FUNDAMENTOS DO REDIS

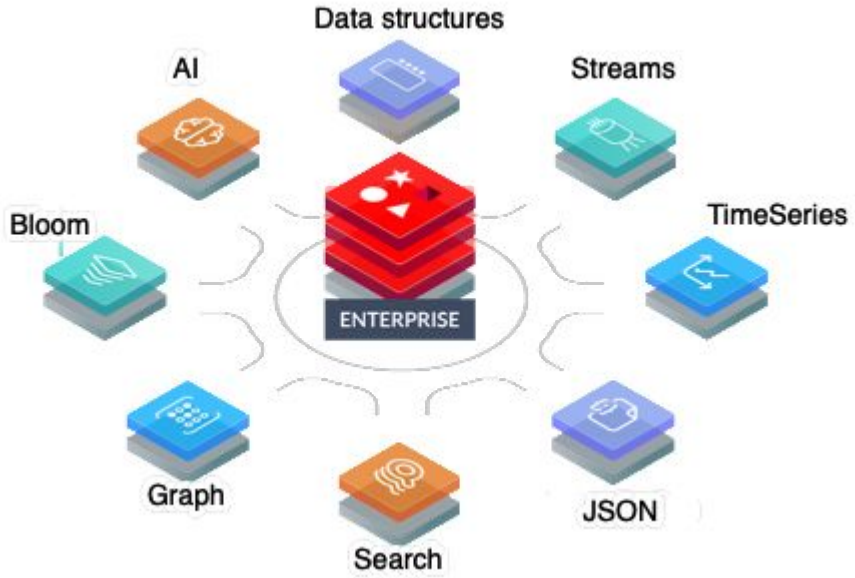
Propósito e Aplicabilidade

Discentes:Yuri Matheus Sousa dos Santos.
Docente:Adriano Antunes Prates.
Data: 26/11/2025

Conceito e Propósito

- O que é o Redis?

Figura 1 – Arquitetura de extensibilidade do Redis via módulos.



Fonte: Adaptado de Kondapalli (2021).

INSTITUTO FEDERAL
Norte de Minas Gerais

Conceito e Propósito

- Quais conceitos de SD ele carrega?
  - Compartilhamento de recursos
  - Nomenclatura
  - comunicação
  - replicação e consistência
  - tolerância a falhas
- Principais Estruturas
  - Strings (Cadeias de Caracteres)
  - Lists (Listas Ligadas)
  - Sets (Conjuntos)
  - Hashes (Mapas ou Dicionários)
  - Sorted Sets (Conjuntos Ordenados ou ZSets)
  - Streams (Fluxos)
  - Geospatial Indexes (Índices Geoespaciais)

Vantagens e Limitações

- Desempenho
  - Escalabilidade Geográfica e Baixa Latência (Vantagem)
  - Problemas de Consistência (Desvantagem)
- Disponibilidade
  - Dependabilidade e Redundância(Vantagem)
  - Complexidade de Coordenação(Desvantagem)

Banco de dados em Memória vs Tradicional

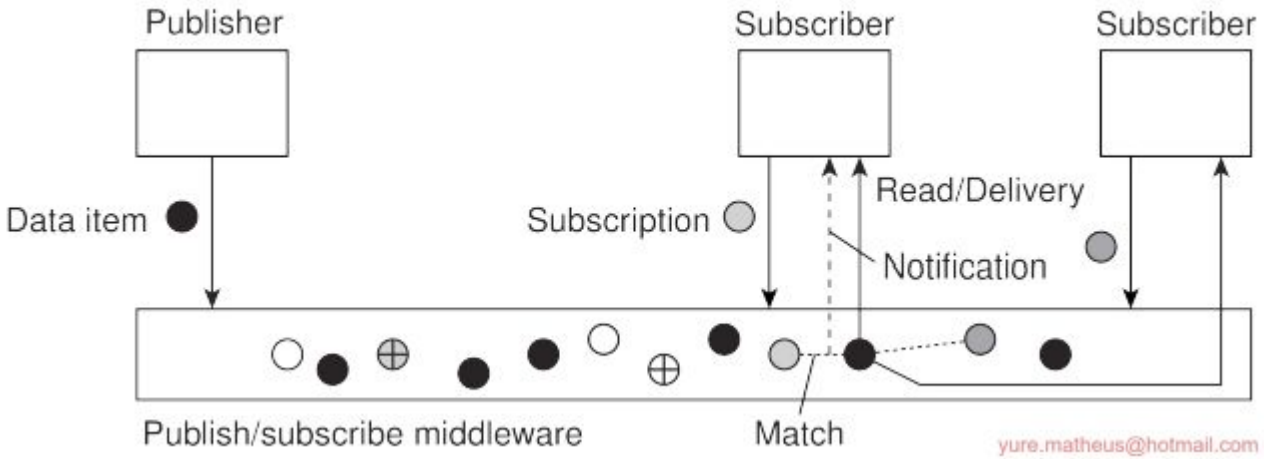
- Cenário Distribuído
  - Desempenho(Memória)
    - Resolve a limitação da capacidade computacional e de armazenamento (incluindo a taxa de transferência I/O)
  - Resiliência e Persistência(Tradicional)
    - focam inerentemente em persistência e recuperação (Recovery)
    - o Redis depende de mecanismos explícitos de persistência (como RDB e AOF) e redundância

Redis como: cache distribuído, armazenamento de dados volátil/persistente e mecanismo de mensageria

- Cache Distribuído (Distributed Cache)
  - caching é uma forma especial de replicação
- Armazenamento de Dados Volátil/Persistente
  - Volatilidade (Soft State): Onde a perda de dados é tolerável e pode ser reconstruída.
  - Persistência (Permanent State): Para dados que precisam de estado permanente (permanent state)
- Mecanismo de Mensageria (Mecanismo de Mensageria)
  - Comunicação Orientada a Mensagens
  - Arquitetura Publish-Subscribe: O Redis implementa o estilo arquitetural Publish-subscribe (Publicação-Assinatura)
  - Comunicação Persistente e Assíncrona: A comunicação pode ser persistente, onde uma mensagem é armazenada pelo middleware de comunicação até ser entregue

Redis como: Mecanismo de mensageria

Figura 2 – Princípio de troca de dados entre publicadores e assinantes em uma arquitetura publish-subscribe.



Fonte: Adaptado de Van Steen e Tanenbaum (2023, p. 72).

Redis como: Mecanismo de mensageria(Broker)

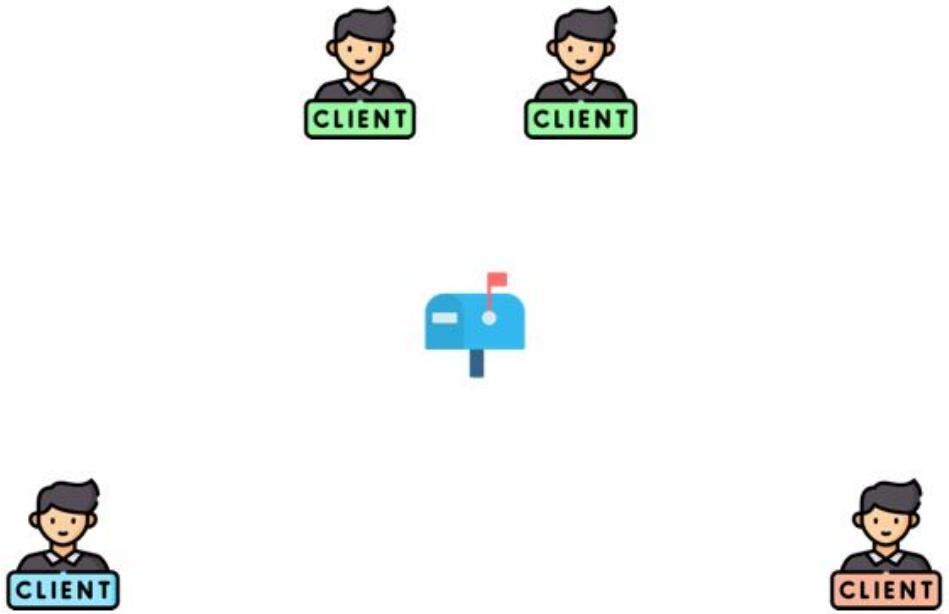
Figura 3 – Analogia da transmissão de rádio ilustrando o desacoplamento entre publicador (emissor) e assinantes (ouvintes) no padrão Publish-Subscribe.

Fonte: De Lio (2023).

INSTITUTO FEDERAL
Norte de Minas Gerais

Redis como: Mecanismo de mensageria(Pub/Sub)

Figura 4 – Arquitetura do mecanismo Pub/Sub no Redis, onde um cliente Publicador envia uma mensagem a um canal e o servidor a distribui para múltiplos clientes Assinantes.



Fonte: De Lio (2023).

INSTITUTO FEDERAL
Norte de Minas Gerais

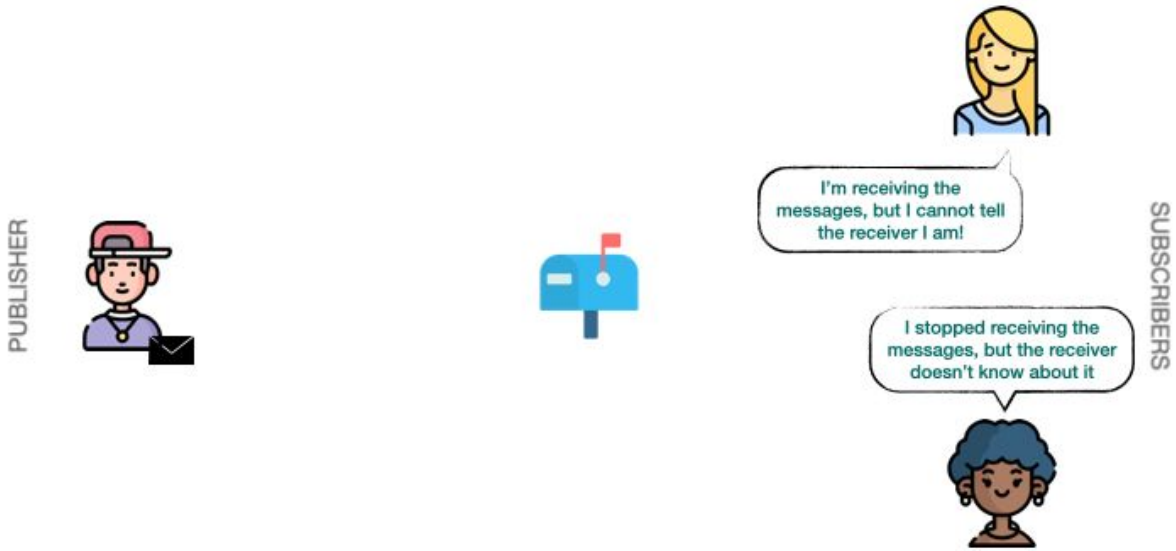
Redis como: Mecanismo de mensageria (Síncrono/Temporalmente Acoplado)

Figura 5 – Demonstração da natureza síncrona do Redis Pub/Sub, ilustrando que as mensagens são perdidas se o assinante não estiver conectado no momento da publicação.

Fonte: De Lio (2023).

Redis como: Mecanismo de mensageria(Dispare e Esqueça)

Figura 6 – Ilustração do padrão de comunicação "Disparar e Esquecer" (Fire-and-Forget), onde o publicador envia a mensagem sem esperar por confirmação de entrega.



Fonte: De Lio (2023).

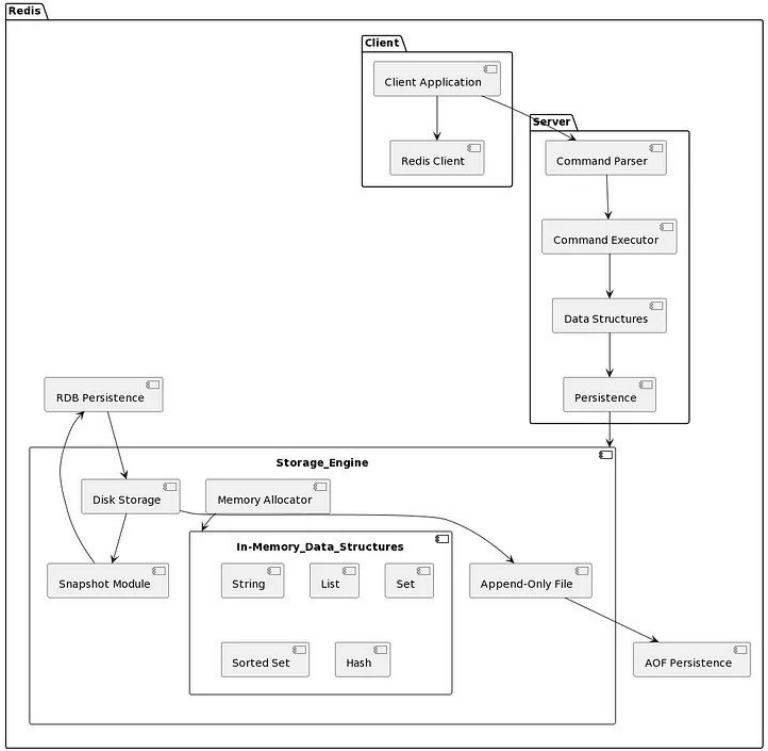
INSTITUTO FEDERAL
Norte de Minas Gerais

Arquitetura e Funcionamento

- Design pattern
    - Reactor Pattern (Padrão Reator)
        i. Fluxo de Eventos
        ii. Registro: Um novo cliente se conecta e o Handler é registrado no Dispatcher.
        iii. Espera: O Reactor usa chamadas de sistema eficientes como epoll no Linux
        iv. Despacho: O Dispatcher encaminha o evento para o Handler registrado para aquele socket.
        v. Processamento (Non-Blocking): O Handler lê os dados imediatamente (operação não bloqueante) e agenda a próxima ação (por exemplo, "quando o socket estiver pronto para escrita, envie a resposta")
    - Redis’s Single-Threaded Design

Arquitetura e Funcionamento(Suporte a Persistência)

Figura 7 – Diagrama de componentes da arquitetura Redis, ilustrando a interação entre o cliente, o servidor (com seu pipeline de processamento de comandos) e o motor de armazenamento (storage engine) com os módulos de persistência RDB e AOF.

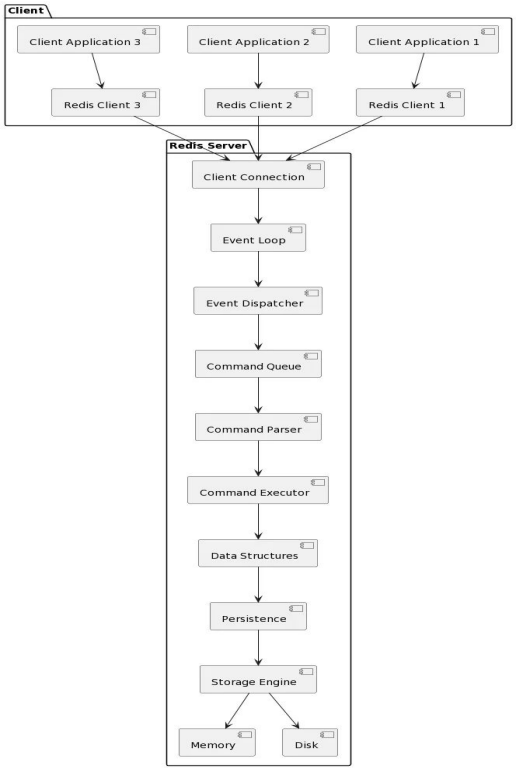


Fonte: Kashyap (2023).

INSTITUTO FEDERAL
Norte de Minas Gerais

Arquitetura e Funcionamento(Loop de Eventos)

Figura 8 – Analogia da transmissão de rádio ilustrando o desacoplamento entre publicador (emissor) e assinantes (ouvintes) no padrão Publish-Subscribe.

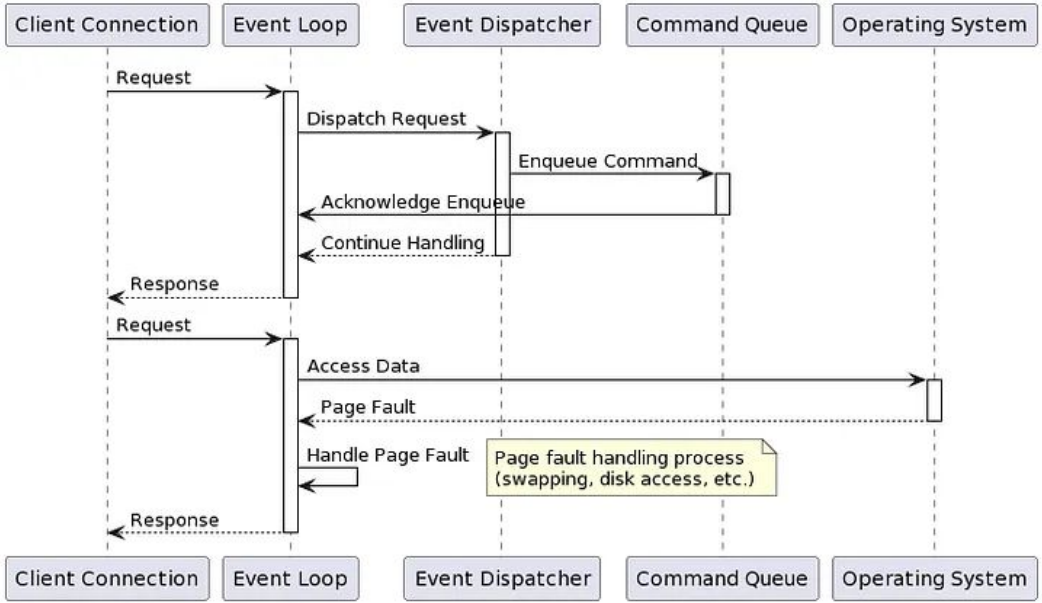


Fonte: Kashyap (2023).

INSTITUTO FEDERAL
Norte de Minas Gerais

Arquitetura e Funcionamento(Loop de Eventos-2)

Figura 9 – Diagrama de sequência do ciclo de eventos (event loop) do Redis, demonstrando o processamento de uma requisição em memória (fluxo superior) e o impacto de uma falha de página (page fault) que exige interação com o sistema operacional (fluxo inferior).

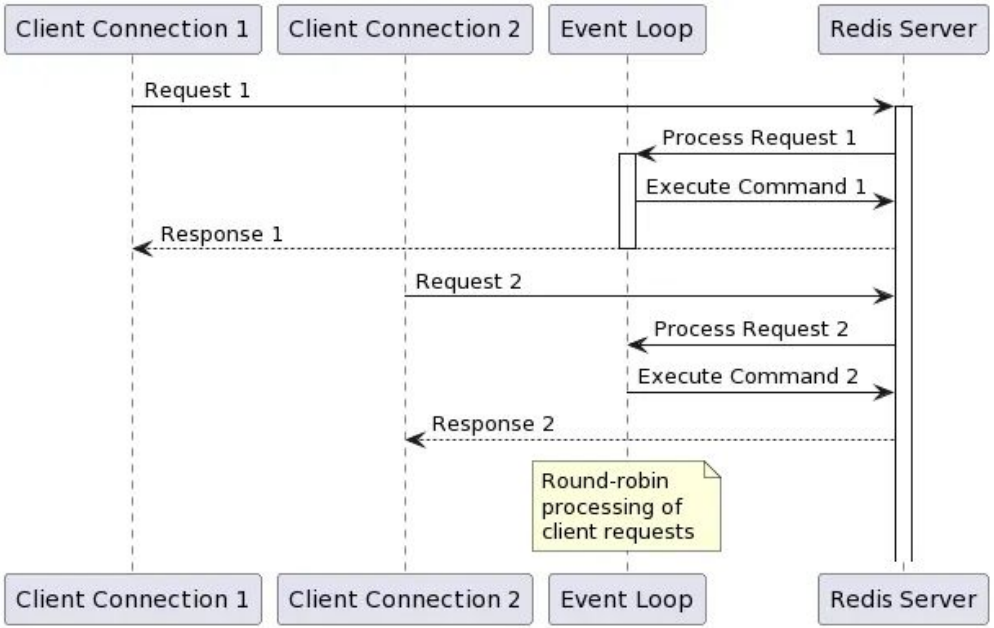


Fonte: Kashyap (2023).

INSTITUTO FEDERAL
Norte de Minas Gerais

Arquitetura e Funcionamento(Loop de Eventos-3)

Figura 10 – Fluxograma da arquitetura cliente-servidor do Redis, detalhando o pipeline sequencial de processamento de comandos, desde a conexão de múltiplos clientes até o armazenamento final em memória ou disco pelo storage engine.



Fonte: Kashyap (2023).

Políticas de Remoção de Dados (Eviction Policies)

1. Proibição de Despejo (noeviction)
   a. “De acordo com a política de não remoção de dados, o Redis não removerá nenhum dado quando o limite de memória for atingido” (DAKOJU, 2022).
2. Volatile-LRU (volatile-lru)
   a. “Remove as chaves menos recentemente usadas que possuem um tempo de expiração definido(DAKOJU, 2022).
3. Allkeys-LRU” (allkeys-lru)
   a. “Remove as chaves menos usadas recentemente, independentemente de terem ou não um tempo de expiração definido” (DAKOJU, 2022).
4. Volátil-Aleatório (volatile-random)
   a. “Remove uma chave aleatória que possui um tempo de expiração definido” (DAKOJU, 2022).
5. Allkeys-Random (allkeys-random)
   a. “Remove uma chave aleatória, independentemente de ela ter um tempo de expiração definido” (DAKOJU, 2022).
6. Volatile-TTL (volatile-ttl)
   a. “Remove a chave com o tempo de expiração mais próximo (ou seja, aquela que expirará em breve)” (DAKOJU, 2022).
7. Volatile-LFU (volatile-lfu)
   a. “Remove as chaves menos usadas que possuem um tempo de expiração definido” (DAKOJU, 2022).
8. Allkeys-LFU (allkeys-lfu)
   a. “Remove as chaves menos usadas, independentemente de terem ou não um tempo de expiração definido” (DAKOJU, 2022).

Referências

BUJARRA, Xavier. Ollama, começando com a IA local! **Bujarra.com**, 17 jan. 2024. Disponível em: https://www.bujarra.com/ollama-empezando-con-la-ia-local/?lang=pt. Acesso em: 14 dez. 2025.

CLEVER CLOUD. **Redis.** [S. l.], 2025. Disponível em: https://www.clever.cloud/product/redis/. Acesso em: 14 dez. 2025.

DAKOJU, Srujana. Redis Cache Eviction Strategies. **Medium**, 23 nov. 2022. Disponível em: https://medium.com/@srujana.dakoju/redis-cache-eviction-strategies-b806c67e37c0. Acesso em: 14 dez. 2025.

KASHYAP, Vaibhav. Redis: Unveiling the Architecture and Functionality. **Medium**, 20 jul. 2023. Disponível em: https://codechefvaibhavkashyap.medium.com/redis-unveiling-the-architecture-and-functionality-66134434de90. Acesso em: 14 dez. 2025.

KONDAPALLI, Venkata Siva Sankara Rao. What is a Redis Module? **LinkedIn**, 1 ago. 2021. Disponível em: https://www.linkedin.com/pulse/what-redis-module-venkata-siva-sankara-rao-kondapalli/. Acesso em: 14 dez. 2025.

TORUÑO, Carlos. Streamlit for Data Apps | Part 1: Getting Started. **Carlos Toruño Blog**, 25 jun. 2023. Disponível em: https://www.carlos-toruno.com/blog/streamlit/01-intro/. Acesso em: 14 dez. 2025.

VAN STEEN, Maarten; TANENBAUM, Andrew S. **Distributed Systems.** 4. ed. [s.l.]: Maarten van Steen, 2023. Disponível em: https://www.distributed-systems.net. Acesso em: 14 dez. 2025.

Demonstração Prática de Aplicação(Redis como cache Semântico)

Figura 11 – Tecnologias utilizadas no desenvolvimento da aplicação de demonstração: (a) Logotipo do Redis, utilizado como cache distribuído e memória de conversação; (b) Logotipo do Ollama, framework para execução local do modelo de linguagem; (c) Logotipo do Python, linguagem de programação base do projeto; (d) Logotipo do Streamlit, framework para a construção da interface web.

a)

b)

c)

Fonte: Adaptado de Clever Cloud (2025), Bujarra (2024) e Toruño (2023).