Lecture 16: Cache and Replica Coherence

16.0 Main Points

Motivation
Polling
Timeouts
Callbacks
Replicas

16.1 Motivation

Earlier in course, said two problems with treating RPC as transparent: failures and performance.

Can use caches to improve performance -- keep copy of data close to client.

Can use replication to improve performance and availability -- client can go to any replica.

Problem is updates: how do we keep information in caches up to date?
How do we keep replicas consistent?

Example: Domain Naming System (DNS), uses both caching and replication.

Hierarchical directory based naming (ex: cs.washington.edu)

16.2 Cache Coherence

Try #1: user manages cache

Try #2: timeouts

clients poll server periodically to check if file has changed.

Try #3: callbacks

Server keeps track of who has what cached. Sends (reliable!) message to invalidate all cached copies. Clients refetch new data.

16.3 Replication

Updates go to all replicas (eventually)

Reads can go to any replica

Replication improves performance , as long as most operations are reads -- work is split among all replicas.

Replication improves availability -- reads can occur even if some servers are down.