INSTITUTO DE INFORMÁTICA
Universidade Federal de Goiás

Design de Software aplicado a Big Data

Prof. Dr. Jacson Rodrigues Barbosa
jacson@inf.ufg.br

Alisson Braz
Arthur Felipe
Beatriz Menezes
Jannderson Oliveira

2025

INSTITUTO DE INFORMÁTICA
UFG

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

O que é Big Data?

Definição:

Conjunto de dados massivos e complexos, difíceis de processar com ferramentas tradicionais.

Os 5 V's do Big Data:

- **Volume:** Grande quantidade de dados gerados constantemente.
- **Velocidade:** Rapidez com que os dados são criados e processados.
- **Variedade:** Dados provenientes de varias fontes diferentes.
- **Veracidade:** Confiabilidade e qualidade dos dados.
- **Valor:** Utilidade das informações extraídas.

Qual a importância de um design adequado para Big Data?

- **Escala** corretamente sem travar;
  - Arquiteturas bem definidas (ex: **microserviços, pipelines distribuídos**) permitem que a aplicação processe **terabytes ou petabytes** de dados sem perda de performance.
- **Entrega** performance em alto volume;
  - Escolhas de arquitetura (batch vs. streaming, índices, particionamento, compressão) impactam diretamente o **tempo de resposta**.
- Facilita **manutenção** e evolução;
  - Projetos Big Data envolvem pipelines complexos (ETL, processamento em tempo real, data lakes, APIs). Sem design modular e claro (camadas bem separadas), fica quase impossível dar manutenção.
- Garante qualidade, **confiabilidade** e segurança dos dados;
  - Um sistema mal desenhado pode gerar **dados inconsistentes ou duplicados**. O design deve prever **tolerância a falhas, retries, idempotência e consistência eventual**.
- Reduz **custos** de operação em nuvem/infraestrutura.
  - Um design ruim pode gerar **armazenamento duplicado, queries ineficientes e uso excessivo de cluster**

O Jogo Smart Decisions

- Breve explicação do jogo: simula escolhas de design em Big Data.
- Objetivo: mostrar trade-offs arquiteturais.
- Relevância: aplicar pensamento crítico em escolhas de software.

O Jogo Smart Decisions

	Iteration #1	Iteration #2	Iteration #3	Iteration #4	Iteration #5	
(a) Design Decisions *(Names of selected design concept(s))*						
(b) Driver selection points *(from cards)*						
(c) Instantiation points *(from dice)*						
(d) Analysis bonus points *(from review)*						**Final score:**
(e) Iteration total *(b + c + d)*						

Figure 4. The game scorecard

Table 1. The design concepts catalog included in the Big Data game content

Type of design concept	Design concept card
Reference Architectures	• Data Refinery • Extended relational • Lambda Architecture • Pure non-relational
Patterns / Technology Families	• Column-Family NoSQL Database • Data Collector • Distributed File System • Distributed Message Broker • Distributed Search Engine • Document-Oriented NoSQL Database • Interactive Query Engine
Specific technologies	• Apache Kafka • RabbitMQ • Amazon SQS • Apache ActiveMQ • Apache Flume • Fluentd • Logstash • Apache HBase • Apache Cassandra • Apache CouchDB • Mongo DB • Apache Hive • Impala • Spark SQL • Apache Solr • Elasticsearch • Splunk

Pilares de Design em Big Data

- Escalabilidade (horizontal vs vertical).
- Disponibilidade (alta disponibilidade, replicação).
- Consistência (CAP theorem).
  - Consistência;
  - Disponibilidade;
  - e, tolerância a partições (falhas de rede)
- Latência (processamento em batch vs real-time).

Arquiteturas Comuns

- Data Warehouse tradicional.
- Data Lake.
- Lambda Architecture (batch + streaming).
- Kappa Architecture (streaming only).

Tecnolgias Envolvidas

Bancos NoSQL

- **Cassandra, MongoDB, HBase**: Soluções escaláveis para grandes volumes de dados, com diferentes níveis de consistência e flexibilidade no modelo de dados.

Frameworks de Processamento

- **Hadoop, Spark, Flink**: Plataformas para processamento distribuído de grandes volumes de dados, com suporte a batch (Hadoop, Spark) e streaming em tempo real (Spark, Flink).

Mensageria/Streaming

- **Kafka**: Sistema de mensageria distribuído, essencial para pipelines de dados em tempo real e arquiteturas orientadas a eventos.

Armazenamento Distribuído

- **S3, HDFS**: Soluções robustas e escaláveis para armazenamento de grandes volumes de dados, com suporte a redundância e alta disponibilidade.

CRITÉRIOS DE DECISÃO

Custos vs Benefícios

Tamanho da Equipe e Curva de Aprendizado

Requisitos de Latência e Disponibilidade

Flexibilidade e Manutenibilidade

Exemplo de Trade-off (Inspirado no Jogo)

Alta consistência → Menor desempenho
Garantir que todos os nós tenham os mesmos dados pode impactar a performance.

Baixa latência → Maior custo de infraestrutura
Reduzir o tempo de resposta exige sistemas mais robustos e, consequentemente, mais caros.

Processamento em batch → Menor custo, sem dados em tempo real
Processamento em lote é mais econômico, mas não atende aplicações que dependem de dados atualizados instantaneamente.

Boas Práticas de Design para Big Data

Arquitetura Escalável e Flexível:

- Pense em Microserviços.
- Adote Arquiteturas Orientadas a Dados (ex: Lambda, Kappa).

Governança e Qualidade de Dados

- Pipelines de validação e limpeza na ingestão.
- Catálogo de dados e metadados.

Segurança e Privacidade:

- Anonimização e pseudo-anonimização de dados.
- Controle de acesso granular (RBAC).
- Conformidade com a LGPD desde a concepção do software.

Casos de Uso na Engenharia de Software

Sistemas de Recomendação (Netflix/Spotify):

- Desafio de Design: Processar terabytes de interações de usuários (cliques, views, likes) em tempo real para gerar recomendações personalizadas com baixa latência.
- Dados: Histórico de consumo, perfil do usuário, metadados de conteúdo.

Logística e Otimização em Tempo Real (iFood/Uber):

- Desafio de Design: Ingerir e processar milhões de eventos de geolocalização por segundo para otimizar rotas, precificar dinamicamente e prever demanda.
- Dados: Posição GPS, trânsito, eventos climáticos, dados de pedidos.

Conclusão

O design de software é a resposta para os desafios do Big Data (Volume, Velocidade, etc.).

Não há uma solução única; o bom design nasce da análise de trade-offs, como custo vs. performance.

Arquiteturas como Lambda/Kappa e as boas práticas que vimos são o que viabilizam aplicações como Netflix e iFood.

No final, um design bem planejado é o que transforma um grande volume de dados em valor real para o negócio.

Obrigado

Dúvidas ou sugestões?

INSTITUTO DE INFORMÁTICA UFG

UFG UNIVERSIDADE FEDERAL DE GOIÁS