Introdução ao Processamento Paralelo e Distribuído

Videoaula

Notas dos slides

APRESENTAÇÃO

O presente conjunto de slides pertence à coleção produzida para a disciplina *Introdução ao Processamento Paralelo e Distribuído* ofertada aos cursos de bacharelado em Ciência da Computação e em Engenharia da Computação pelo Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas.

Os slides disponibilizados complementam as videoaulas produzidas e tratam de pontos específicos da disciplina. Embora tenham sido produzidos para ser assistidos de forma independente, a sequência informada reflete o encadeamento dos assuntos no desenvolvimento do conteúdo programático previsto para a disciplina.

Sistemas Distribuídos

Replicação

Notas da videoaula

DESCRIÇÃO

Nesta videoaula são apresentados questões relacionadas à replicação em Sistemas Distribuídos

OBJETIVOS

Nesta videoaula são apresentados aspectos ligados à replicação em Sistemas Distribuídos.

“ You got them all except the original.

Multiman, The Impossibles

Avant Propos

A **replicação** é um recurso que pode ser explorado de diferentes formas em um sistema distribuído:

- Melhorar o desempenho
- Aumentar a disponibilidade
- Promover Tolerância a Falhas

Avant Propos

Correção e integridade das réplicas

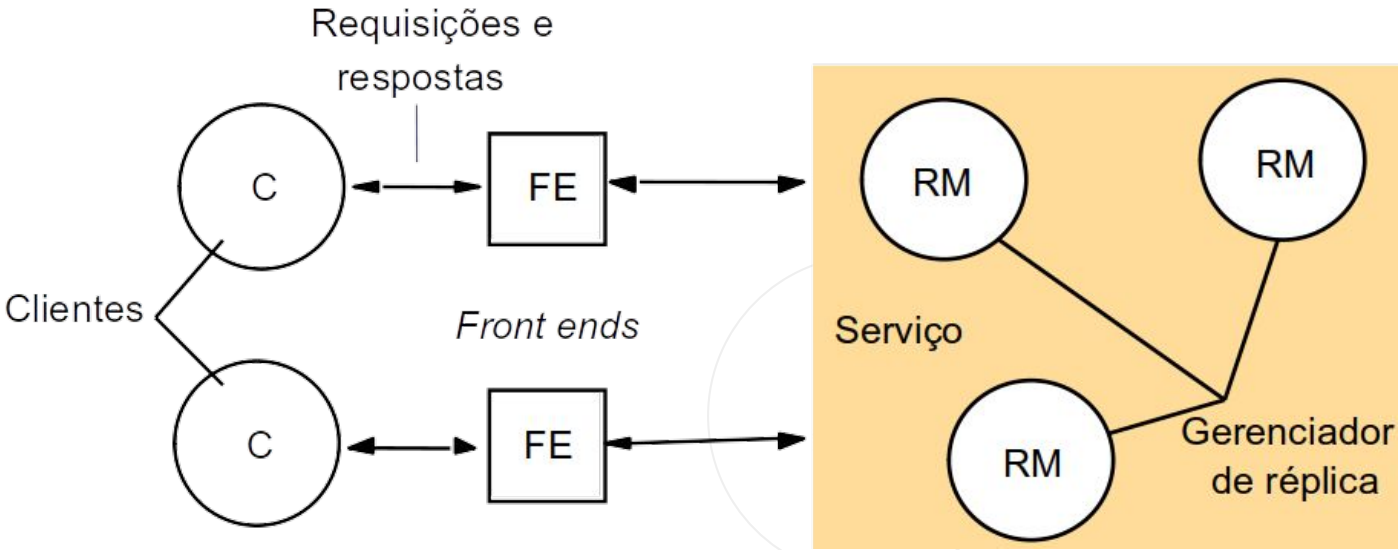
Linearização: atualização das informações a todos clientes, ou mesmo a todas réplicas, em tempo real é um requisito desejável, porém não atingível. A capacidade de linearização é a interposição das operações para todos os clientes, para qualquer operação. A replicação é dita linearizada quando:

- O serviço é realizado como se não existissem réplicas
- É consistente com os tempos reais verificados na execução das operações nos clientes

Consistência sequencial: consiste na interposição das operações, mas não garante o tempo real. A noção de ordem aplicada é a da ocorrência dos eventos em cada cliente, assim, a interposição de operações pode inverter ordem de operações entre os clientes, desde que a ordem de cada cliente não seja violada.

Modelo Base

O serviço encontra-se implementado em diferentes réplicas. Os clientes acessam os serviços por meio de front ends. O gerenciador de réplica grante atendimento ao serviço considerando a existência de réplicas mas não deixando transparecer esta situação ao cliente.



Modelo Base: Etapas

1. Requisição: o FE envia uma requisição

- a um RM, o qual comunica-se com os demais
- a todos os RM (multicast)

2. Coordenação: determina a ordem em que as requisições são tratadas pelo RM

- FIFO: as requisições são tratadas, pelos RM, na mesma ordem em que foram enviadas pelo FE
- Causal: se uma requisição X ocasionou uma requisição Y, a requisição X deve ser tratada, pelos RMs, antes do tratamento de Y
- Total: se um RM tratou a requisição X antes da requisição Y, todos os RM devem tratar X antes de Y.

3. Execução: todos os RM executam a operação relacionada a requisição (de modo que possa ser desfeita).

4. Acordo: os RM devem entrar de acordo de as execuções podem ser aceitas.

5. Resposta: retorno, ao FE, da resposta a sua requisição.

- pode ser enviada por um único RM.
- pode ser enviado por vários RMs.

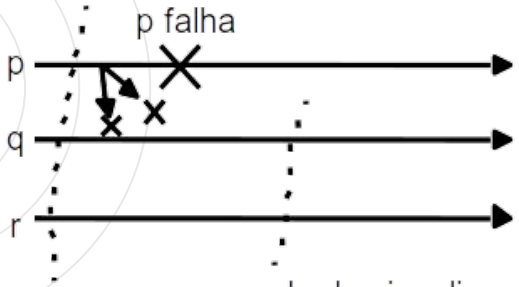
Modelo Base: Comunicação em Grupo

Modo de visualização

- **Ordem:**
  - Se um processo ***p*** envia o modo de visualização ***v*** e depois o modo de visualização ***v'***, então nenhum outro processo ***q*** envia ***v'*** antes de ***v***.
- **Integridade:**
  - Se um processo ***p*** envia o modo de visualização ***v*** então $p \in v$
- **Não trivialidade:**
  - Se o processo ***q*** entra em um grupo a partir de um processo ***p***, então ***q*** estara sempre nos modos de visualização enviados por ***p***. De forma análoga, havendo uma repartição dos grupos, os processos em cada grupo excluirão os processos dos outros grupos das visualizações.

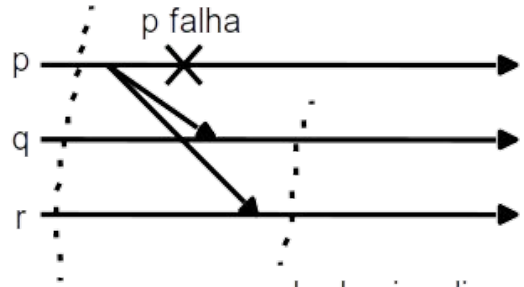
Modelo Base: Comunicação em Grupo

a (permitido).



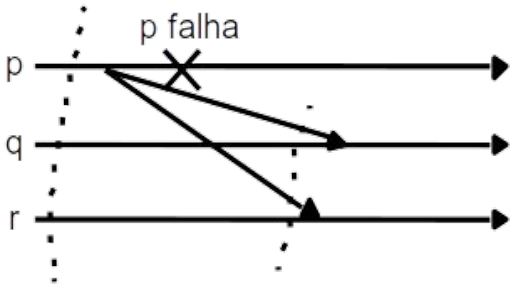
modo de visualização (p, q, r)

b (permitido).



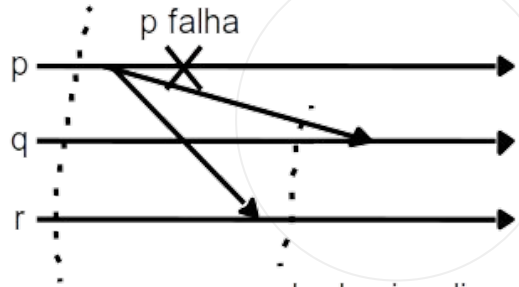
modo de visualização (p, q, r)

c (proibido).



modo de visualização (p, q, r)

d (proibido).



modo de visualização (p, q, r)

Modelo Passivo: Backup Primário

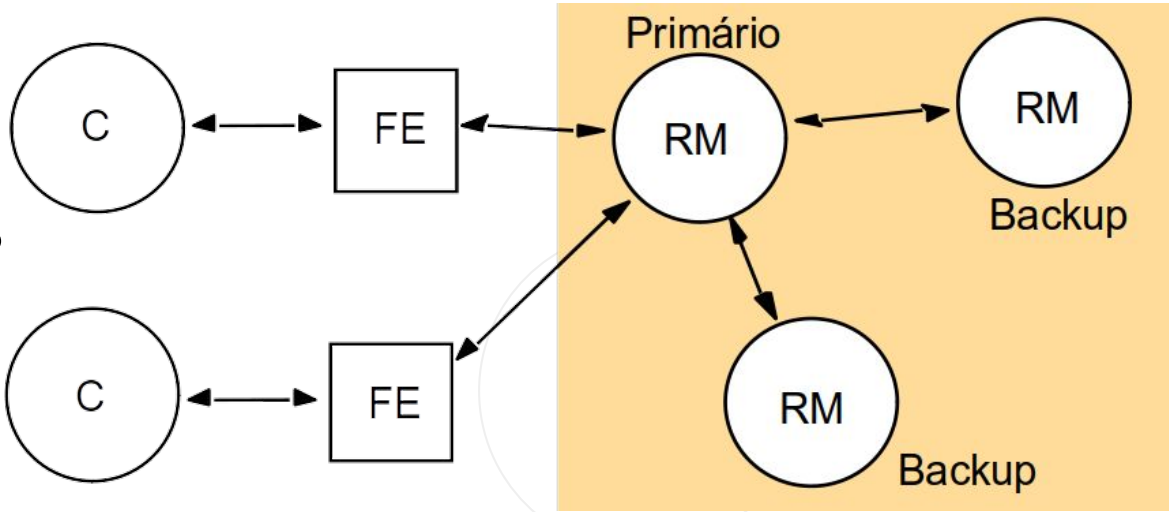
1. Requisição: as requisições possuem id único e são enviadas para um RM primário

2. Coordenação: trata as requisições de forma atômica

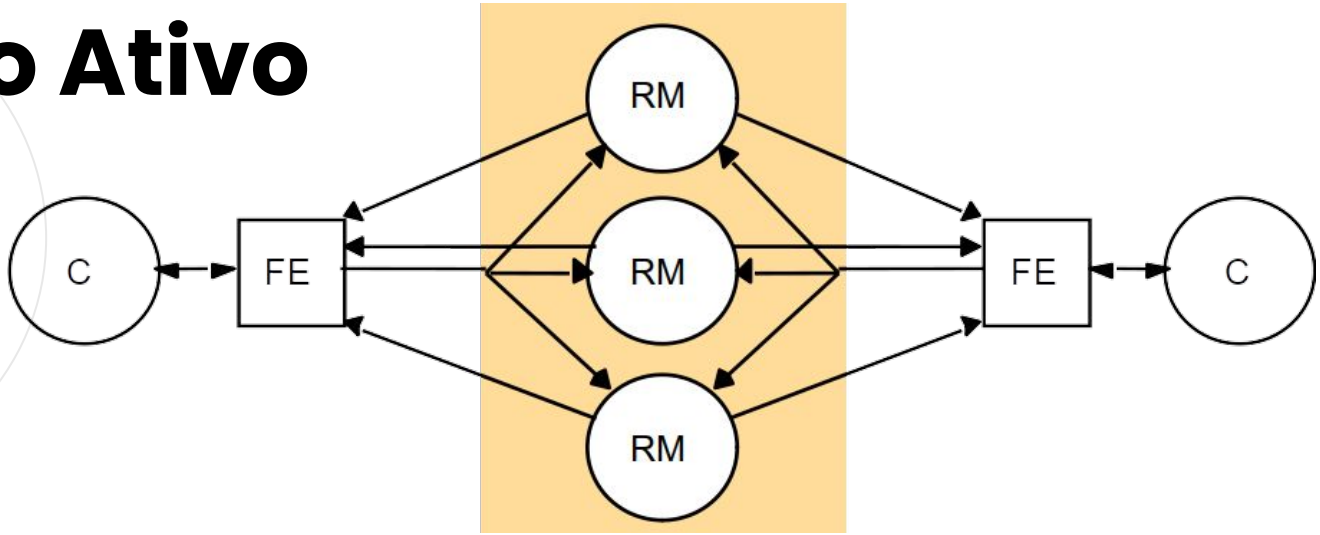
3. Execução: o RM primário executa a requisição e armazena a resposta

4. Acordo: se a requisição alterou dados, o RM primário envia uma requisição de atualização nos demais RM e aguarda uma resposta destes

5. Resposta: ao final é retornada a resposta, pelo RM primário, ao FE



Modelo Ativo



1. Requisição: as requisições possuem id único e são enviadas para todos RM e tratadas na mesma ordem

2. Coordenação: a ordem é garantida pela estratégia de comunicação em grupo

3. Execução: todas as réplicas executam a requisição, sendo respeitada a ordem, todas as requisições são respeitadas da mesma forma

4. Acordo: não há necessidade de acordo, em função da semântica de entrega do mecanismo de comunicação em grupo

5. Resposta: todos respondem ao FE - daí depende da política de TF empregada: se for à crash, a prime resposta é suficiente; se for a imprecisão, considera todas as respostas

Arquitetura GOSSIP

Estudo de Caso

Fofoca

Por meio da replicação, o objetivo da arquitetura Gossip é oferecer alta disponibilidade replicando dados próximo à localização de grupos de clientes, onde há demanda, portanto.

Os RM trocam mensagens (fofocas) periodicamente relatando atualizações do que foi recebido dos clientes.

Acesso dos clientes com duas operações:

- ***Query***: operação de leitura, sem alteração do estado
- ***Update***: operação de escrita, altera o estado

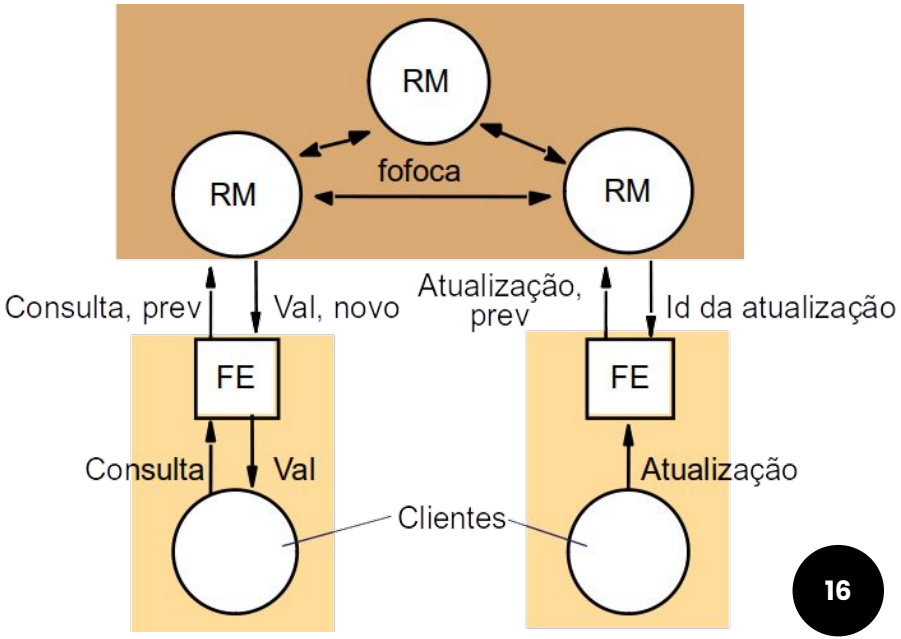
Vantagem: resiliente ao particionamento do sistema (por falha ou decisão)
Desvantagem: escalabilidade, pois “fofocar” é custoso

Usos típicos: lista de discussão eletrônica, um serviço de agenda de alta disponibilidade, log de operações dos clientes (como vendas ou desempenho esportivo).

Arquitetura: Fofoca

Query: Cada cliente obtém o dado que reflete a última atualização observada pelo cliente, independentemente do FE acessado.

Update: Todos os RM recebem, eventualmente, todas as atualizações, pelas fofocas, e as aplicam de forma que todas as réplicas são suficientemente semelhantes para atender às necessidades da aplicação. Embora esta arquitetura possa ser usada para alcançar consistência sequencial, a garantia oferecida é de consistência mais fracas. Dois clientes podem observar réplicas diferentes, mesmo que as réplicas incluam o mesmo conjunto de atualizações, e um cliente pode observar dados desatualizados.

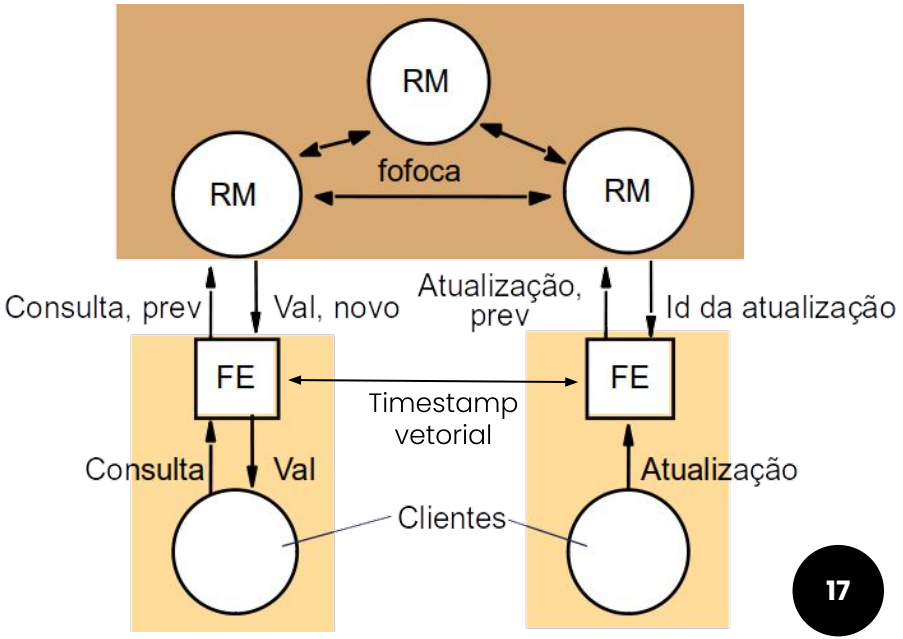


Arquitetura: Fofoca

Timestamp vetorial

Sempre que um FE for acionado por algum cliente, ele propaga, aos demais FEs, seu timestamp, representado por um vetor. Cada entrada deste timestamp vetorial representa uma réplica e identifica a versão do dado em cada réplica.

O timestamp vetorial também é enviado às réplicas (prev) e recebido após a execução da operação, query/update, realizada (novo).



Arquitetura: Fofoca

1. Requisição: update: retorna imediatamente e o FE se encarrega de processar a requisição; query: o FE consulta um RM.

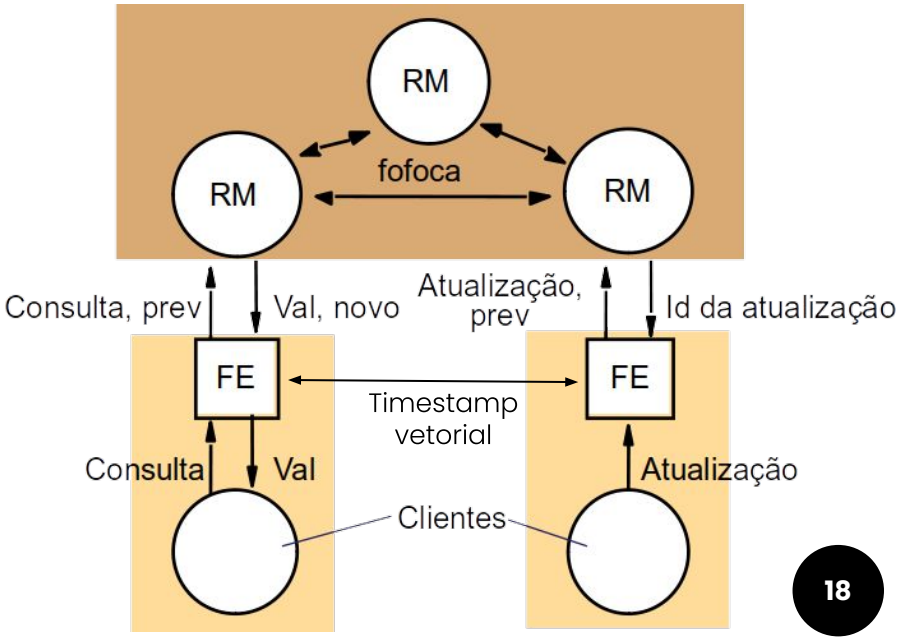
- No caso de um update, o RM responde ao FE indicando que recebeu a requisição

2. Coordenação: a RM que não aplica uma requisição enquanto as restrições de ordem não forem atendidas, o que envolve receber updates a partir de fofocas.

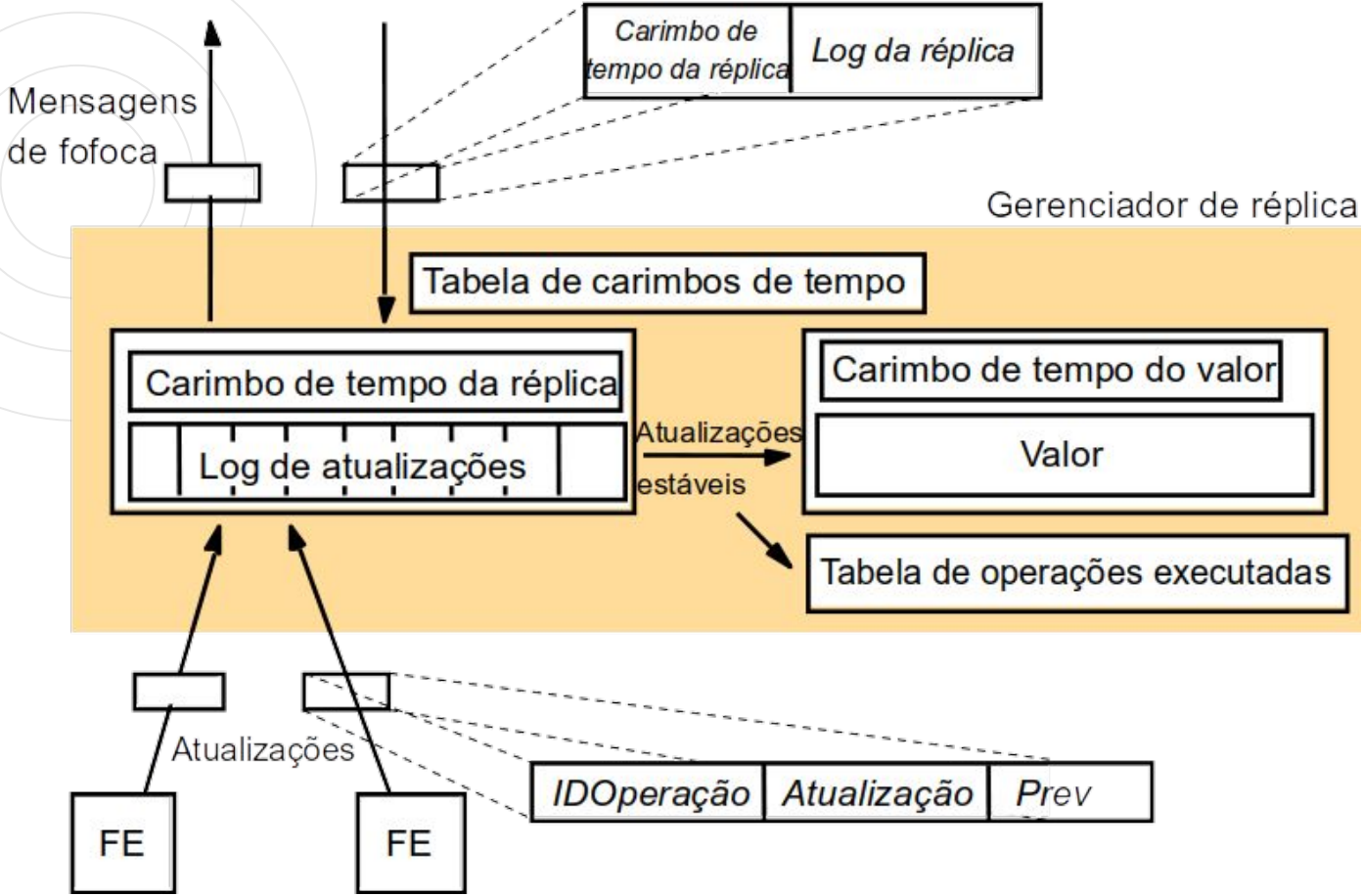
3. Execução: cada RM executa a requisição

4. Resposta: se a requisição for uma query, o retorno é neste ponto.

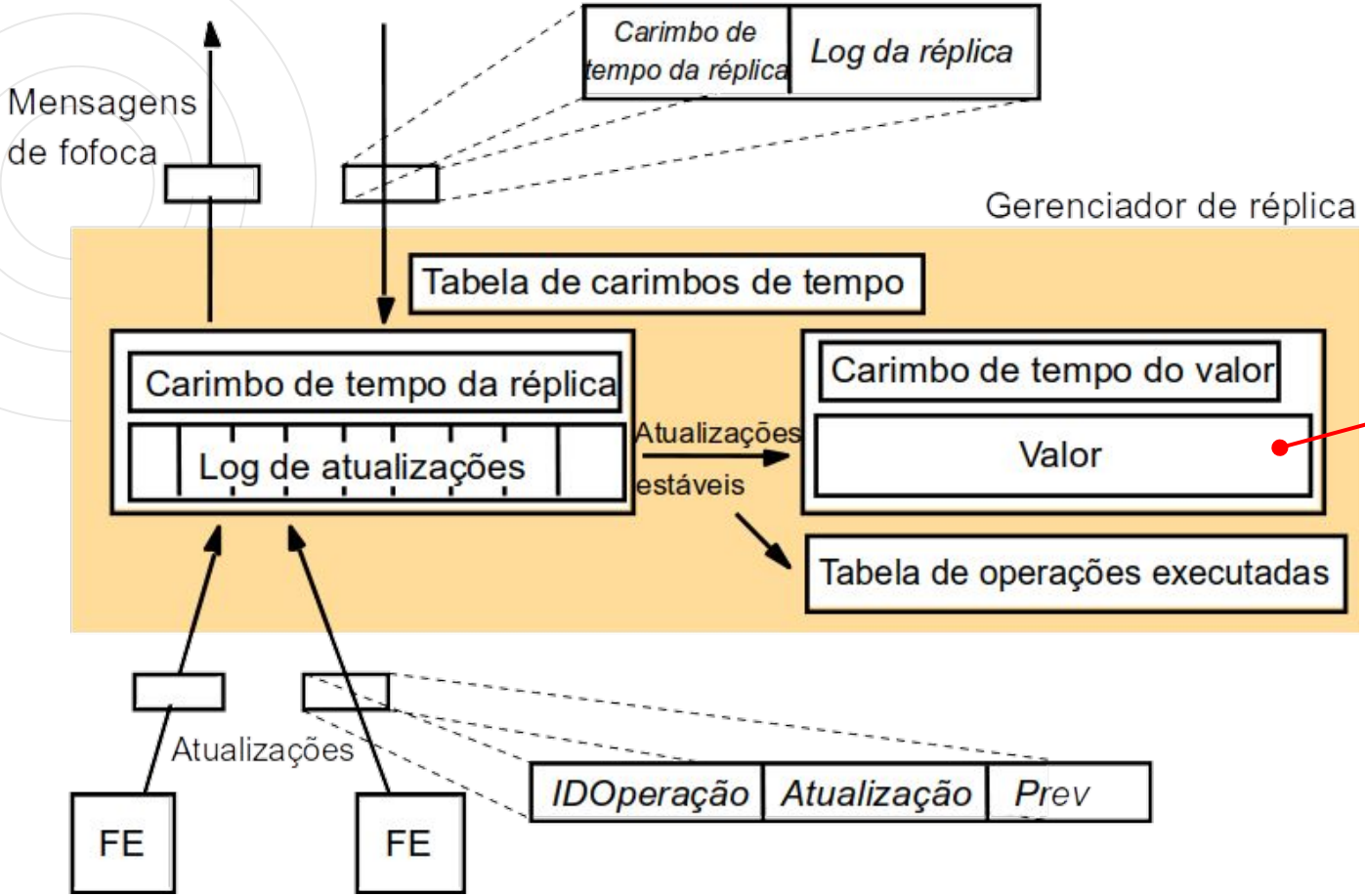
5. Acordo: os RM atualizam-se mutuamente pelas mensagens de fofocas. A atualização é preguiçosa, uma vez que as fofocas são ocasionais.



Arquitetura: Fofoca

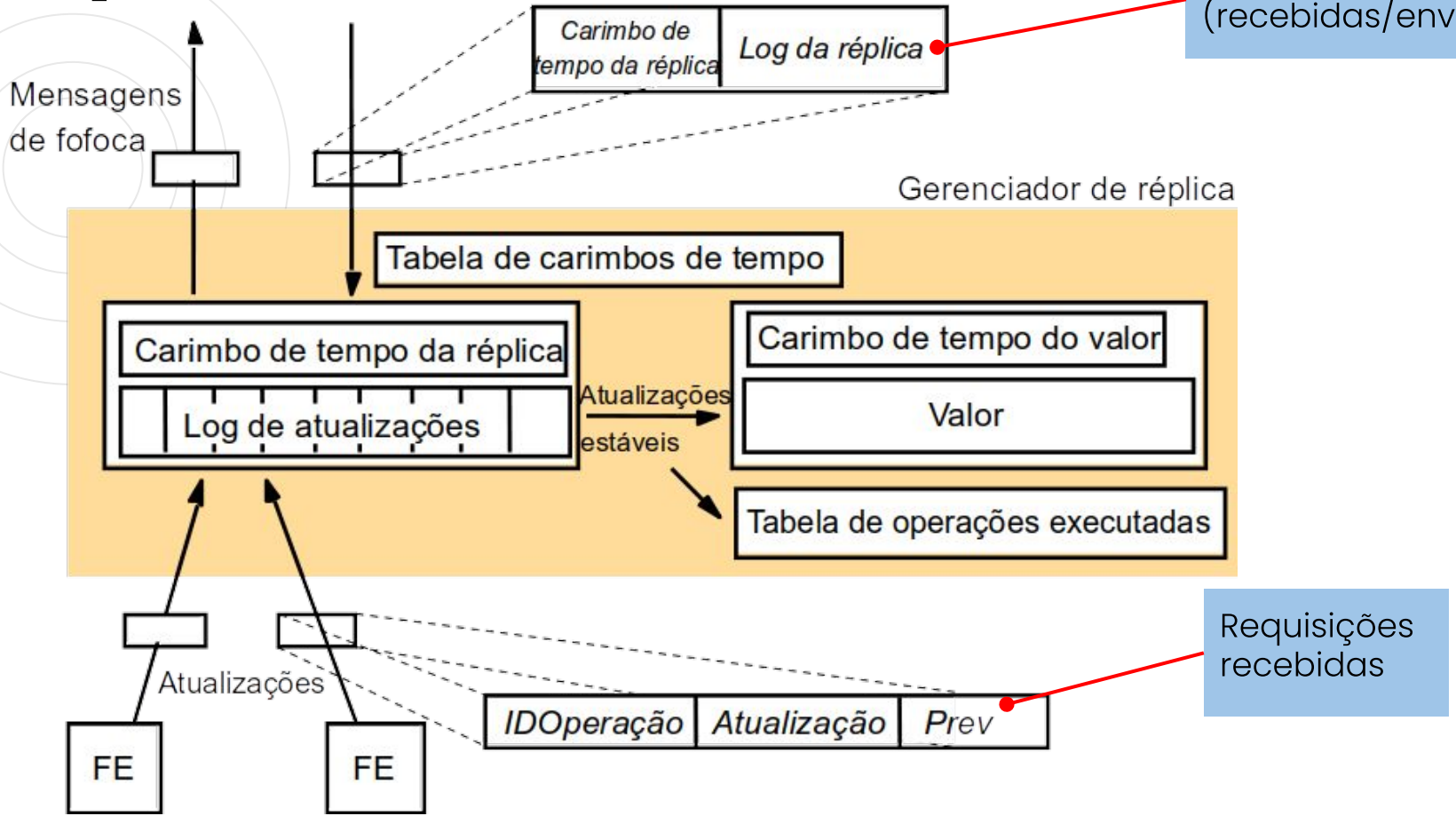


Arquitetura: Fofoca



É o valor armazenado na réplica (estável)

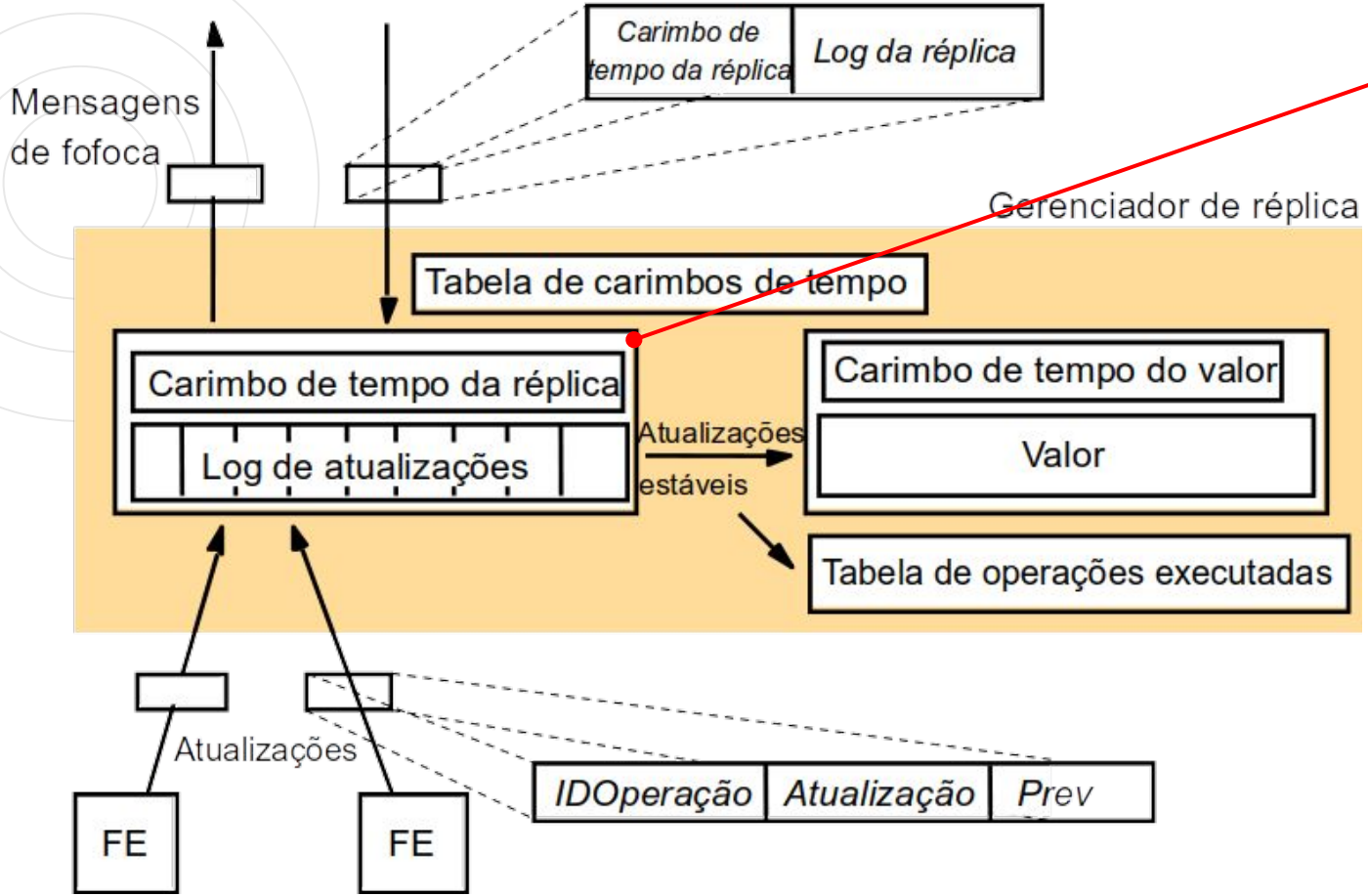
Arquitetura: Fofoca



Mensagens de fofoca (recebidas/enviadas)

Requisições recebidas

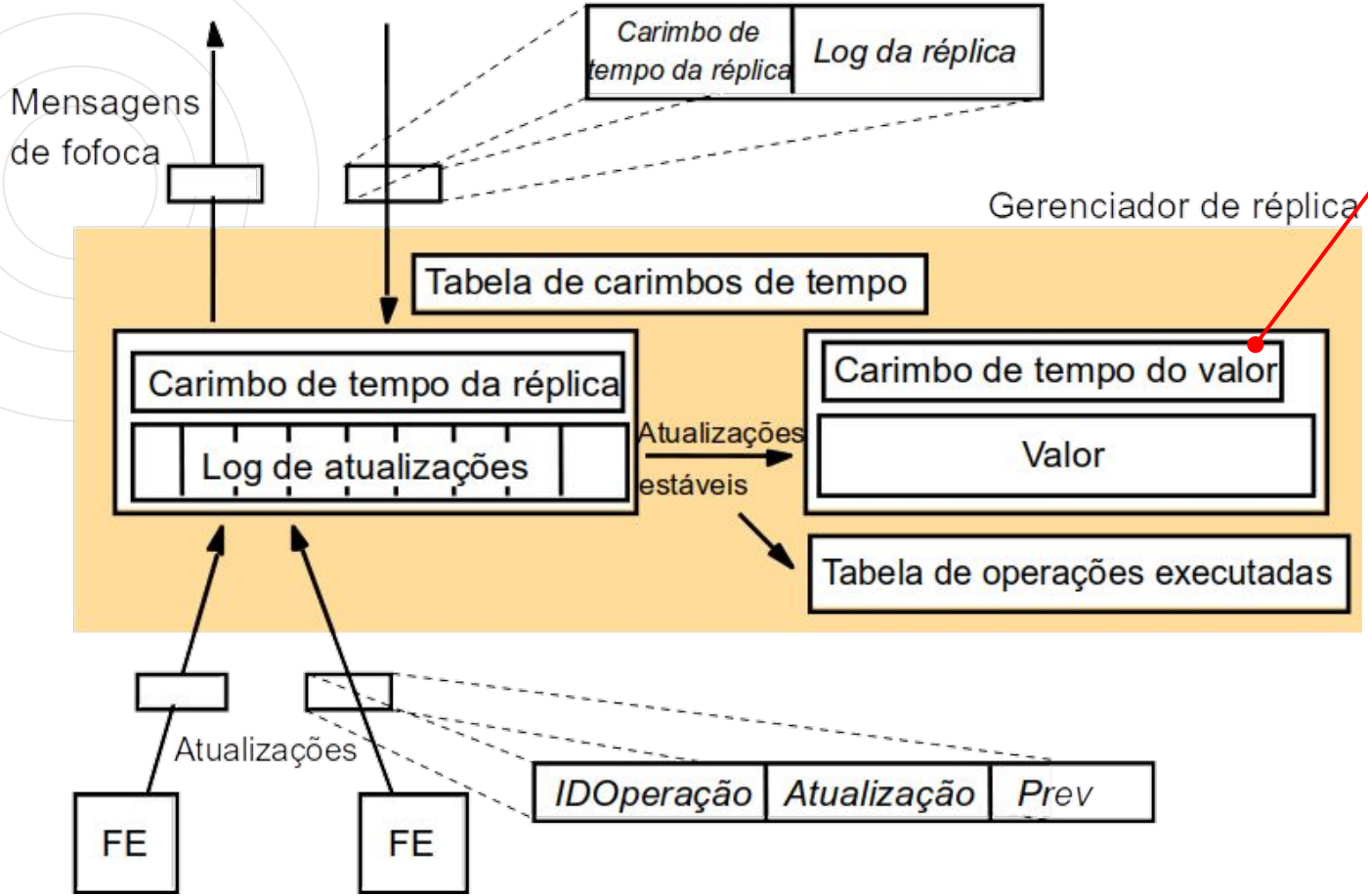
Arquitetura: Fofoca



Permite que apenas atualizações consistentes com suas garantias de ordenação sejam aplicadas.

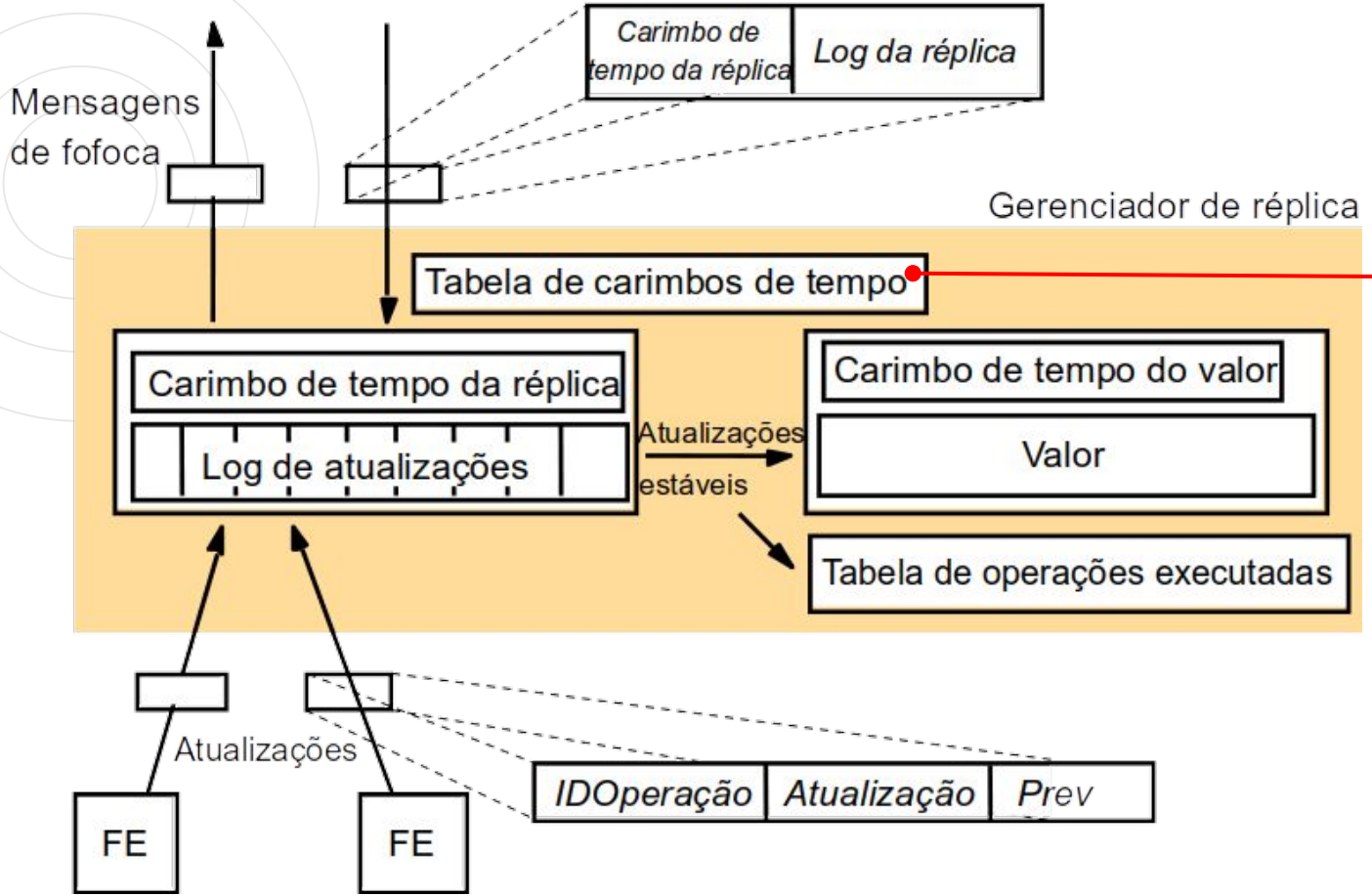
Log: Mantém informação das operações já aplicadas

Arquitetura: Fofoca



Carimbo de Tempo do Valor: timestamp vetorial que registra a atualização anotada no valor.

Arquitetura: Fofoca



Carimbo de Tempo da Réplica:
Carimbo de tempo vetorial de todos os outros gerenciadores de Réplica (preenchido no recebimento de fofocas).

Arquitetura: Fofoca

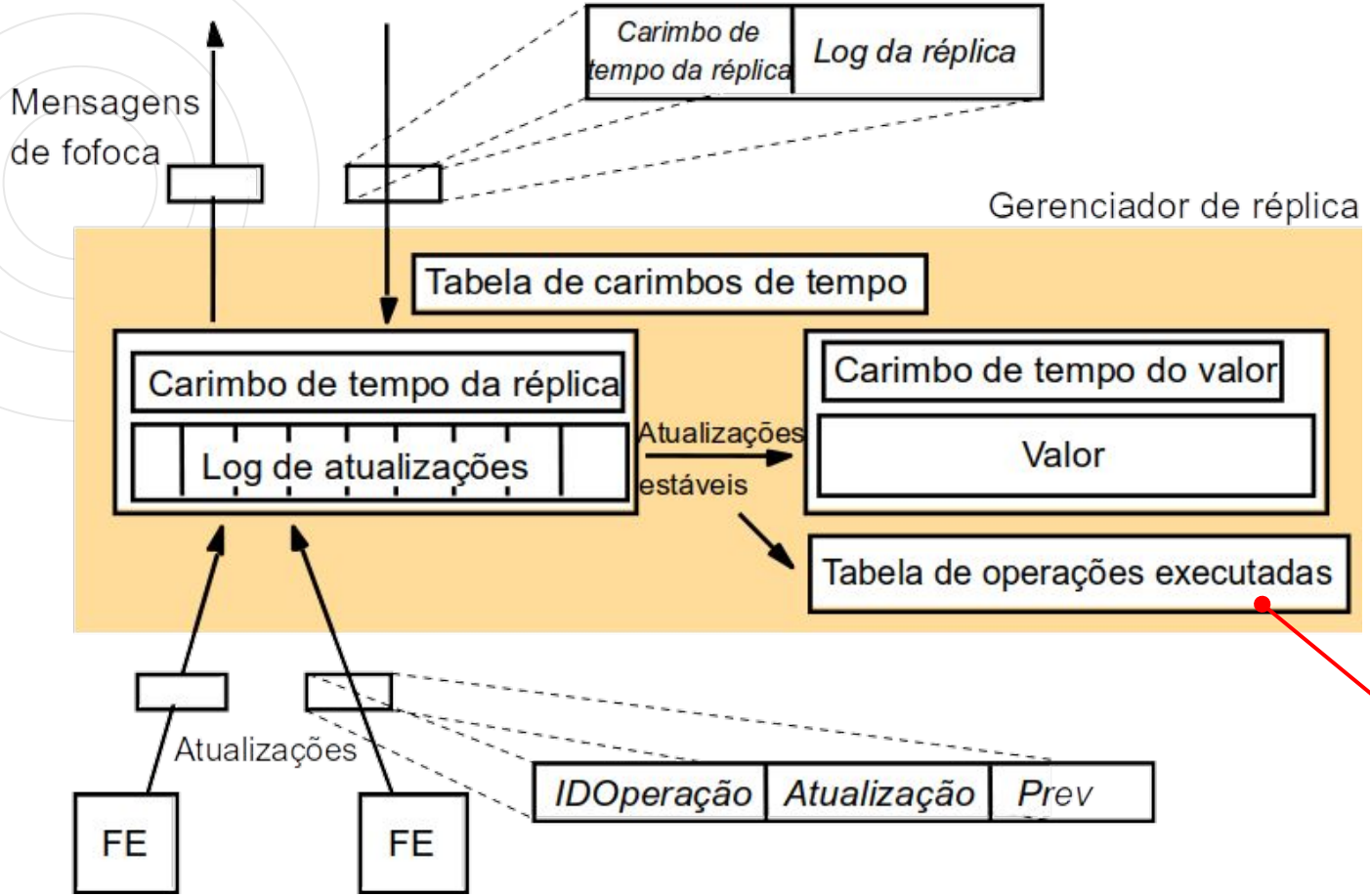


Tabela e operações executadas:
Registra as operações efetivadas (id do FE e da requisição), evitando atendimentos em duplicidade..

Arquitetura: Fofoca

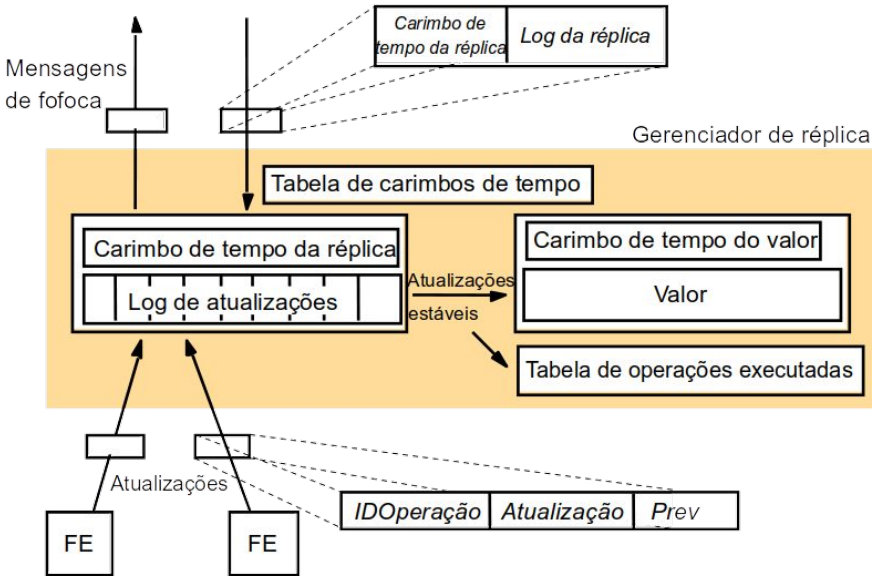
Situações: *Query*

Se **prev ≤ CarimboValor**, retorna o dado, caso contrário, mantém a requisição em espera até a condição ser atendida. Exemplo:

prev = (2,4,6)
CarimboValor = (2,5,5)

Como **6 > 5**, o dado no RM está desatualizado em relação ao RM2

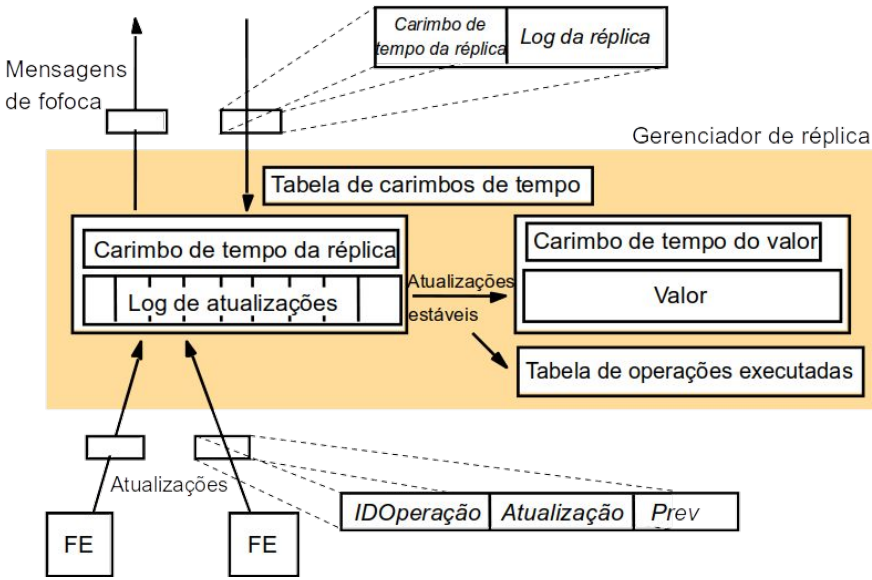
Nesta situação, o pedido vai ficar suspenso até que uma fofoca seja recebida e atualize o dado. Alternativamente, o próprio RM pode solicitar, pois ele sabe que, no caso, o RM2 é quem está "devendo" uma boa história.



Arquitetura: Fofoca

Situações: *Update*

As requisições possuem um **id**, mesmo se a mesma requisição for enviada pelo FE a vários RM simultaneamente. Ao receber uma requisição de um FE, o RM **i**, verifica se ela já foi atendida (na **tabela de operações executadas**), então descarta. Caso contrário, incrementa a **i-ésima** posição da tabela de carimbos de tempo. É realizado o registro no log de **<i, TabelaCarimboTempo,op,prev,id>**.
Então:
TabelaCarimboTempos[i] = CarimboTempoRéplica e **∀ j ≠ i TabelaCarimboTempo[j] = prev[j]**.
Quando prev ≤ CarimboTempoValor, é encontrado um estado de estabilidade do dado, então o valor é atualizado, a **CarimboTempoValor[k] = max(CarimboTempoValor[k],TabelaCarimboTempo[k])** e a operação é armazenada na **tabela de operações executadas**

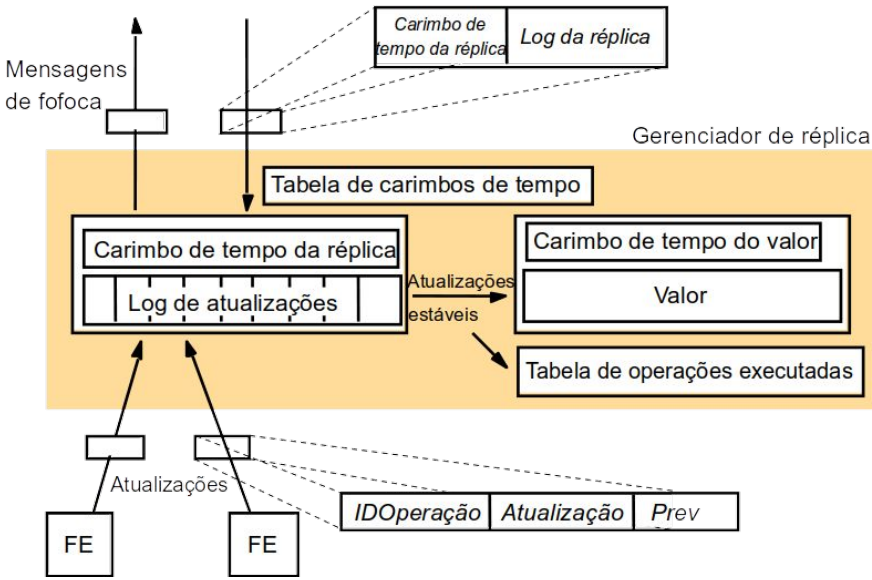


Arquitetura: Fofoca

Situações: *Fofoca*

Diferentes políticas podem ser utilizadas para envio de mensagens de fofoca e a frequência deve levar em consideração os tempos de atualização dos dados e o tamanho do sistema. Portanto, é dependente do caso.

A mensagem de fofoca contém o log de operações realizadas e o carimbo de tempo da réplica. Ao receber uma fofoca, um RM atualiza seu log, inserindo as operações que não realizou, aplica os updates estáveis.



Obrigado!