



Transparência em Sistemas Distribuídos

Fazendo o Complexo Parecer Simples



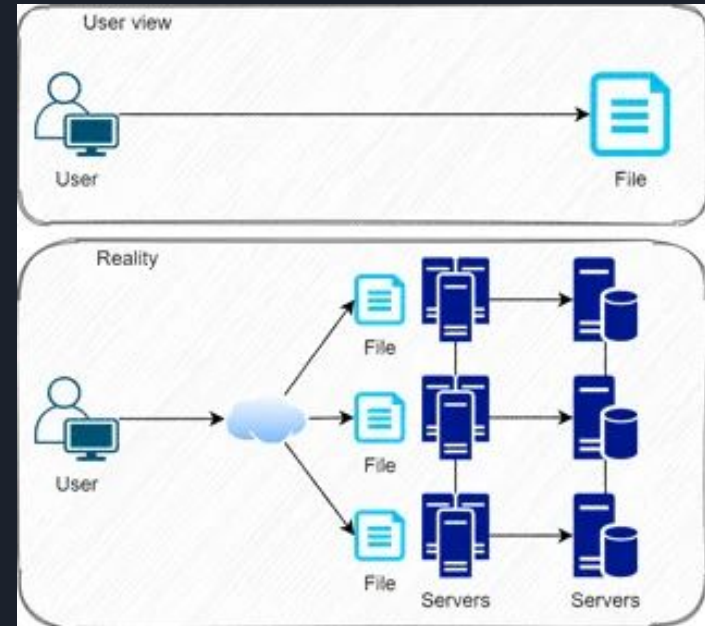
Definição e Importância

- Transparência: ocultar a complexidade de um sistema distribuído do usuário e do desenvolvedor.
- Objetivo: tornar o sistema **simples de usar**, mesmo sendo composto por vários computadores.
- Importância:
 - Melhora a **usabilidade**.
 - Facilita o **desenvolvimento**.
 - Aumenta a **portabilidade** das aplicações.

Tipos de Transparência

1. Acesso

- Definição: Oculta as diferenças na representação de dados e nos mecanismos de acesso a um recurso.
- Exemplo: Ler um arquivo via NFS como se fosse local.



2. Localização

- Definição: Oculta onde um recurso (dado, serviço, etc.) está fisicamente localizado. Os recursos são acessados por nome, sem que sua localização física precise ser conhecida.
- Exemplo: DNS permite acessar sites sem saber o IP real nem onde o servidor está localizado.

LOCATION TRANSPARENCY: Access by Name, Not Address

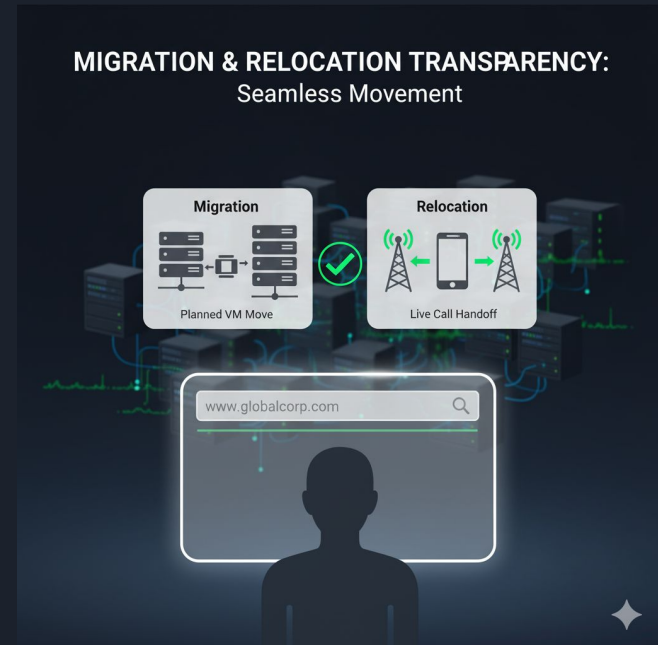


3. Migração

- Definição: Garante que um recurso possa ser movido de um local para outro no sistema **sem que os usuários ou as aplicações percebam**.
- Exemplo: Provedores de nuvem como Google Cloud, AWS e Azure movem VMs em execução de um servidor físico para outro para realizar manutenção de hardware ou para balancear a carga.

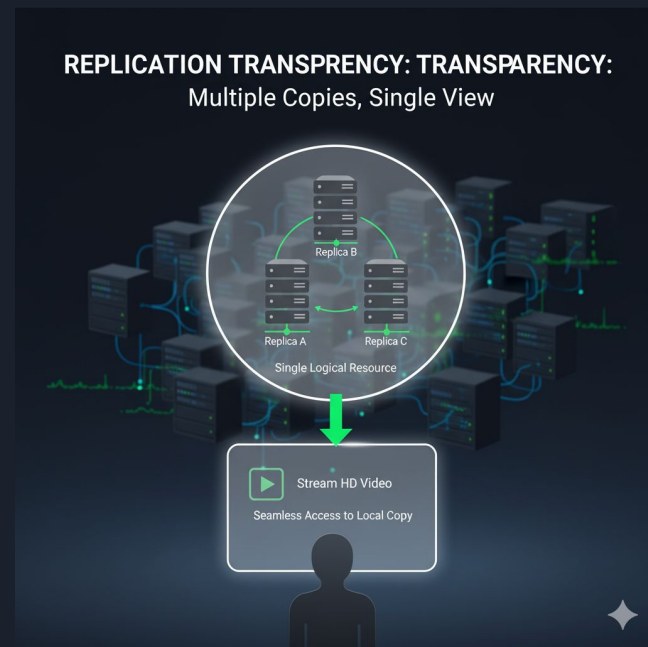
4. Relocação

- Definição: Oculta a mudança de recurso enquanto está em uso.
- Exemplo: **Redes de Telefonia Móvel**.



5. Replicação

- Definição: Oculta que um recurso tem múltiplas cópias.
- Exemplo: Bancos de dados replicados no Google Cloud, Serviços de streaming como Netflix.



6. Concorrência

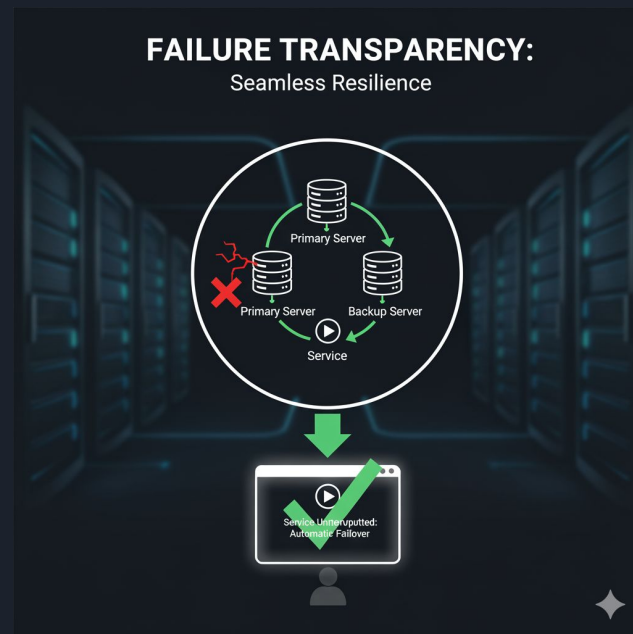
- Definição: Permite que múltiplos usuários acessem e modifiquem recursos compartilhados simultaneamente, sem interferir uns nos outros ou corromper os dados.
- Exemplo: Google Docs permitindo edição colaborativa em tempo real.

CONCURENCY TRANSPRENCY: TRANSPARENCY: Harmononous Sitmainaloous Access



7. Falha

- Definição: Oculta falhas e recupera o sistema automaticamente.
- Exemplo: Netflix usa servidores redundantes para continuar funcionando mesmo com falhas.



Benefícios da Transparência

Benefícios para **Usuários**

Simplicidade e Usabilidade: Interação direta com o serviço, sem a complexidade da infraestrutura.

Confiabilidade: O sistema continua disponível mesmo com falhas parciais (Transparência de Falha).

Performance: Acesso mais rápido a recursos através de réplicas locais (Transparência de Replicação).

Mobilidade: Capacidade de usar serviços em movimento sem interrupções (Transparência de Relocação).

Benefícios para **Desenvolvedores**

Produtividade Aumentada: Foco na lógica de negócio em vez de problemas de baixo nível da rede.

Portabilidade: Aplicações podem ser movidas para diferentes ambientes sem modificação (Transparência de Migração).

Escalabilidade Simplificada: É mais fácil adicionar novos recursos e usuários sem reescrever a aplicação.

Manutenção Facilitada: Recursos podem ser atualizados ou movidos sem impactar as aplicações em execução.



Limitações

- Transparência total nem sempre é possível.
- Pode gerar **sobrecarga de comunicação**.
- Impacto em **latência e desempenho**.
- Trade-off: simplicidade vs. eficiência.



OBRIGADO!