### ****Consistência Forte vs. Consistência Eventual****

**Consistência Forte:**  
Quando um sistema garante que, assim que você grava um dado, todos os usuários veem a mesma versão atualizada, independente de onde ou como eles acessam. É como uma fila de bilheteria: você compra o ingresso, e imediatamente ninguém mais consegue aquele mesmo lugar.

**Exemplo:**  
Imagine um sistema de controle de vagas para um curso online onde só há 1 vaga restante. Se duas pessoas tentarem se inscrever ao mesmo tempo, a consistência forte assegura que apenas uma delas conseguirá, sem chances de erro ou conflito. Isso é essencial em sistemas financeiros, venda de ingressos ou controle de estoque.

**Consistência Eventual:**  
Aqui, o sistema aceita atrasos na sincronização, permitindo que diferentes partes do sistema vejam versões diferentes temporariamente, mas eventual **convergência**. É como amigos compartilhando uma playlist: nem todo mundo vê as músicas adicionadas ao mesmo tempo, mas eventualmente, todos terão a mesma lista.

**Exemplo:**  
Considere um aplicativo de redes sociais onde você comenta numa foto. Pode demorar alguns segundos para o comentário aparecer para seus amigos, mas isso não impede a funcionalidade principal. Esse modelo funciona bem em sistemas de alta escala, como redes sociais e serviços de streaming, onde a rapidez é mais importante do que a precisão imediata.

### ****Eventos de Dados vs. Eventos de Aplicação****

**Eventos de Dados:**  
Focado em mudanças nos **dados**: criação, atualização ou exclusão. É como receber uma notificação porque alguém atualizou sua ficha médica. O evento em si não carrega o "motivo", só a mudança.

**Quando usar:**  
Ideal para sistemas onde outros serviços precisam ser notificados sobre alterações, mas não importa o contexto, como replicação de banco de dados ou sincronização entre sistemas.

**Eventos de Aplicação:**  
Refletem ações ou **intenções do negócio**, como "compra realizada" ou "pedido cancelado". É mais semântico, dizendo **por que algo aconteceu**.

**Quando usar:**  
Perfeito para arquiteturas orientadas a eventos, como um e-commerce notificando diferentes microsserviços (estoque, entrega, pagamentos) após uma venda.

**Analogia simples:**

* + Dados: "Nome do cliente mudou para João".
  + Aplicação: "Cliente atualizou seu perfil".

### ****Métricas Importantes para um Microsserviço****

**Latência:**  
Quanto tempo o serviço leva para processar uma requisição. Se for um microsserviço de emissão de boletos, ninguém quer esperar 2 minutos por uma resposta.

**Taxa de Sucesso:**  
Quantos pedidos foram processados corretamente em relação ao total. Para sistemas críticos, como emitir passagens aéreas, a confiabilidade é tudo.

**Taxa de Erros:**  
A frequência de falhas, como timeouts, respostas inválidas ou indisponibilidade.

**Throughput (Vazão):**  
Quantas requisições o serviço aguenta processar por segundo, especialmente em horários de pico.

**Ponto de Vista Criativo:**  
Imagine que esse microsserviço é um robô em um armazém que empacota pedidos. Se ele demora para processar um pedido (latência), falha em reconhecer itens (erros) ou não consegue lidar com muitos pedidos ao mesmo tempo (throughput), todo o sistema desanda.