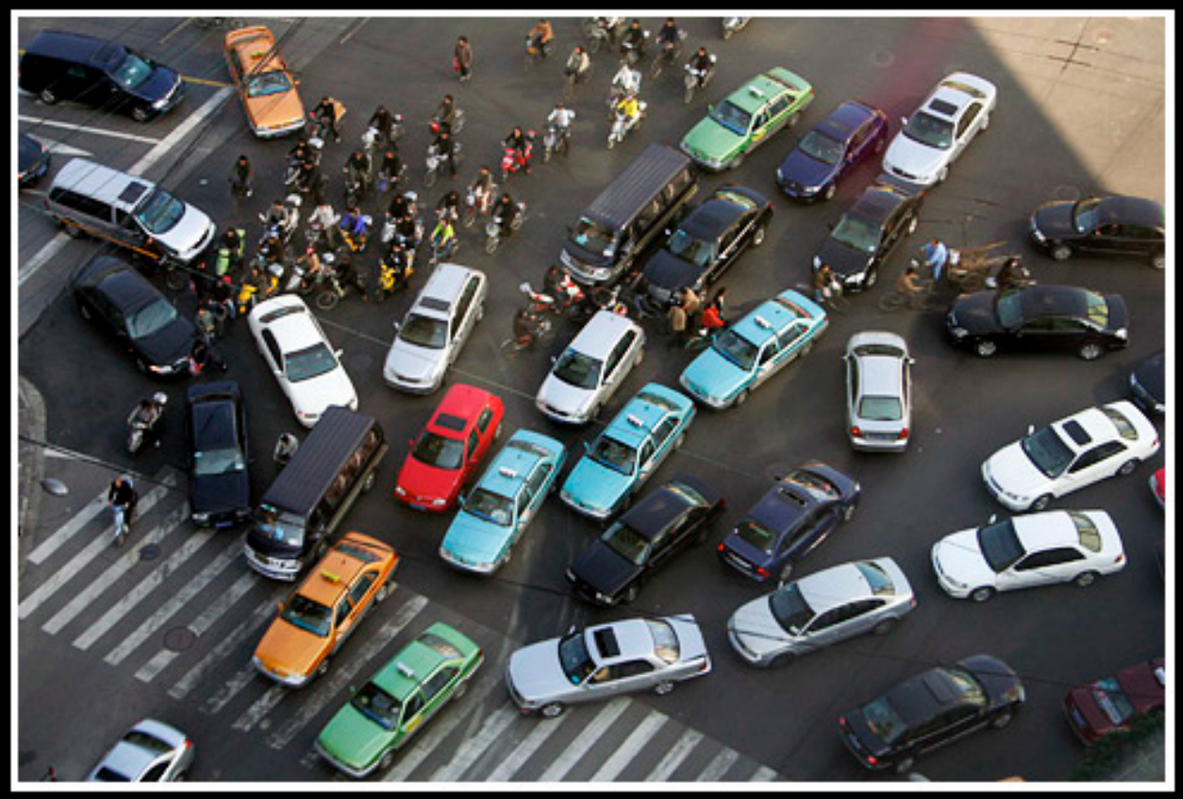
Coordenação Distribuída

Flavio Junqueira
Yahoo! Research, Barcelona

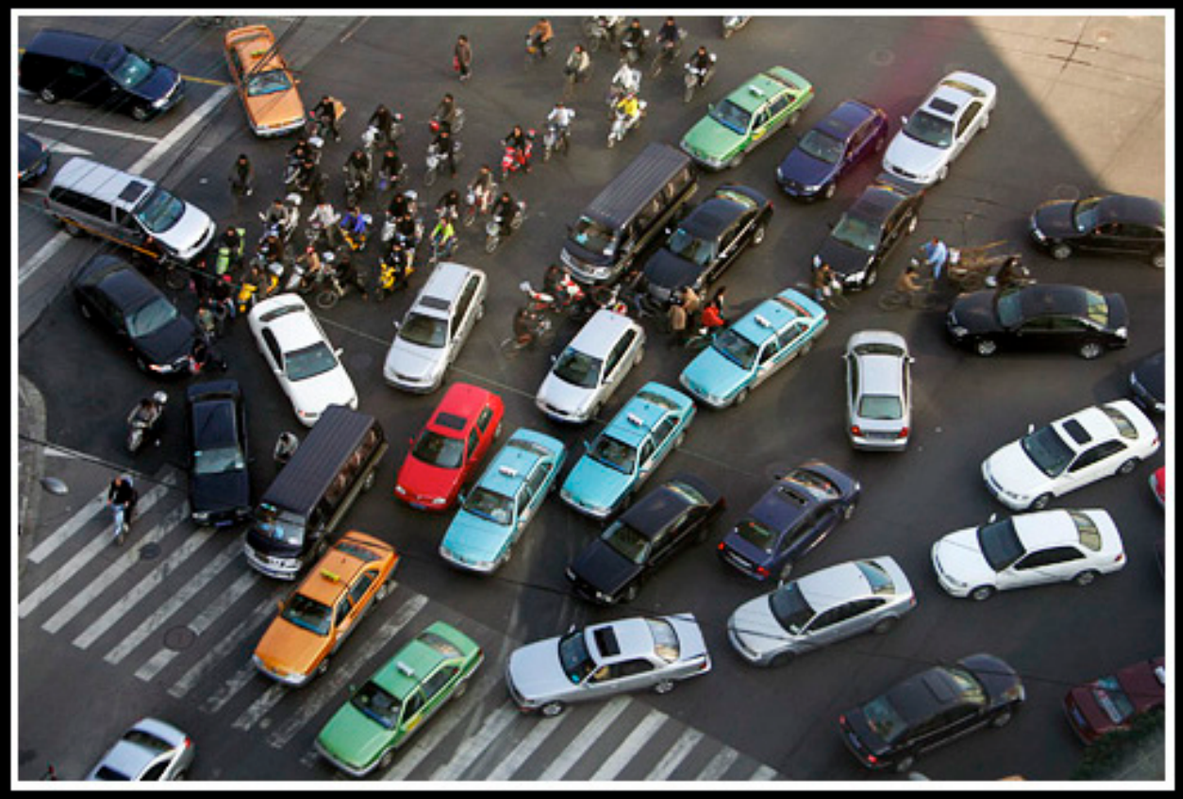
INFORUM 2010

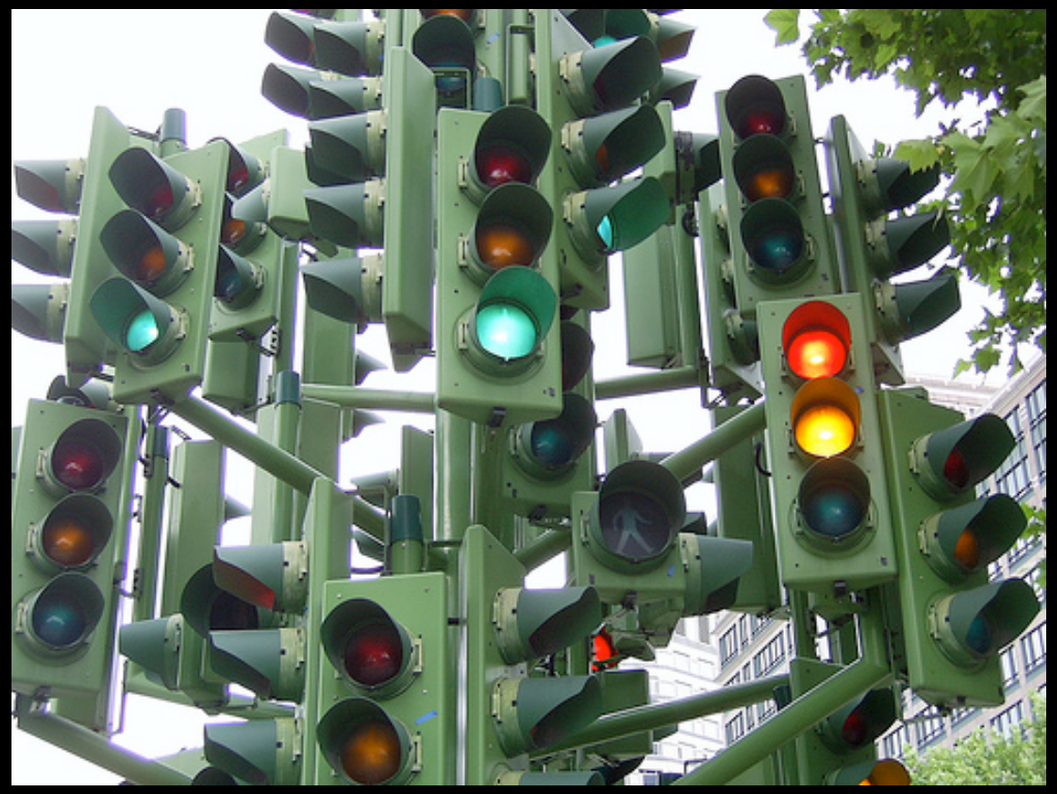
Coordenar é importante

Coordenar é importante

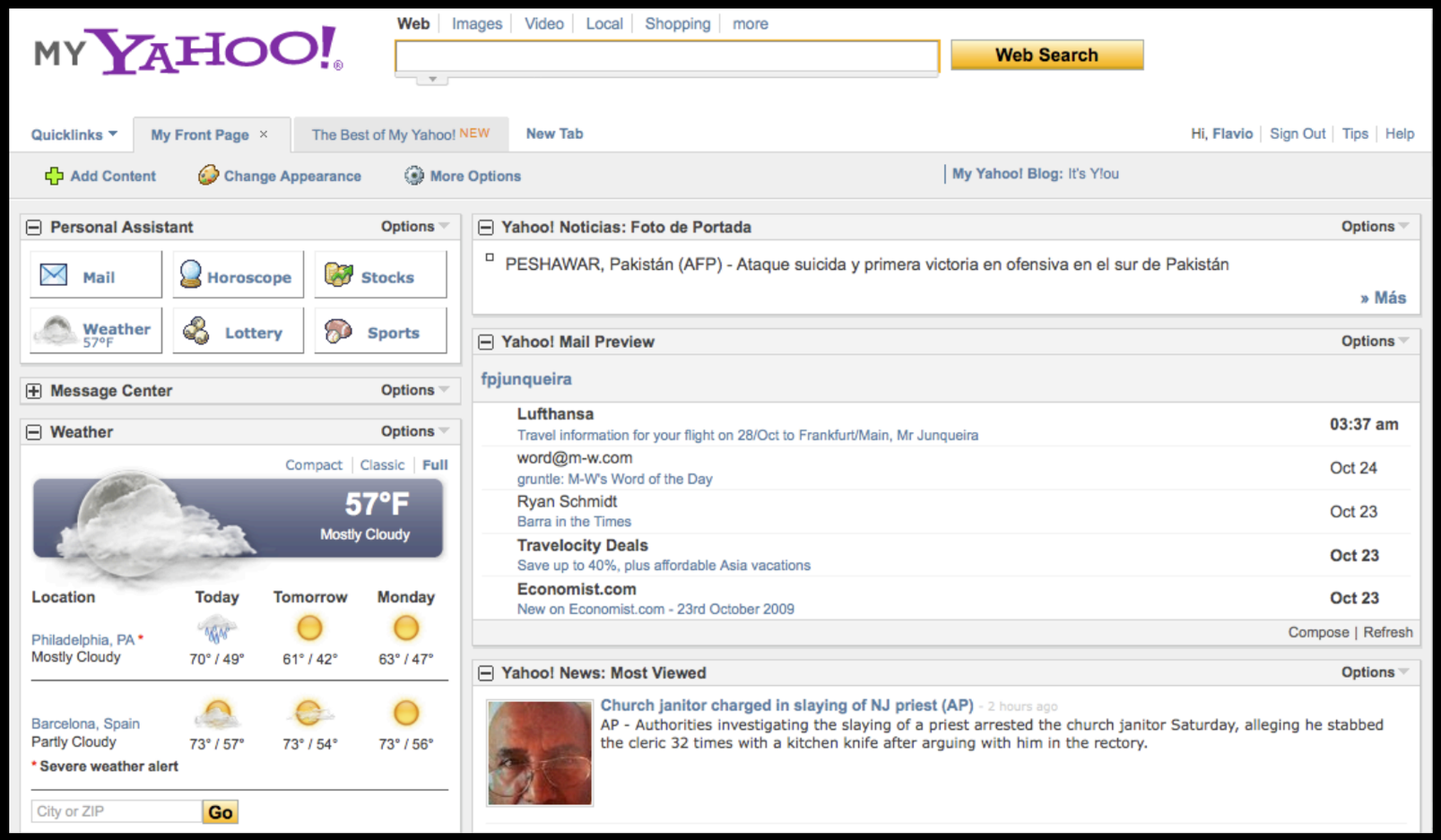


Coordenar é importante

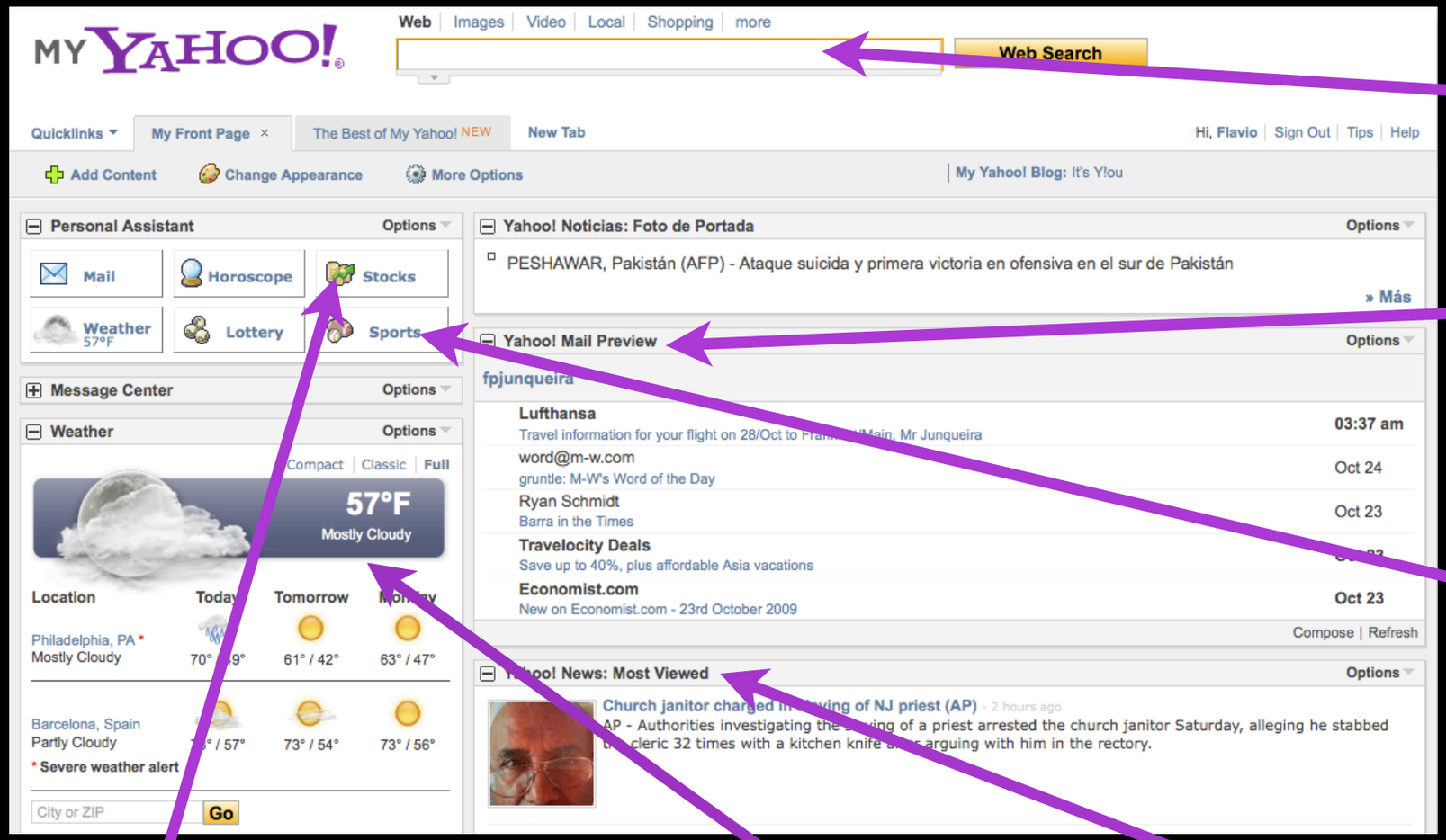




Portal da Yahoo!



Portal da Yahoo!



Search

Mail

Sports

Finance

Weather

News

Portal da Yahoo!

- Home page:
  - ✓ 38 milhões de usuários por dia (EUA)
  - ✓ 2.5 bilhões de visitas por mês (EUA)
- Search:
  - ✓ +3 bilhões de consultas Web
- Mail:
  - ✓ +90 milhões de usuários
  - ✓ +10 min/visita

Fontes: Yahoo! e comScore

Infra-estrutura

- Muitos servidores
- Muitos processos
- Grande volume de dados
- Alta complexidade dos sistemas de software
- ... engenheiros de software são meros mortais

Sistemas da Web

- Sistemas que requerem coordenação
  - ✓ Em geral
    - ➡ Grandes conjuntos de dados em servidores convencionais
  - ✓ Exemplos
    - ➡ Sistemas de arquivos distribuídos
    - ➡ Sistemas para o processamento de grandes volumes de dados

O que é a coordenação distribuída?

- Técnicas para coordenação de processos

✓ Metadados (configuração)

✓ Eleição de líder

✓ *Locks* distribuídos, barreiras, filas

Onde se utiliza?

- Sistemas de arquivo distribuídos: HDFS, GFS
- Sistemas de armazenamento: Bigtable, HBase
- Motores de busca: *Crawling*, Índice

Google File System (GFS)

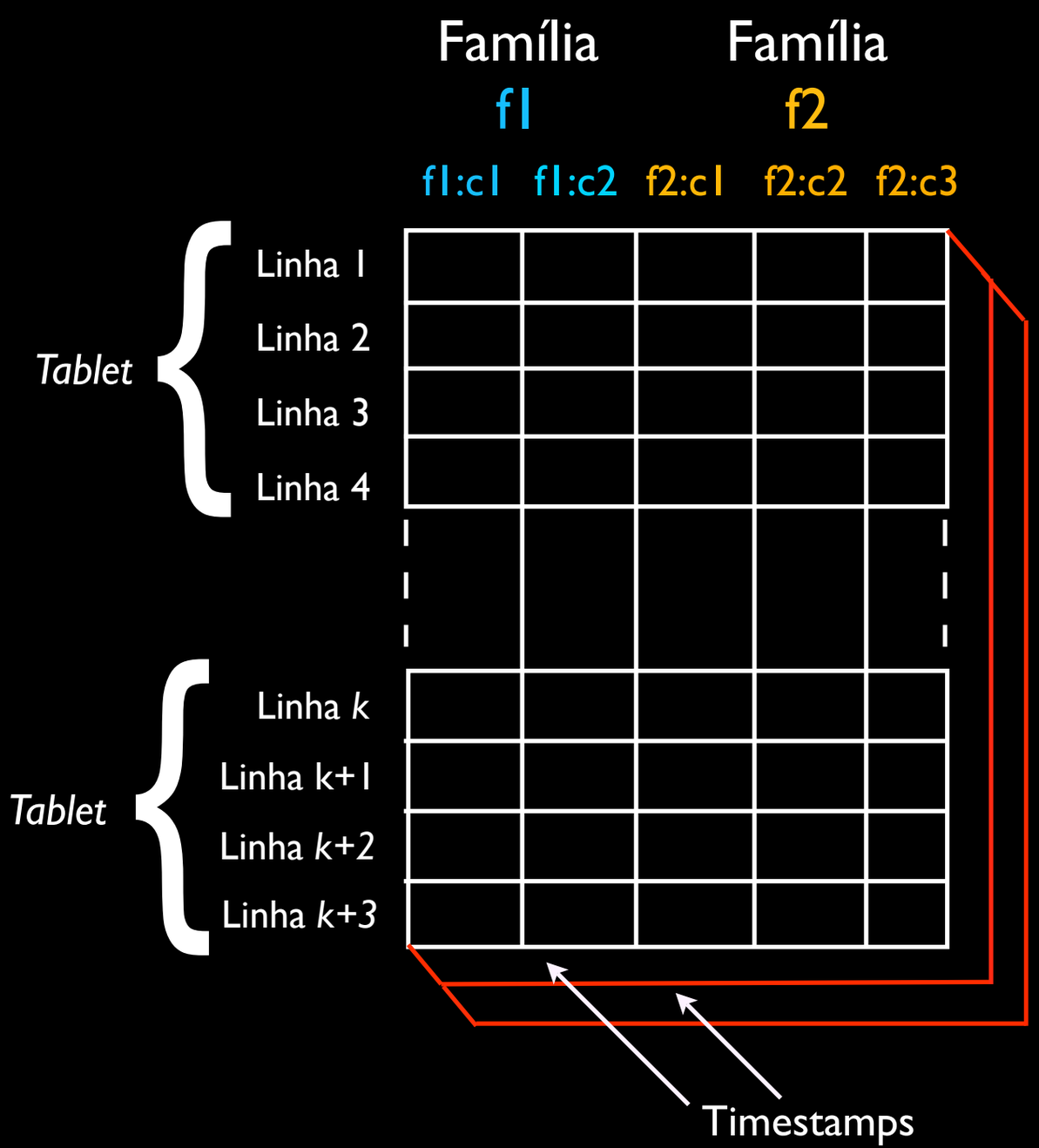
- Sistema de arquivos escalável
- Aplicações com uso intensivo de dados
- Cluster GFS
  - ✓ Um mestre
  - ✓ Múltiplos servidores de *chunks*

Google File System (GFS)

- Eleição do mestre
  - ✓ Assegurar que a cada momento existe no máximo um mestre ativo

Bigtable da Google

- Tabelas de dados
  - ✓ *Tablet* = unidade de distribuição
  - ✓ Família de colunas = unidade de controle de acesso
- Arquitetura
  - ✓ Um mestre
  - ✓ Servidores de *tablets*

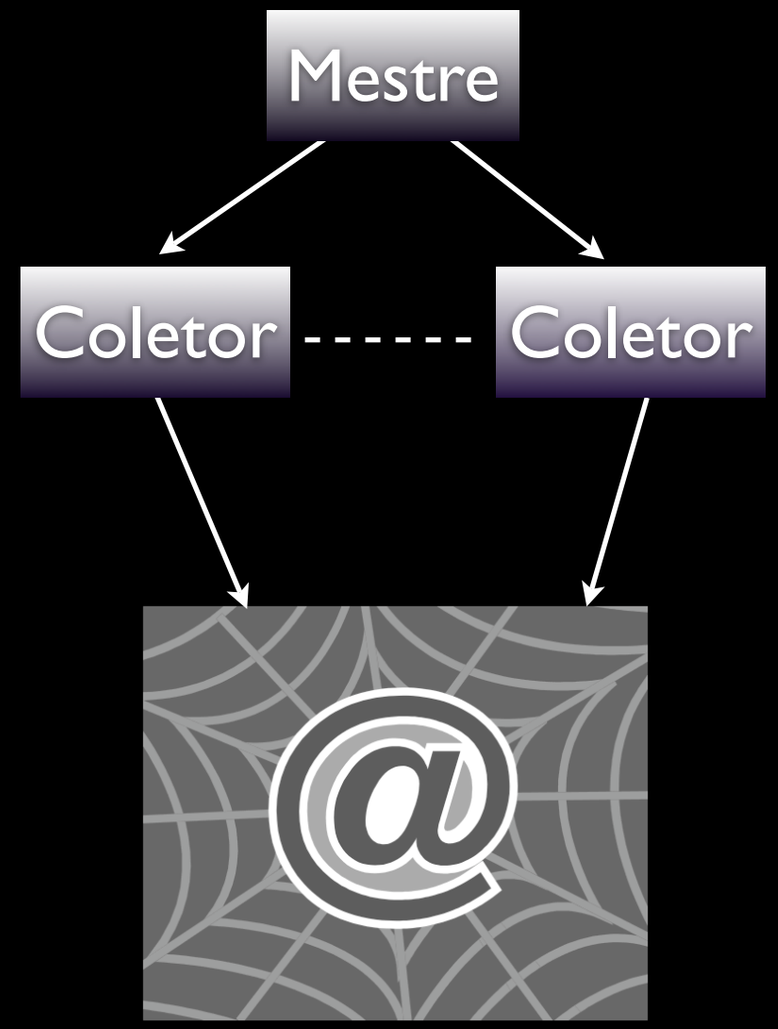


Bigtable da Google

- Eleição do mestre
  - ✓ Assegurar que existe no máximo um mestre ativo
- Metadados
  - ✓ Armazenar o esquema e as listas de controle de acesso (ACLs)
- Encontro (*rendezvous*)
  - ✓ Descobrir servidor de *tablet*
- Detecção de falhas
  - ✓ Servidores de *tablet* disponíveis

Crawler da Yahoo!

- Serviço de coleta
  - ✓ Páginas da Web
  - ✓ Coleta = copia de documentos
- Milhares de servidores
  - ✓ Documentos da Web coletados concorrentemente
  - ✓ Mestre comanda os coletores
  - ✓ Coletores recolhem documentos

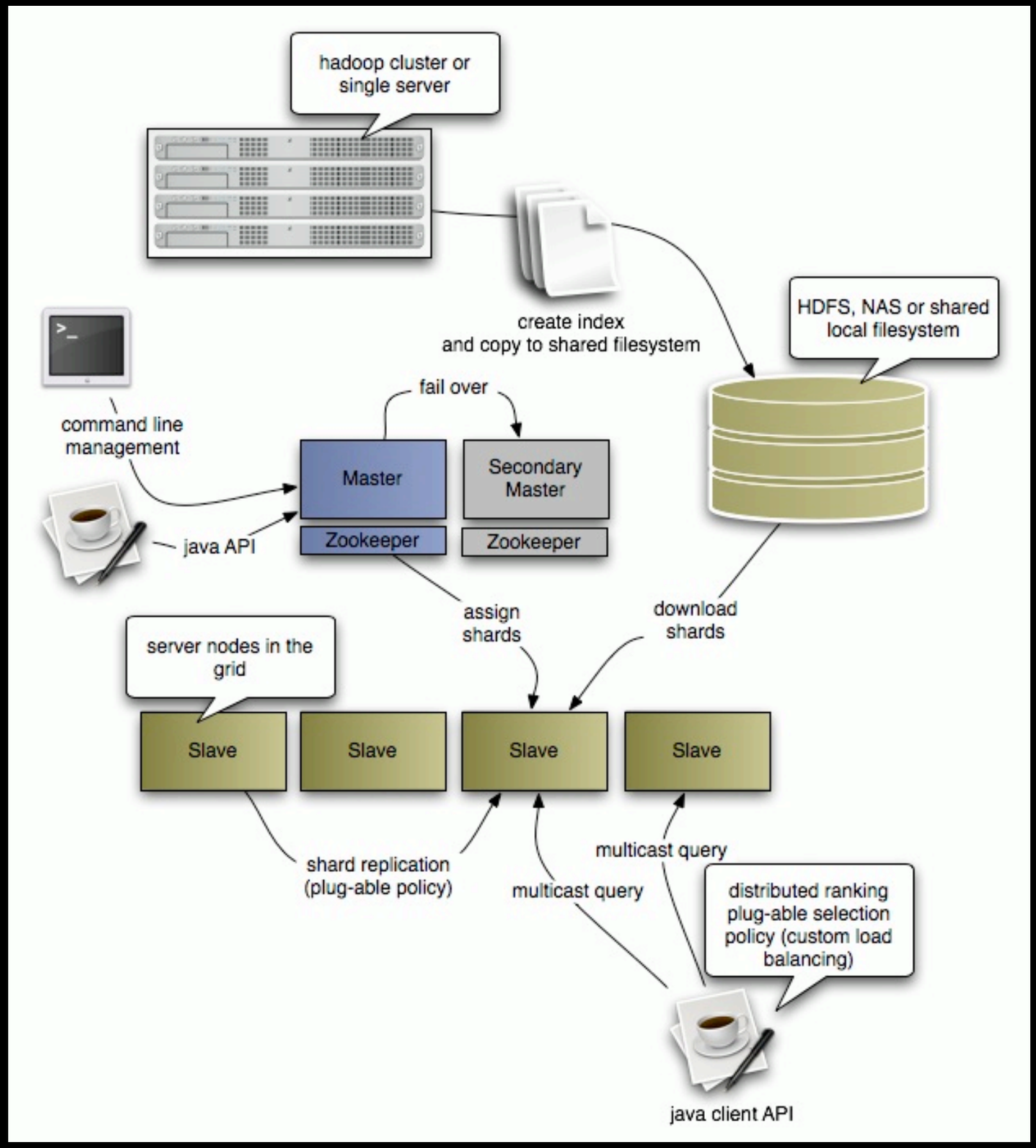


Crawler da Yahoo!

- Eleição do mestre
  - ✓ Processo mestre que lidera o grupo
- Designação de trabalho (configuração)
  - ✓ Páginas a coletar, *politeness*, *etc*.
- Detecção de falhas
  - ✓ Coletores disponíveis

Katta

- ✓Coordenação
  - Eleição do mestre
  - Detecção de falhas
  - Avisos, *etc.*



http://katta.sourceforge.net/

Sistemas para coordenação

- Motivação
- Implementação de primitivas de coordenação não é trivial
  - ✓ Envolve algoritmos distribuídos complexos...
  - ✓ ... complexo no sentido de racionalizar.
- Não é o principal objetivo de muitos projetos
  - ✓ Frequentemente mal desenhados ou implementados
  - ✓ Duplicação

Sistemas para coordenação

- Chubby, Google [*USENIX OSDI 2006*]
  - ✓ Serviço de *locks*
- Centrifuge, Microsoft [*USENIX NSDI* 2010]
  - ✓ Serviço de leases
- ZooKeeper, Yahoo! [*Hunt* et al. *USENIX ATC 2010*]
  - ✓ Núcleo para coordenação
  - ✓ Código aberto (Apache, desde 2008)

Sistemas para coordenação

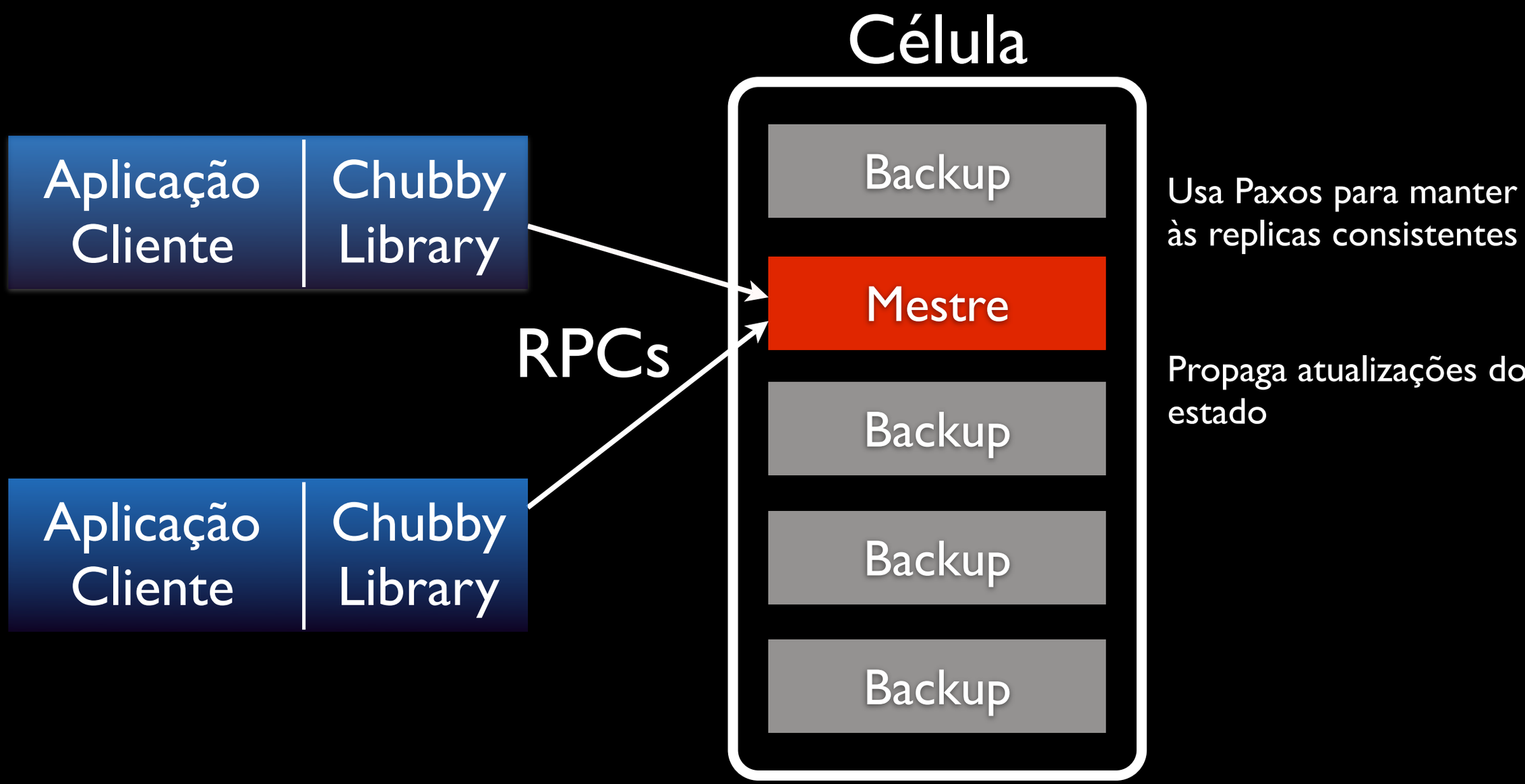
- Chubby, Google [*USENIX OSDI 2006*]
  - ✓ Serviço de *locks*
- Centrifuge, Microsoft [*USENIX NSDI* 2010]
  - ✓ Serviço de leases
- ZooKeeper, Yahoo! [*Hunt* et al. *USENIX ATC 2010*]
  - ✓ Núcleo para coordenação
  - ✓ Código aberto (Apache, desde 2008)

Chubby

Chubby: Resumo

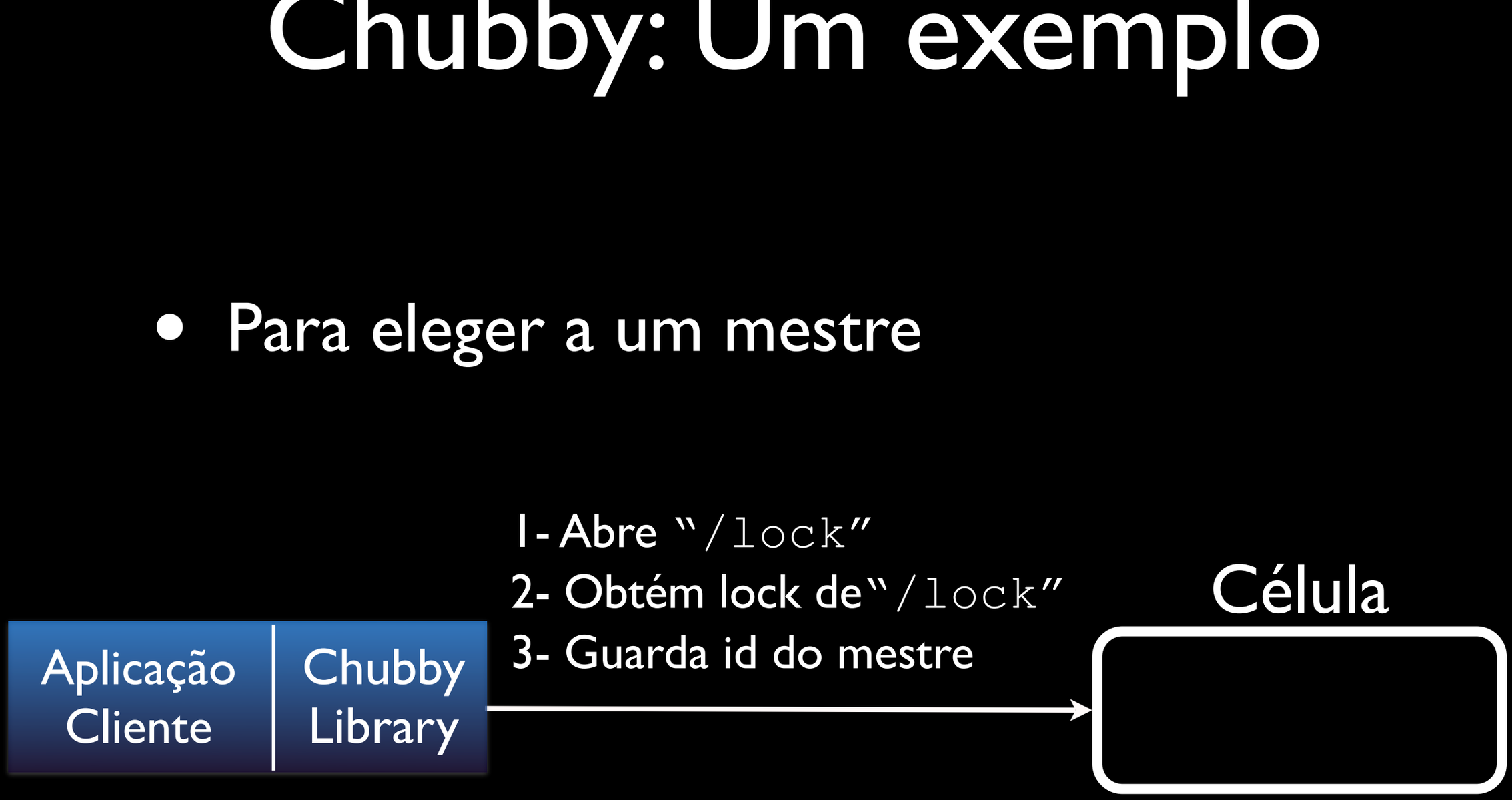
- API semelhante a de um sistema de arquivos
- Arquivos de dados estruturados em árvore
- Locks
- Operações para abrir e fechar arquivos
- Invalidação da cache do cliente
- Operações linearizáveis

Chubby: Design



Chubby: Um exemplo

- Para eleger a um mestre



Chubby: Invalidação da Cache

Modificações são bloqueadas enquanto houver invalidações pendentes

Cliente 1

Cache

Escreve em "/A"

Célula

Cliente 2: /A

Cliente 3: /A

Cliente 2

Cache /A

Invalida "/A"

Cliente 3

Cache /A

Invalida "/A"

Chubby: Linearizabilidade

- Operações linearizáveis
  - ✓ Sequenciável + ordem temporal de precedência

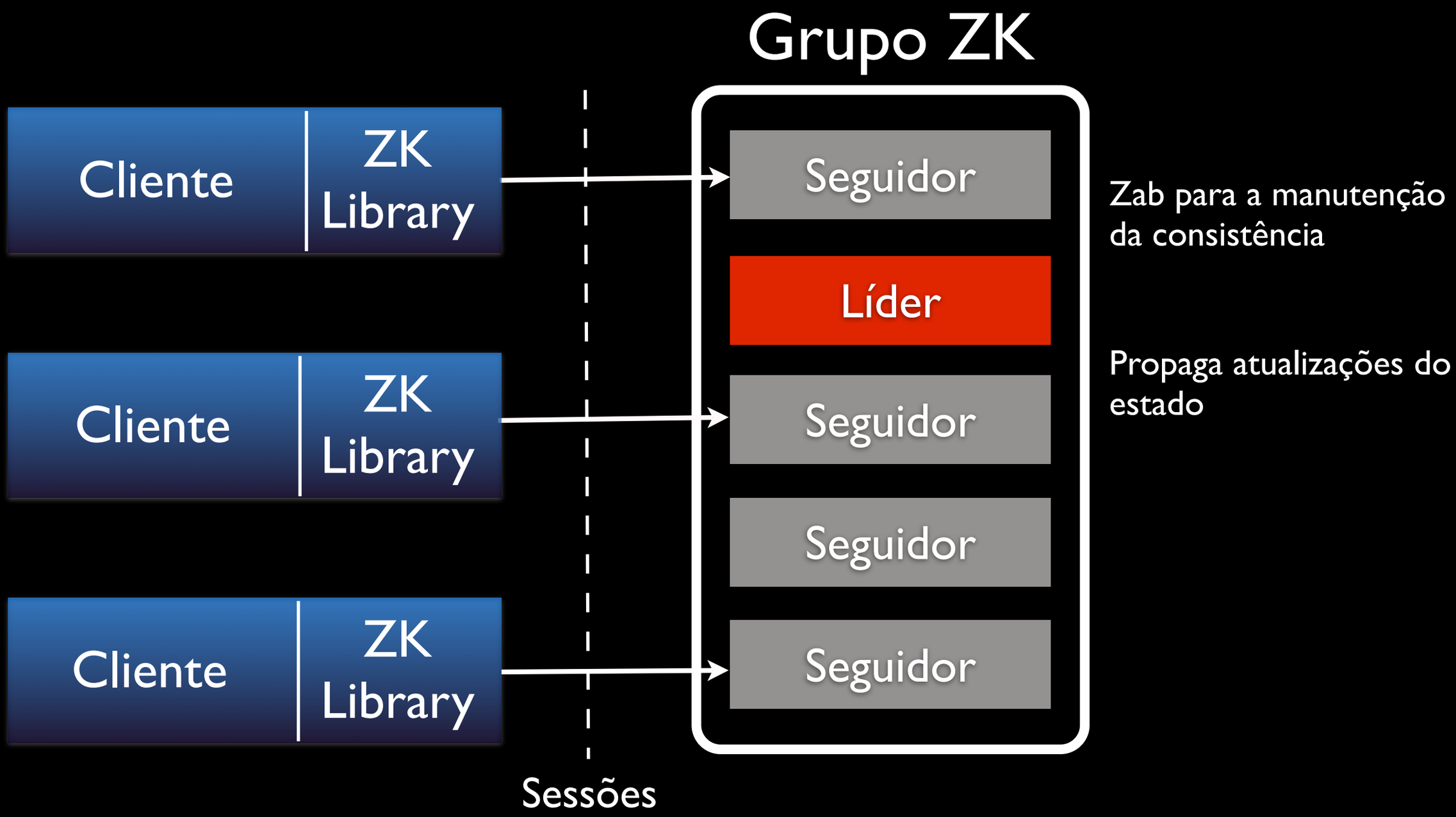
- Mestre de uma célula
  - ✓ Inicia todas as operações
  - ✓ Processa a todas as operações em ordem
  - ✓ Paxos: uma única proposta pendente por vez

ZooKeeper

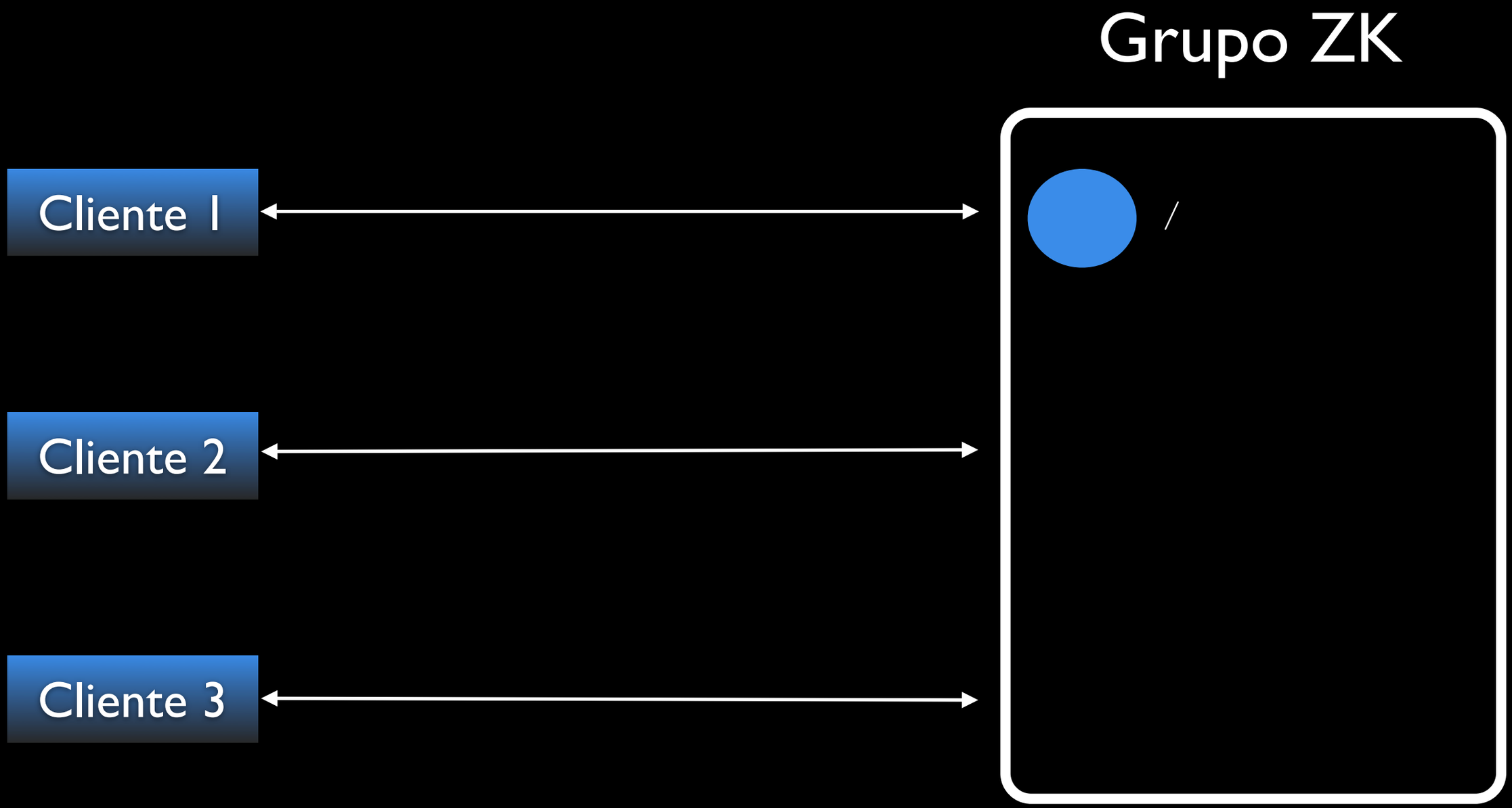
ZooKeeper: Resumo

- Núcleo de coordenação
- Estrutura de *znodes* em árvore
- Receitas para a implementação de primitivas
- Gestão da cache de clientes delegada
  - ✓ Uso de *watches*
- Não há operações para abrir e fechar znodes
- Escritas são linearizáveis

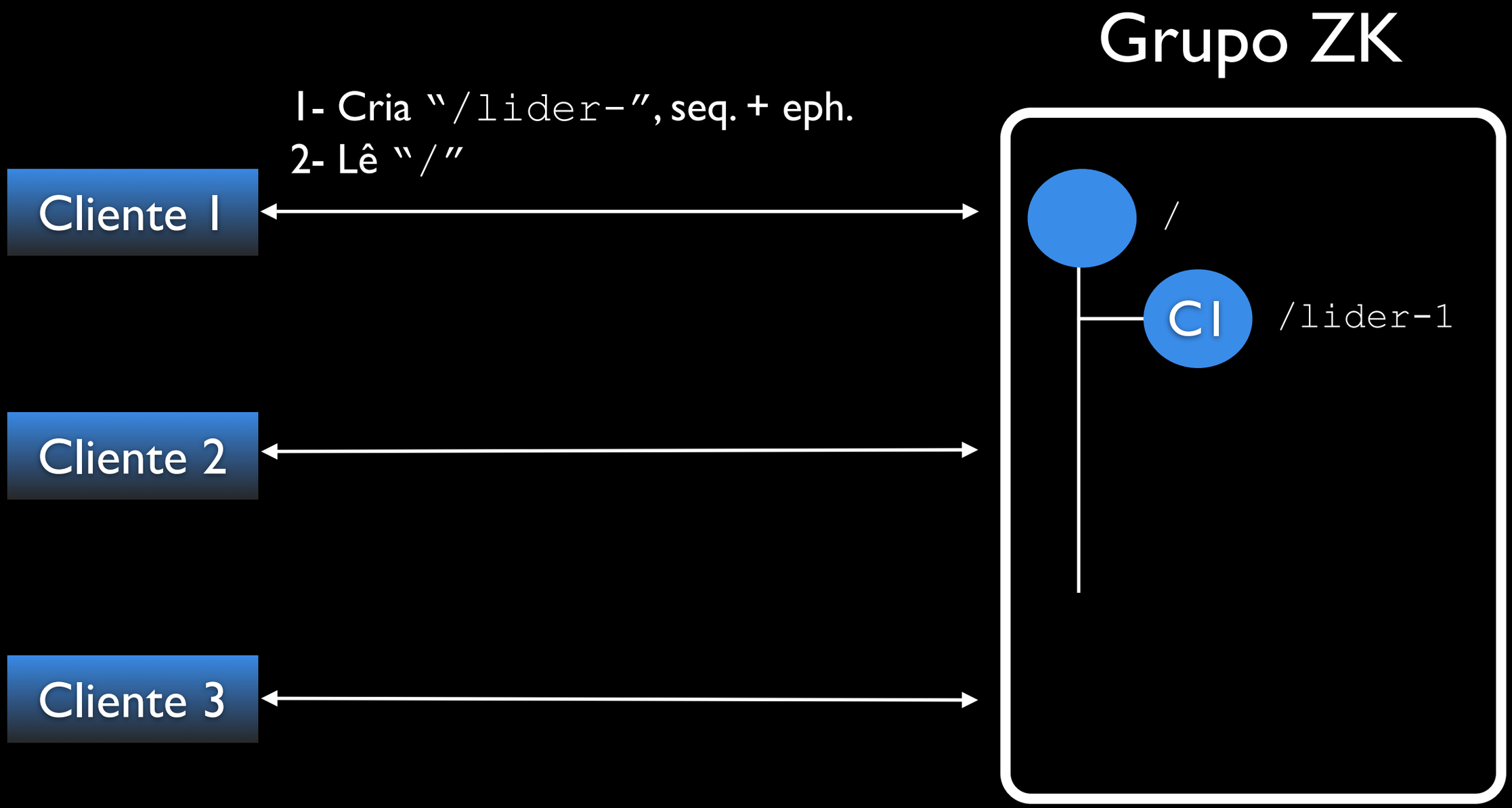
ZooKeeper: Design



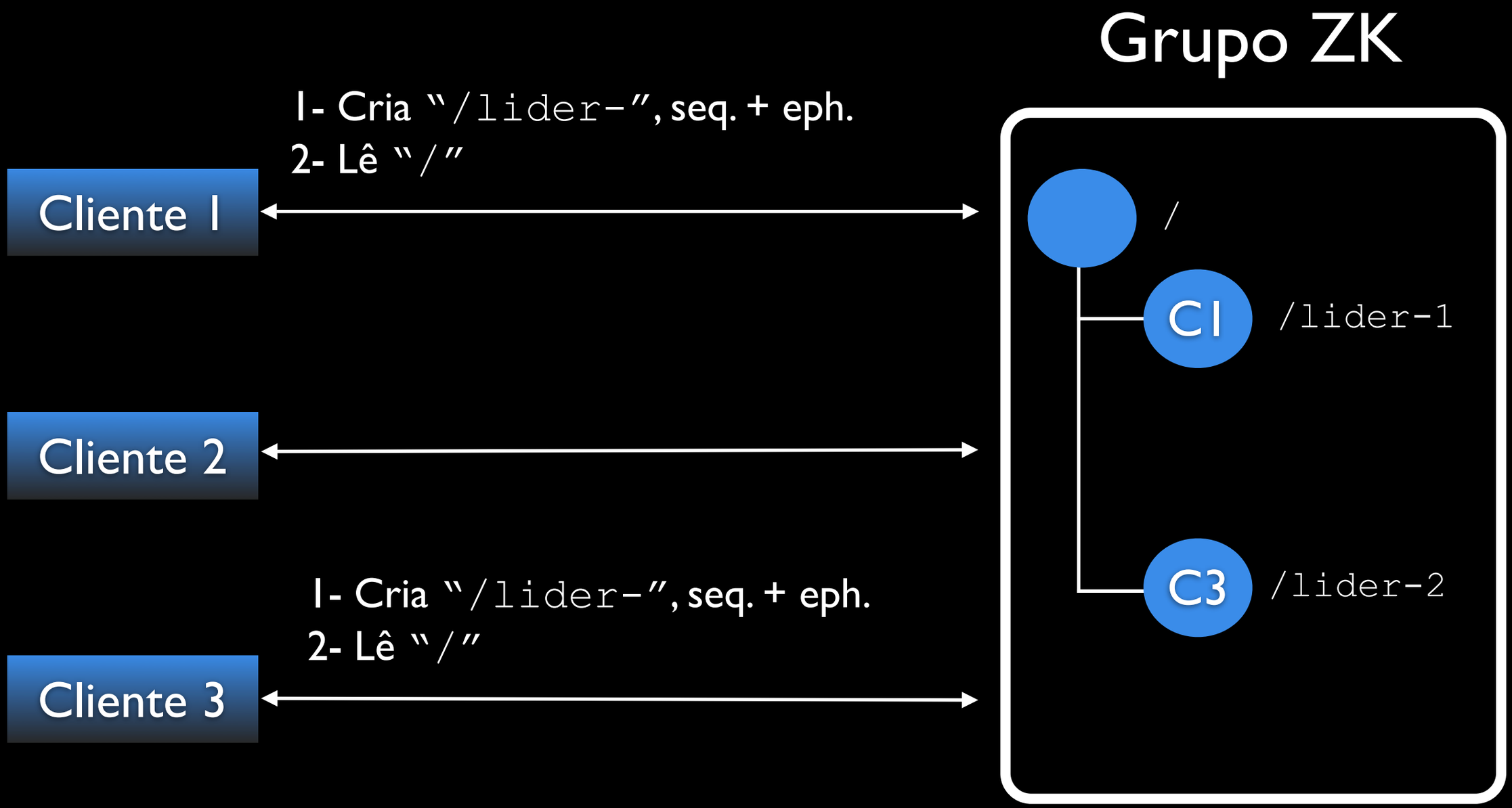
ZooKeeper: Exemplo



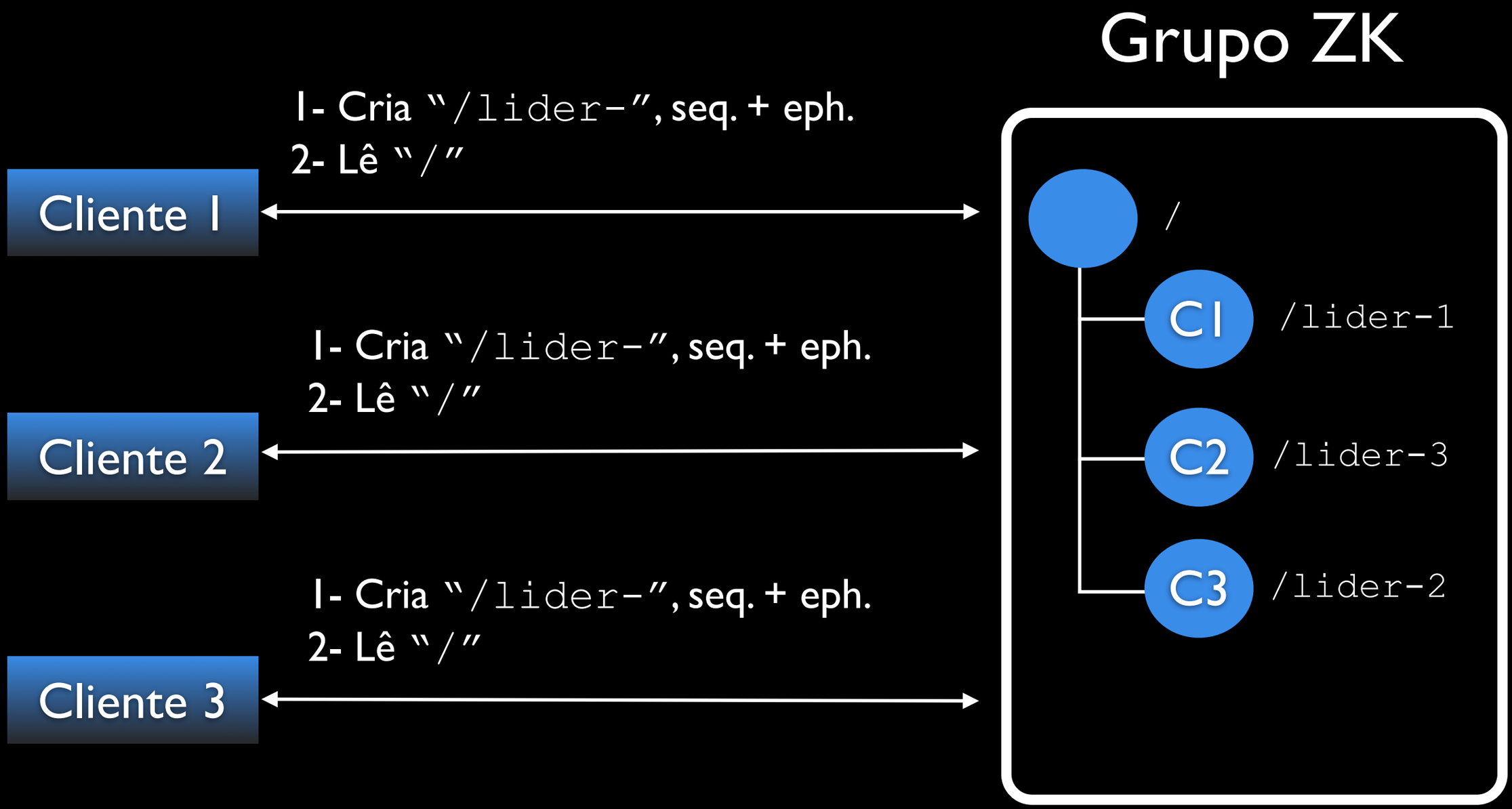
ZooKeeper: Exemplo



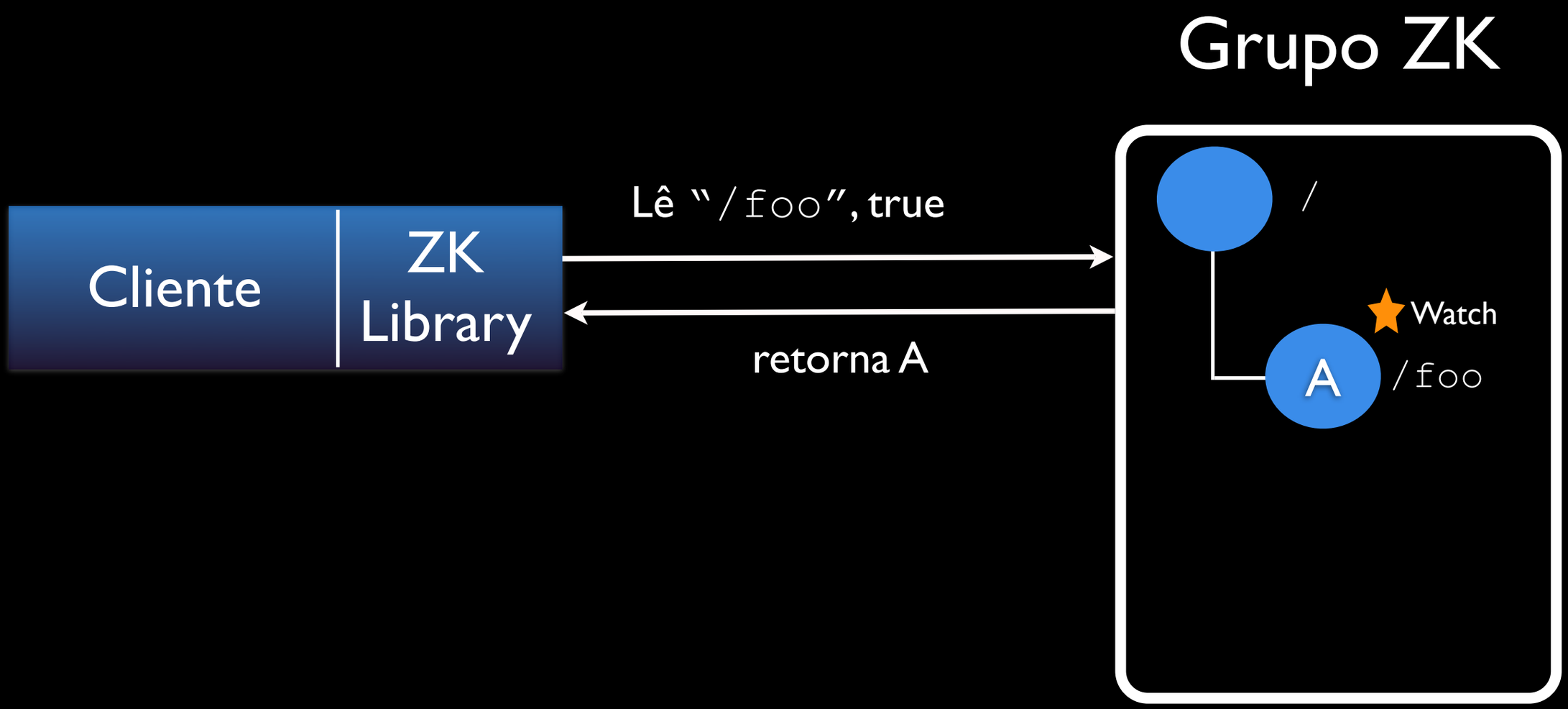
ZooKeeper: Exemplo



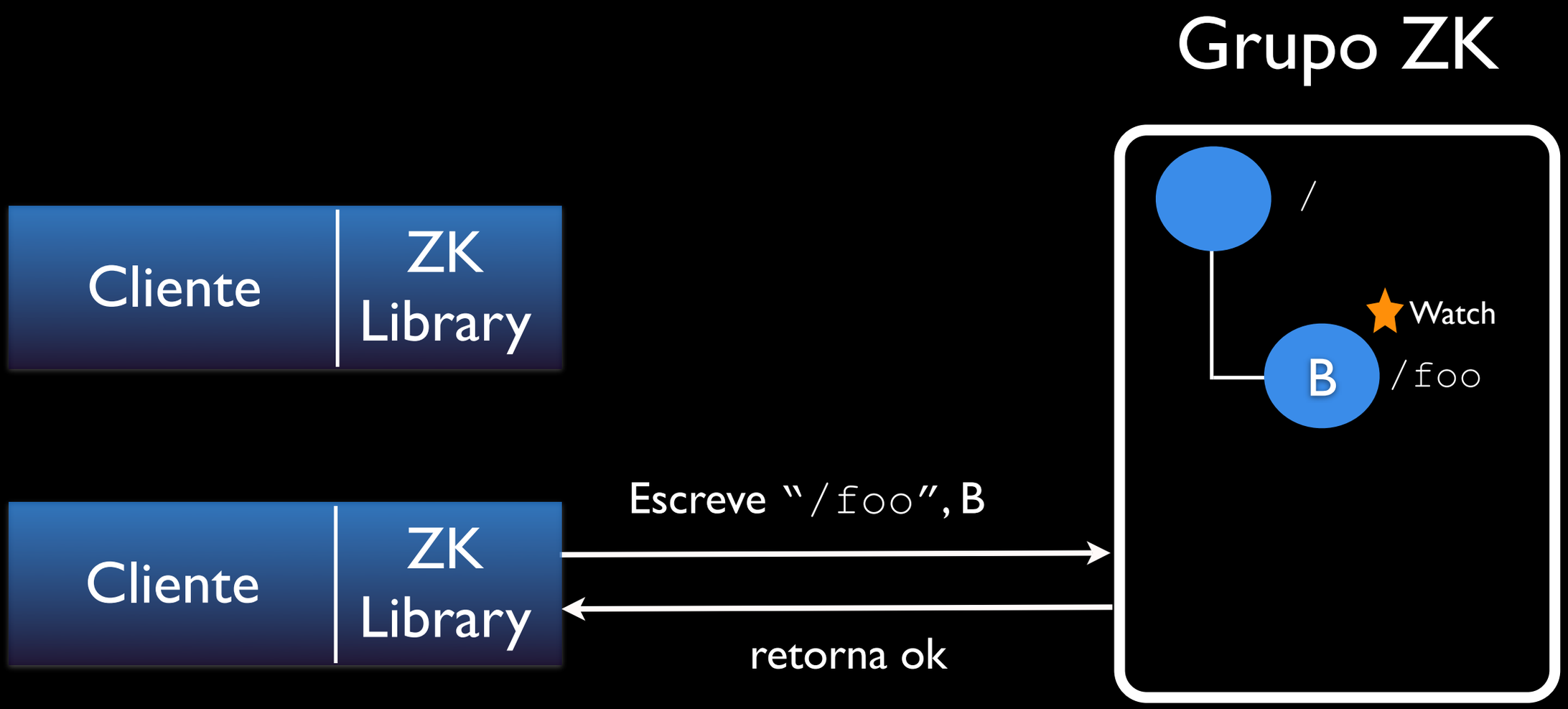
ZooKeeper: Exemplo



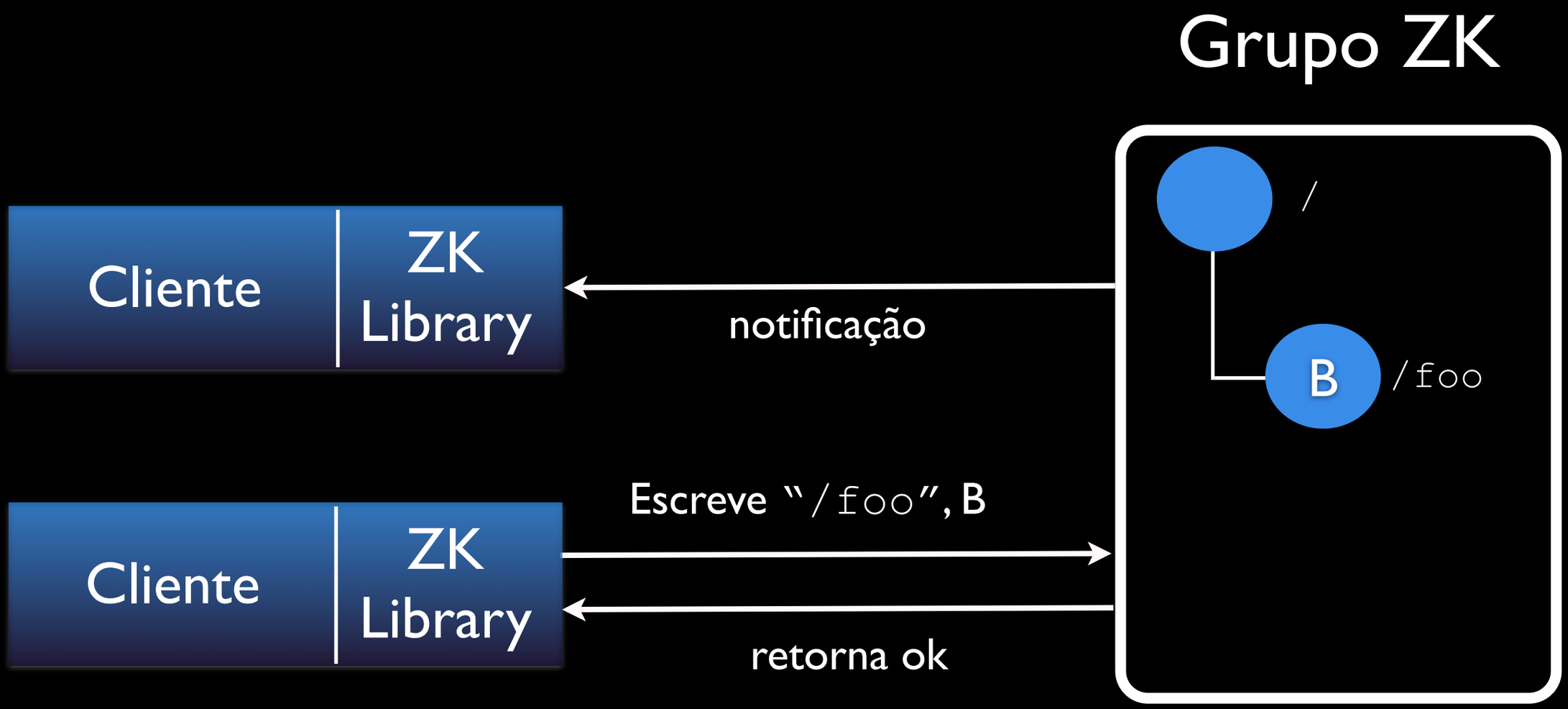
ZooKeeper: *Watches*



ZooKeeper: *Watches*



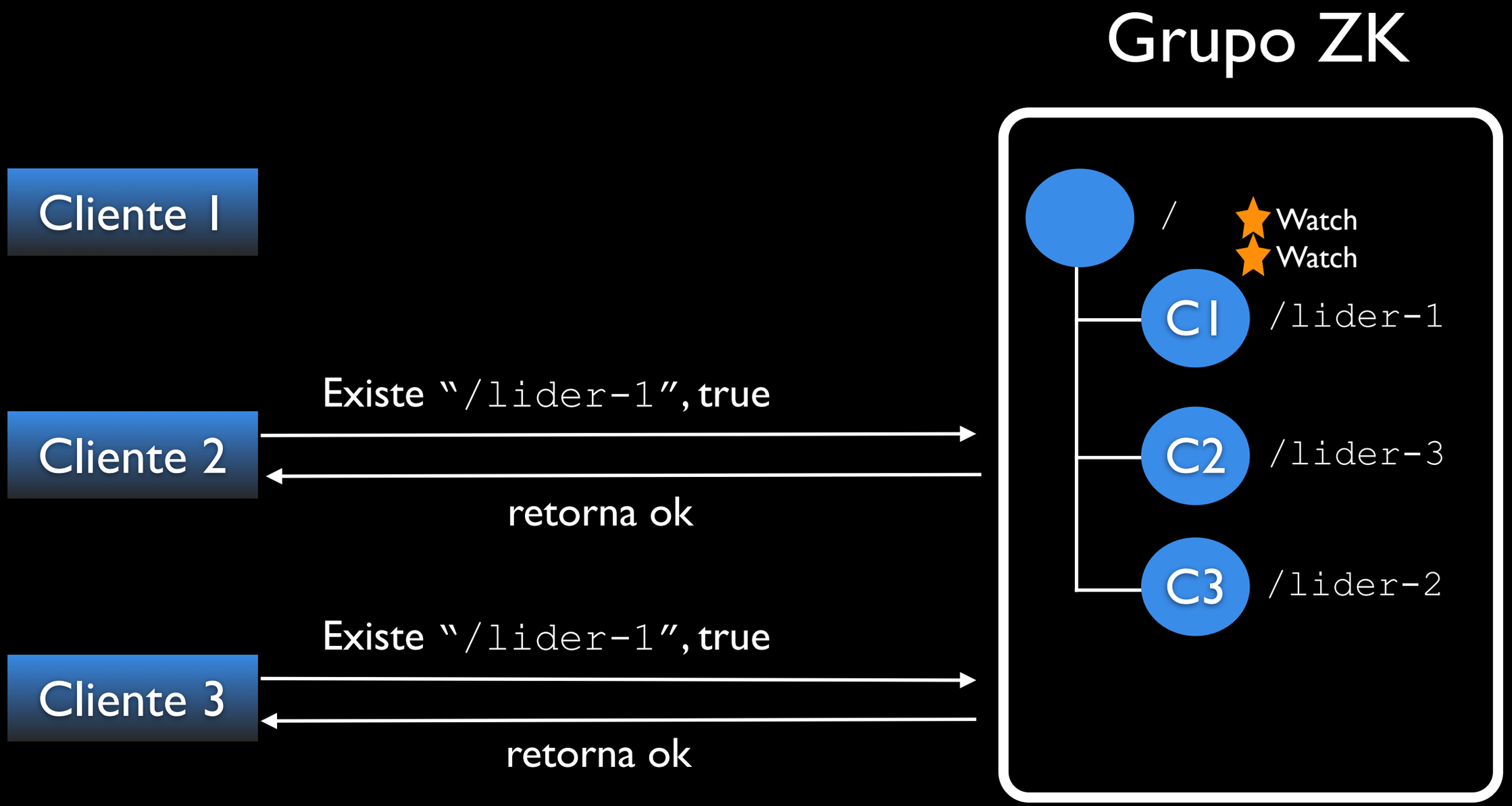
ZooKeeper: *Watches*



Watches, *Locks*, e o Efeito rebanho

- Efeito rebanho
  - ✓ Um elevado número de processos desperta simultaneamente
- Pico de carga
  - ✓ Altamente indesejável

Watches, *Locks*, e o Efeito rebanho

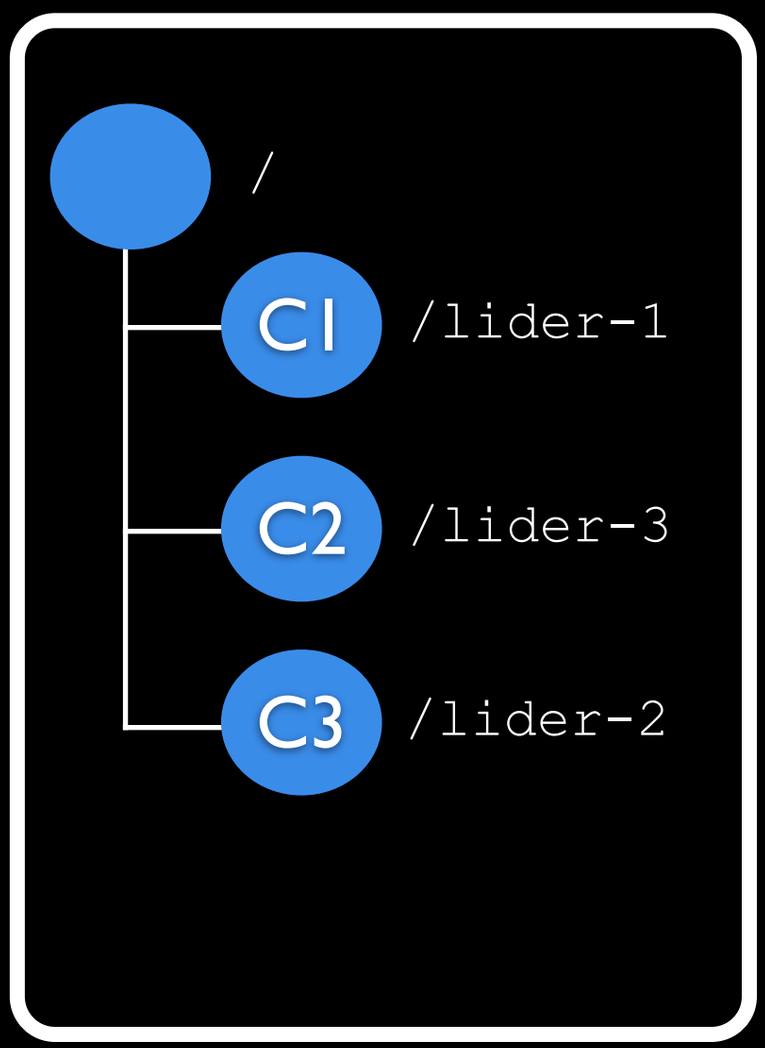


Watches, *Locks*, e o Efeito rebanho

Cliente 1

Cliente 2

Cliente 3

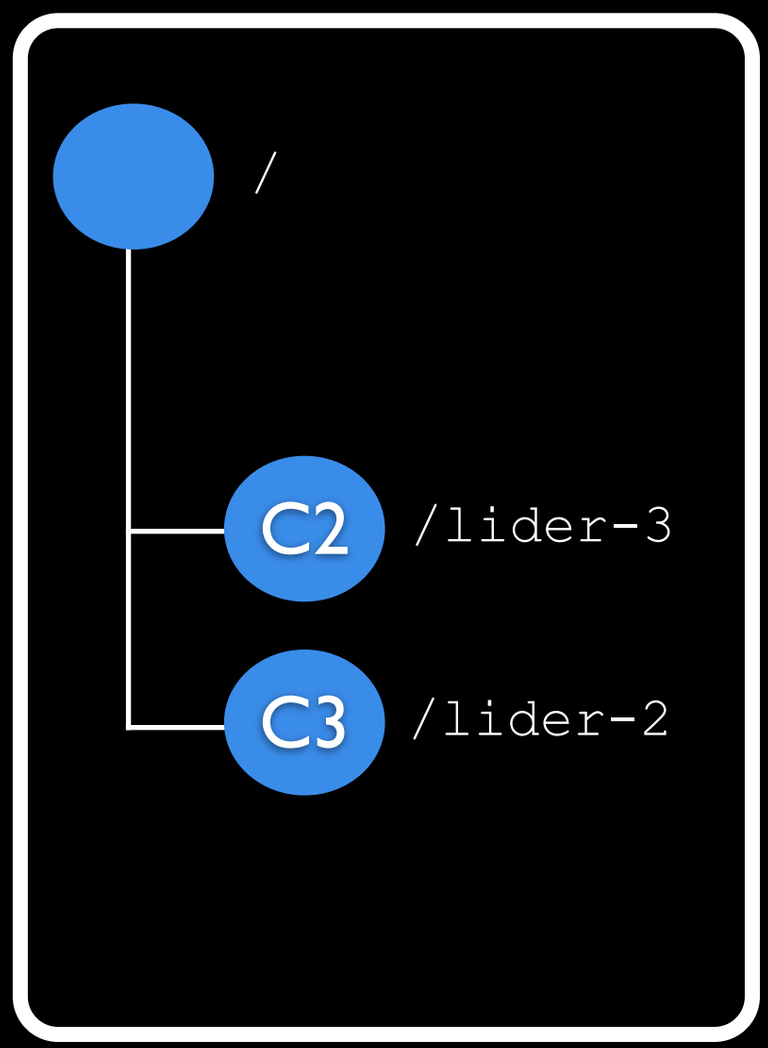


Watches, *Locks*, e o Efeito rebanho

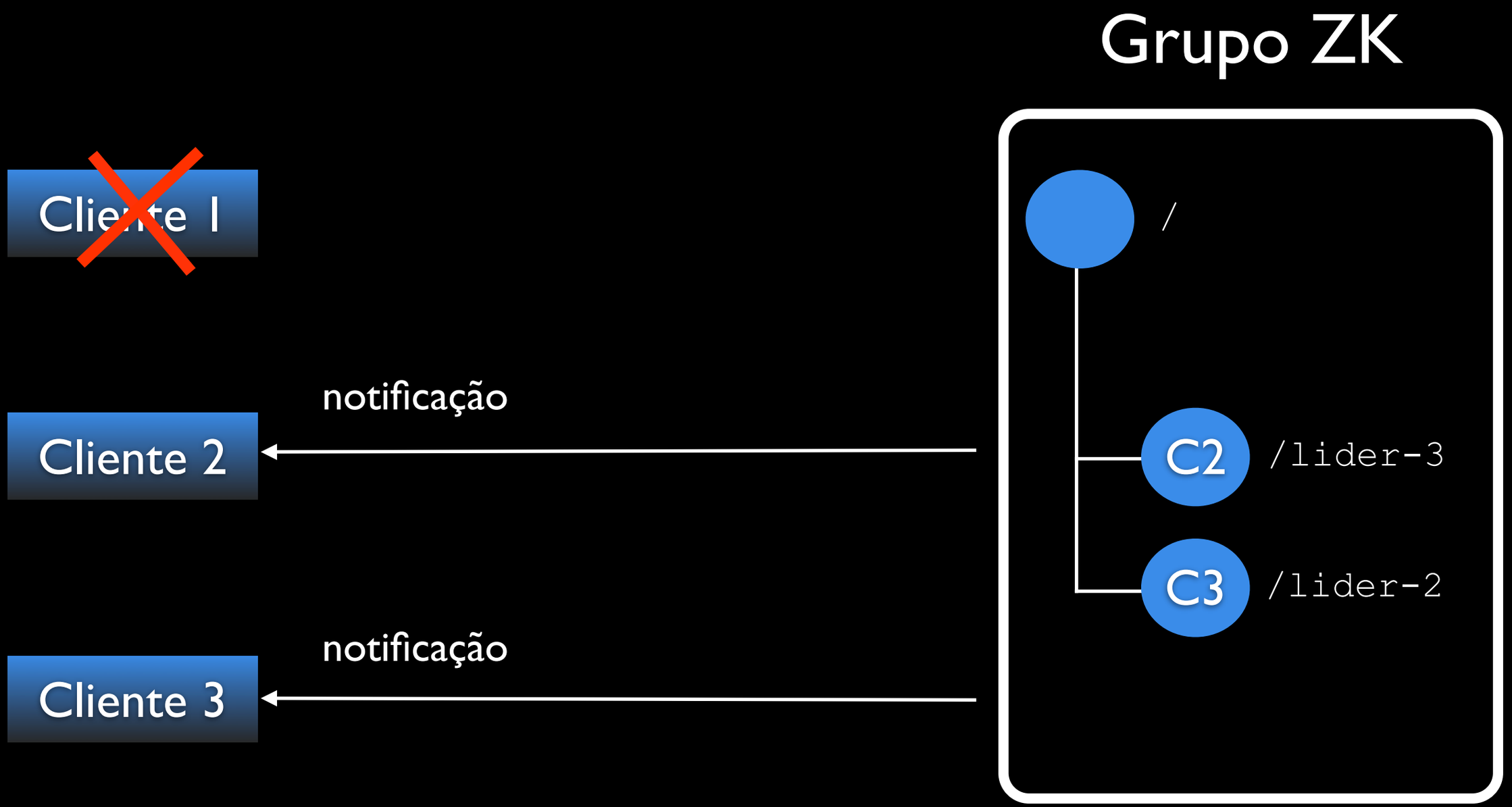
Cliente 1

Cliente 2

Cliente 3



Watches, *Locks*, e o Efeito rebanho



Watches, *Locks*, e o Efeito rebanho

- Solução
  - ✓ Líderes em potencial são ordenados
  - ✓ Cada cliente observa o cliente anterior na sequência estabelecida
  - ✓ Cada falha aciona uma única notificação
- Desvantagem
  - ✓ Somente um único cliente é notificado de uma mudança de líder

Chubby x ZooKeeper

	Chubby	**ZooKeeper**
Cache do cliente	Invalidação direta	*Watches*
Primitivas	Locks	Núcleo (receitas)
Consistência	Todas as operações são linearizáveis	Somente escritas são linearizáveis
Escalabilidade	Não	Maioritariamente leituras
Performance	Baixa	Alta

Linearizabilidade: Quão importante é?

- Depende...
- ZooKeeper implementa um objeto universal

✓ Objeto universal: Herlihy [Herlihy ACM TPLS 1991]

✓ Implementa consenso para *n* processos

Implementando consenso

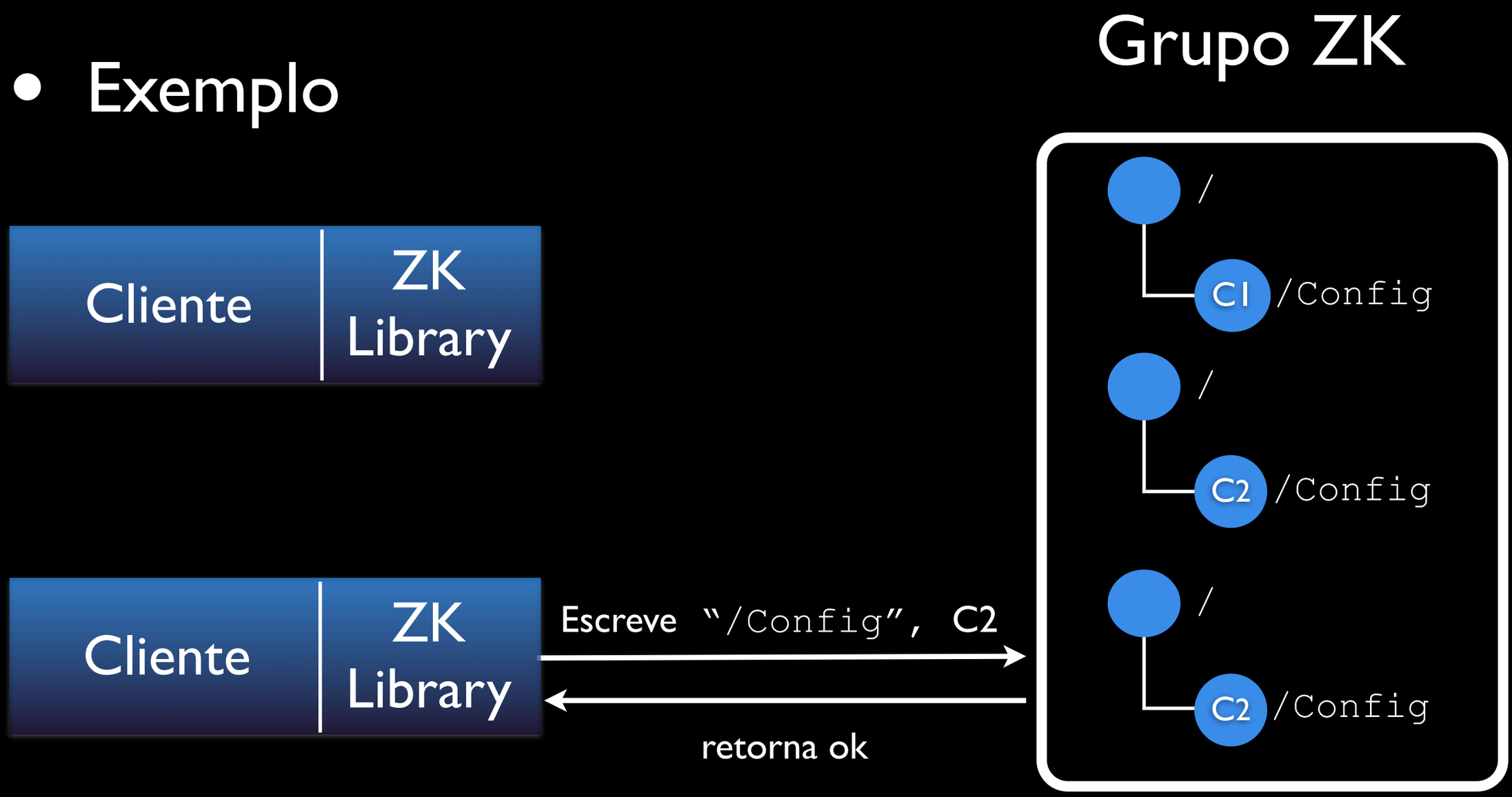
- Cada processo
  - ✓ Propõe(v)
    - ➡ Escreve v com flag sequencial
  - ✓ Decide
    - ➡ Lendo todos os znodes filhos
    - ➡ Selecionando o valor v' com menor número de sequencia
    - ➡ Retornando v'

Linearizabilidade: Quão importante é?

- Depende...
- ZooKeeper implementa um objeto universal segundo Herlihy [Herlihy ACM TPLS 1991]
  - ✓ Implementa consenso para *n* processos
- Visível através de canais escondidos

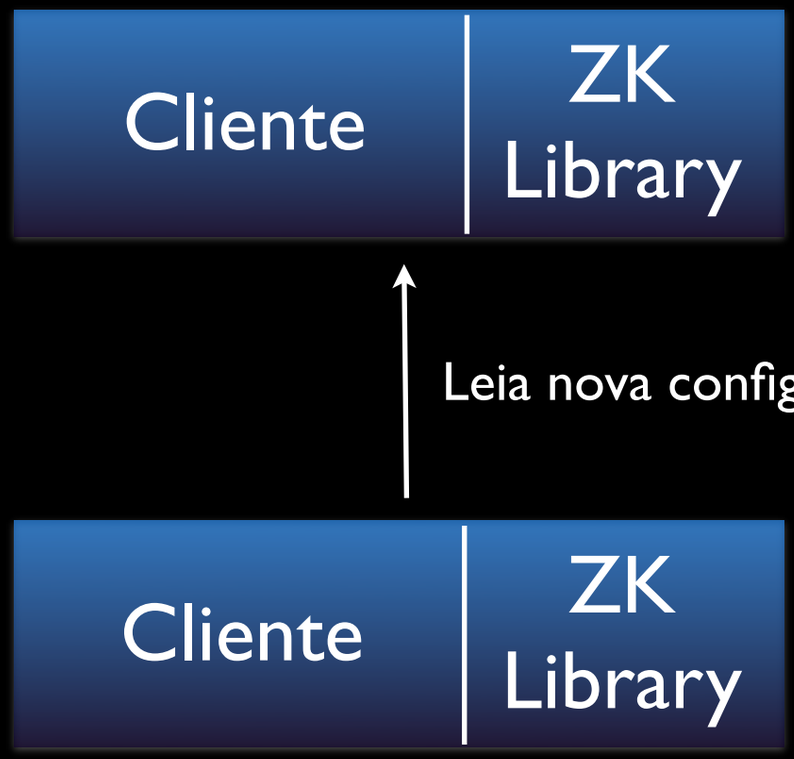
Linearizabilidade: Quão importante é?

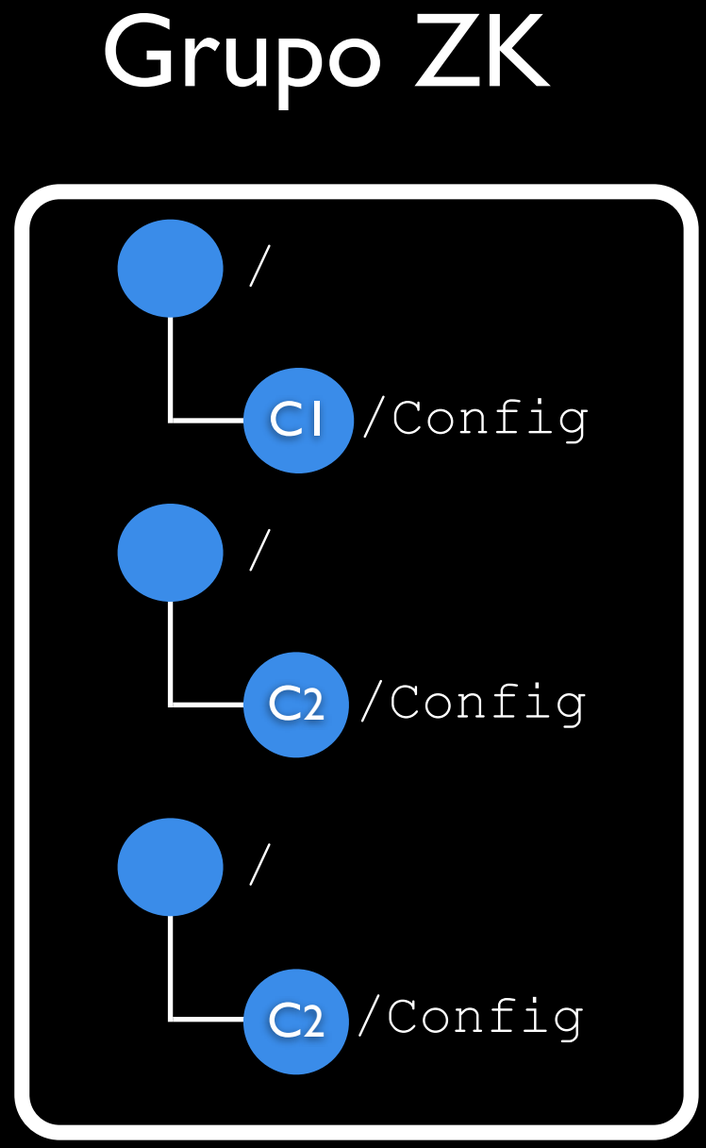
- Exemplo



Linearizabilidade: Quão importante é?

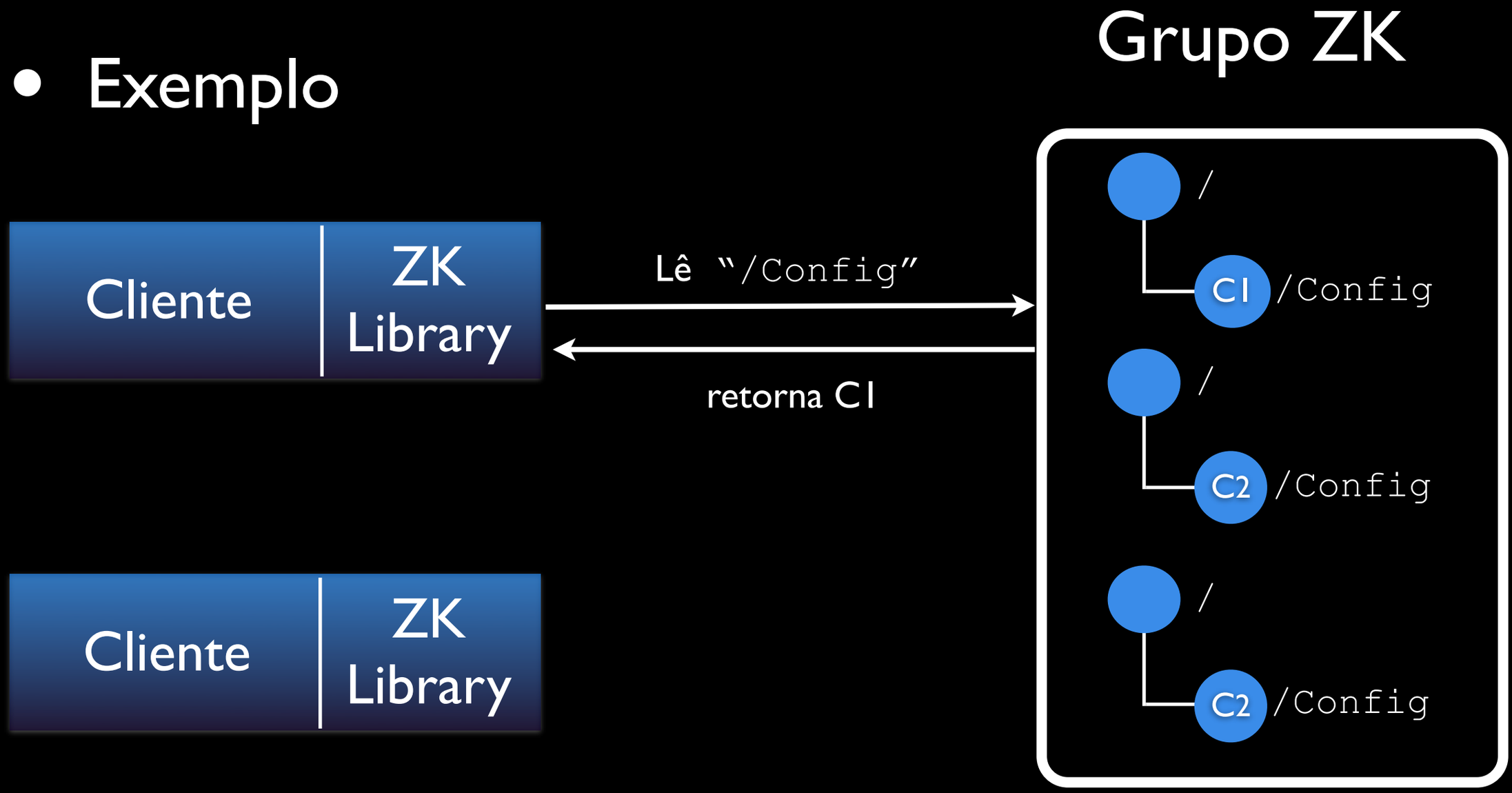
- Exemplo





Linearizabilidade: Quão importante é?

- Exemplo



Da cartola ...

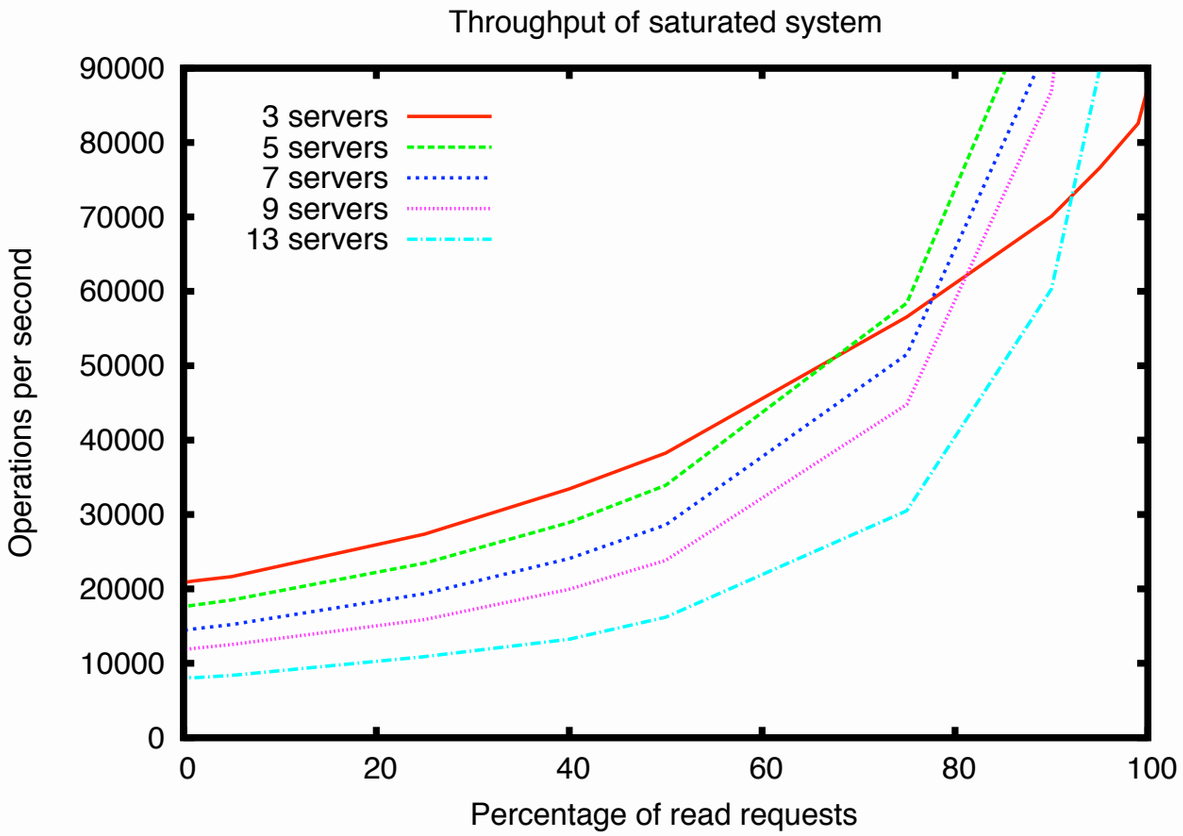
- `sync()`
  - ✓ Operação asíncrona
  - ✓ Descarga o canal entre o seguidor e o líder
  - ✓ Transforma operações em linearizáveis
  - ✓ Menos custoso que executar broadcast atômico

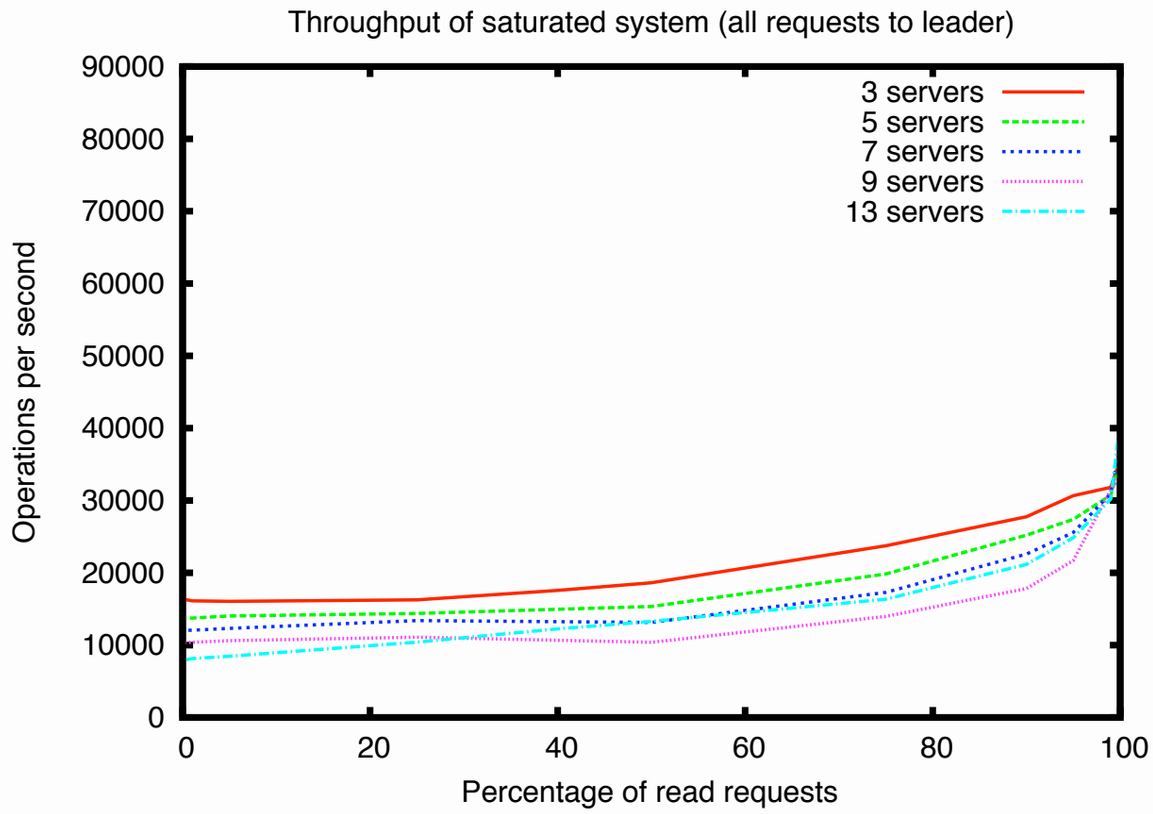
Avaliação & Experiência

Avaliação

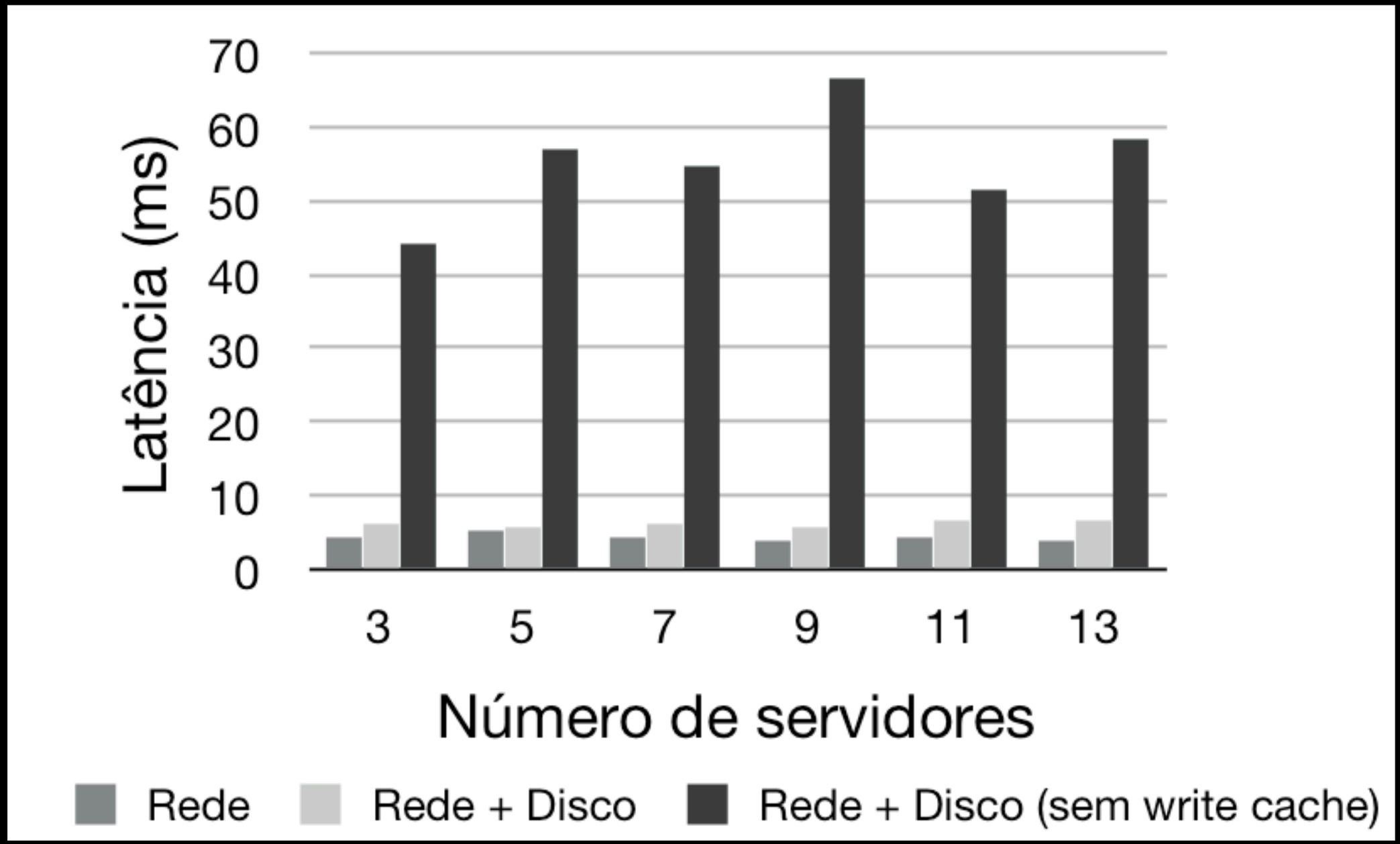
- Cluster de 50 servidores
- Xeon dual-core 2.1 GHz
- 4 GB de RAM
- Dois discos SATA

Vazão (*throughput*)



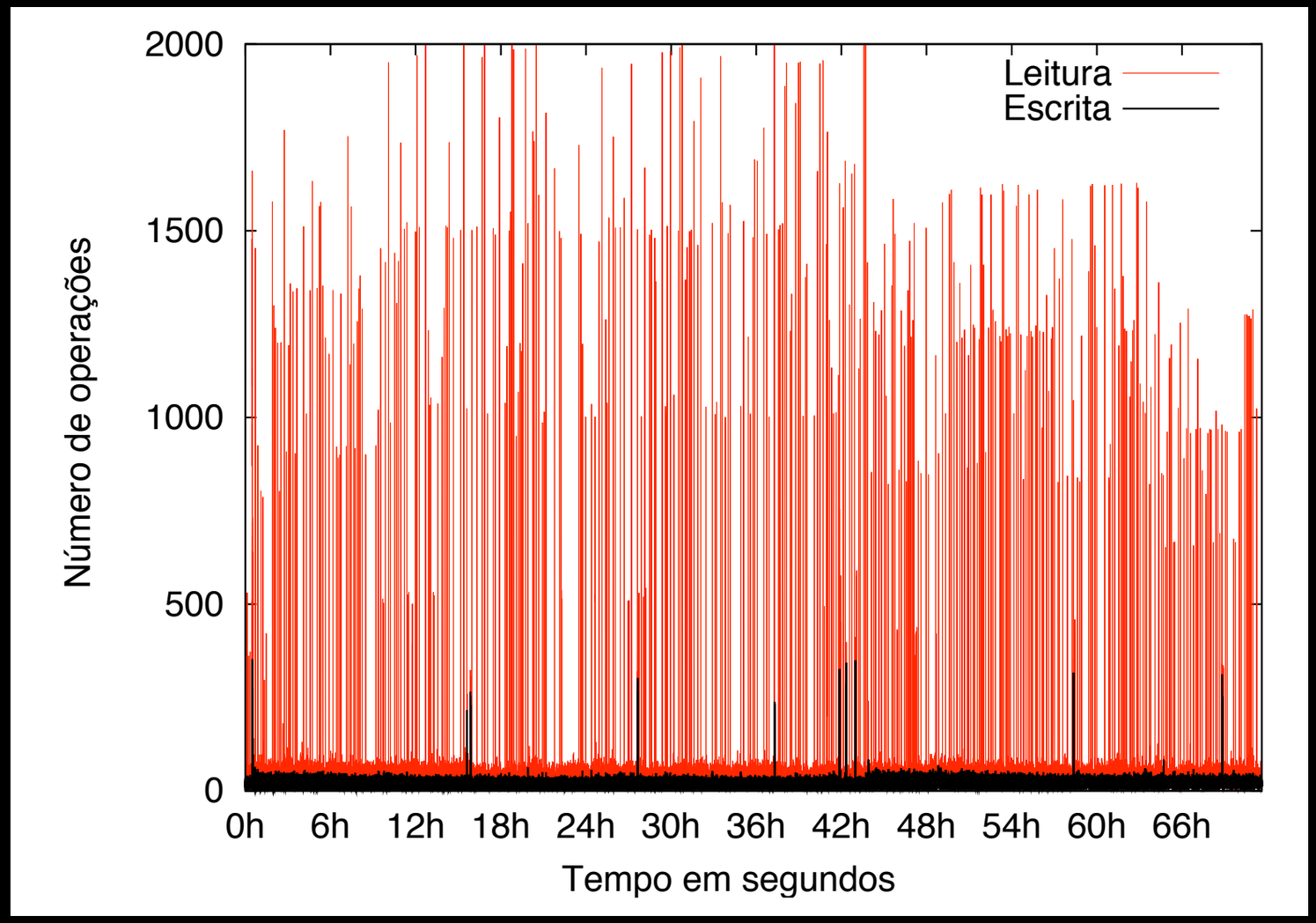


Latência


Latência (ms)
70
60
50
40
30
20
10
0
3
5
7
9
11
13
Número de servidores
Rede
Rede + Disco
Rede + Disco (sem write cache)

Carga em produção

- Serviço de coleta
- Carga de um servidor
- Picos de leitura de mais de 2000 ops/s
- Picos de escrita de menos de 500 ops/s



Em Yahoo!...

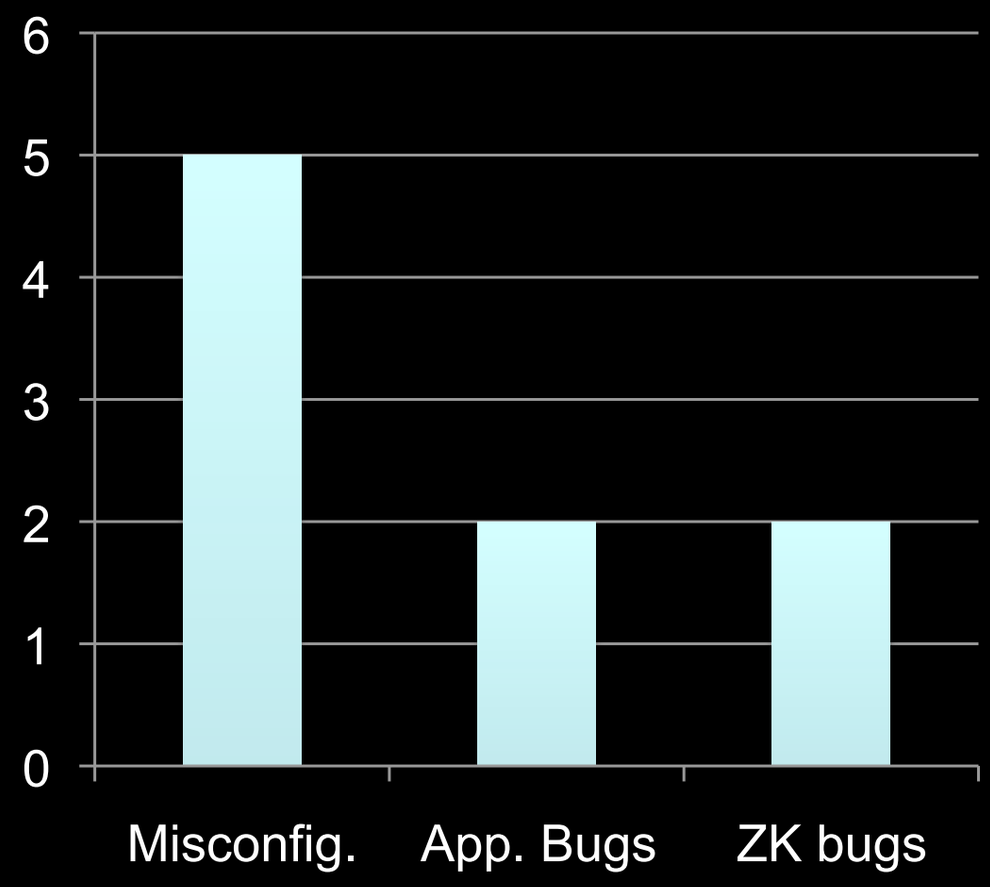
- Se encontra em uso para:
  - ✓ Cross-colo locks
  - ✓ Web crawling
  - ✓ Large-scale publish-subscribe (Hedwig: ZOOKEEPER-775)
  - ✓ Portal front-end
- Maior cluster do qual eu tenho conhecimento
  - ✓ Aproximadamente 5,000 nodes

Faltas na prática

- Bugzilla
  - ✓ Sistema de *tickets* para defeitos de software, melhorias, etc
- Fila do serviço de coleta
  - ✓ Mais de 2 anos em produção
  - ✓ 9 tickets reportando problemas relacionados ao ZooKeeper

Faltas na prática

- Configuração: 5 problemas
  - ✓ System configuration, not ZK
  - ✓ *E.g.*, misconfigured net cards, DNS clash
- *Bugs* da aplicação: 2 problemas
  - ✓ Misunderstanding of the API semantics
  - ✓ E.g., race condition using async api
- Bugs do ZK: 2 problemas
  - ✓ Nossa culpa... =\
  - ✓ API e servidor (todos afetados)



Fechamento

Resumo

- Coordenação
- Chubby
- ZooKeeper
- Experiência

Créditos

- Patrick Hunt, *Cloudera*
- Mahadev Konar, *Yahoo! Grid*
- Benjamin Reed, *Yahoo! Research*
- Henry Robinson, *Cloudera*
- The ZooKeeper community