  - RDBMS failover and recovery technology is mature
- Clients are no longer affected by temporary server outages
- Simplicity of communications mechanism -- port number on a host -- minimizes complexity and eliminates configuration issues associated with DCE
- Brevity of communications minimizes impact of network disruptions, especially transient interruptions
- Design forces emphasis on good checkpointing and recovery for each server

Testability

- Overhead for testing
  - DCE: Requires CDS entries (access issues) and other servers must be stubbed or operational
  - Thread Manager: Requires only a test database
- Client simulation
  - DCE: drivers are synchronous, typically designed for one-at-a-time request submission
  - Thread Manager: completely data-driven via SQL
    - Simple SQL scripts support load testing with a universal driver to start test
    - Scripting facilities and trigger-based capture of stored procedure invocations permit scenario capture for simulating end-to-end testing and rapid reproducing of obscure error conditions

Operability

- Request status is uniformly available for all requests in a central table
- Progress tracking is incorporated in the design
- Multiple states provide clear visibility into server and request status
  - Processing state indicates whether a request is active, suspended, or completed
  - Checkpoint state indicates where a request is in processing
  - Error code indicates error associated with request, including severity and scope of error