Big Data Real-Time Analytics com Python e Spark

Seja muito bem-vindo(a)!

Real-Time Analytics com Spark Streaming

Real-Time Analytics com Spark Streaming

A vida não acontece em batches!

Real-Time Analytics com Spark Streaming

- Batch vs. Real-Time Processing

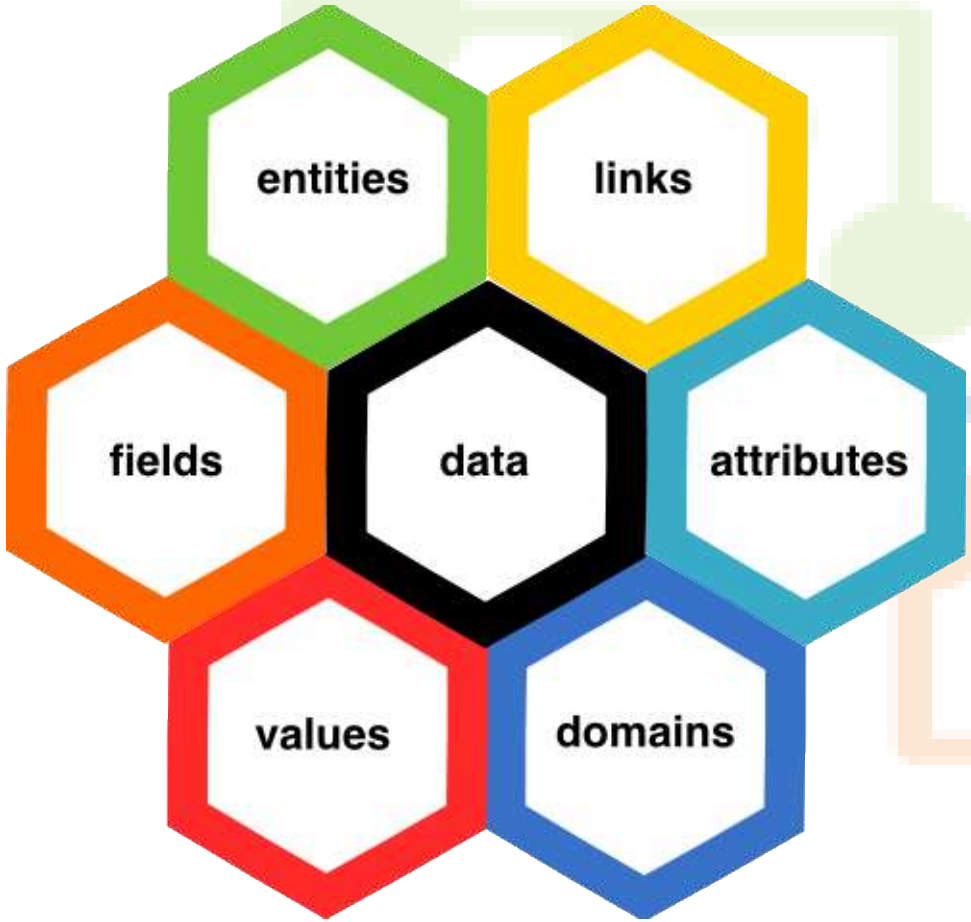
Real-Time Analytics com Spark Streaming

São muitas as vantagens na manipulação de dados como fluxos

Real-Time Analytics com Spark Streaming

Streaming de dados não é apenas para projetos altamente especializados. Computação baseada em Streaming está se tornando a regra para empresas orientadas a dados.

Real-Time Analytics com Spark Streaming



Novas tecnologias e projetos de arquitetura permitem construir sistemas flexíveis!

Real-Time Analytics com Spark Streaming

Sensores = Dados Contínuos

Real-Time Analytics com Spark Streaming

waze
OUTSMARTING TRAFFIC, TOGETHER

Real-Time Analytics com Spark Streaming

E o que vamos estudar sobe Spark Streaming?

- Arquitetura Spark Streaming
- DStreams
- Windowing
- Integração com outros sistemas – Kafka, Flume, Amazon Kinesis
- Processamento de Linguagem Natural – NLTK
- **Análise de Sentimentos do Twitter em Tempo Real (próximo capítulo)**
- **Deploy em Cluster na Nuvem (próximo capítulo)**

Big Data Real-Time Analytics com Python e Spark

Batch x Streaming

Batch x Streaming

Batch

Você inicia o processamento de um arquivo ou dataset finito, o Spark processa as tarefas configuradas e conclui o trabalho.

Streaming

Você processa um fluxo de dados contínuo (Stream); a execução não para até que haja algum erro ou você termine a aplicação manualmente.

Batch x Streaming

Batch

- Análise exploratória de dados
- Analisar dados de Data Warehouses
- Treinar um modelo de aprendizado de máquina sobre grandes conjuntos de dados
- Outras tarefas analíticas feitas com Hadoop MapReduce

Batch x Streaming

Streaming

- Monitoramento de serviços
- Processamento de eventos em tempo real para alimentar dashboards
- Processamento de dados de cliques e eventos em web sites
- Processamento de dados de sensores de Internet das Coisas
- Processamento de dados vindos de serviços como: Twitter, Kafka, Flume, AWS Kinesis

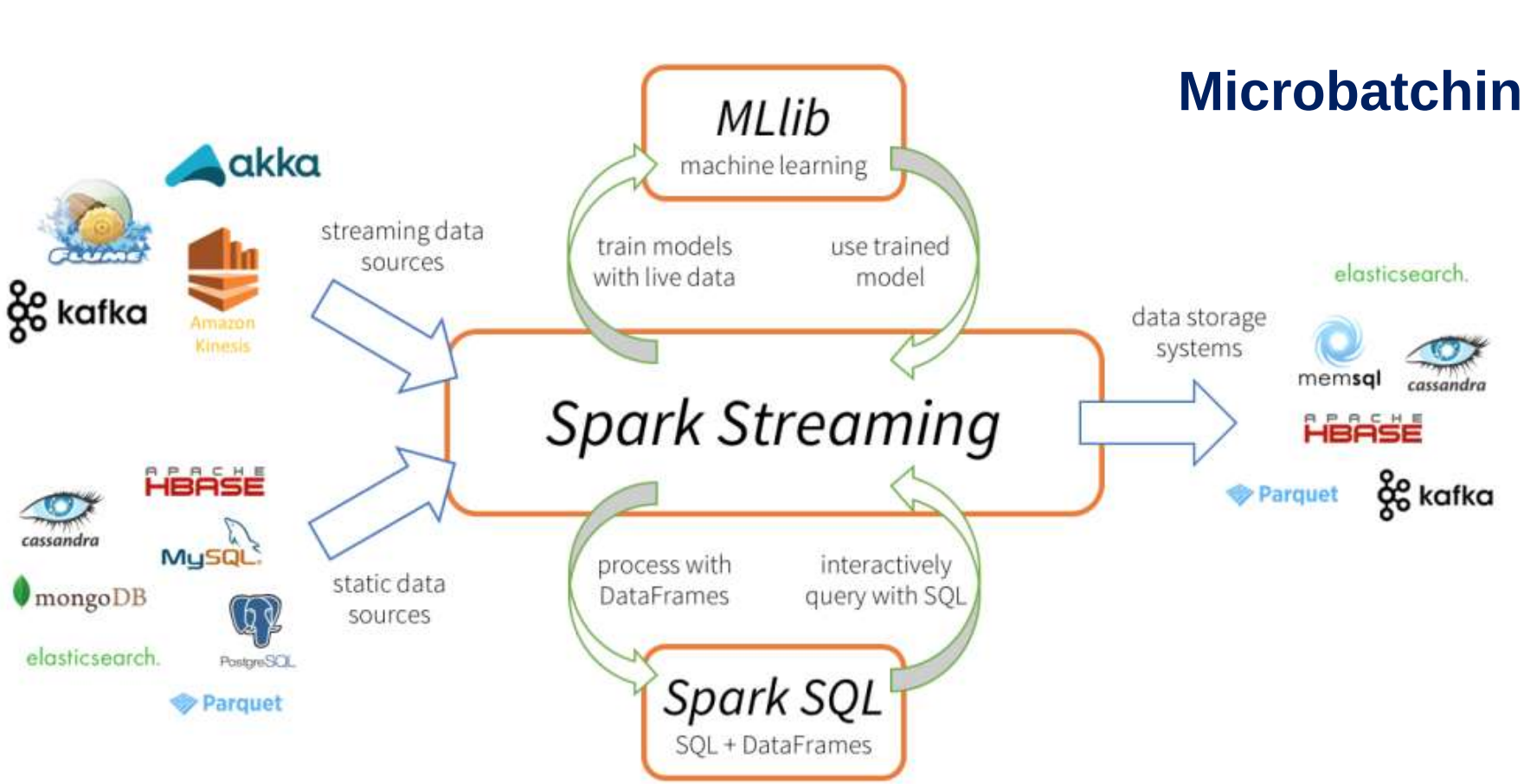
Batch x Streaming

Big Data never stops!

Big Data Real-Time Analytics com Python e Spark

Apache Spark Streaming

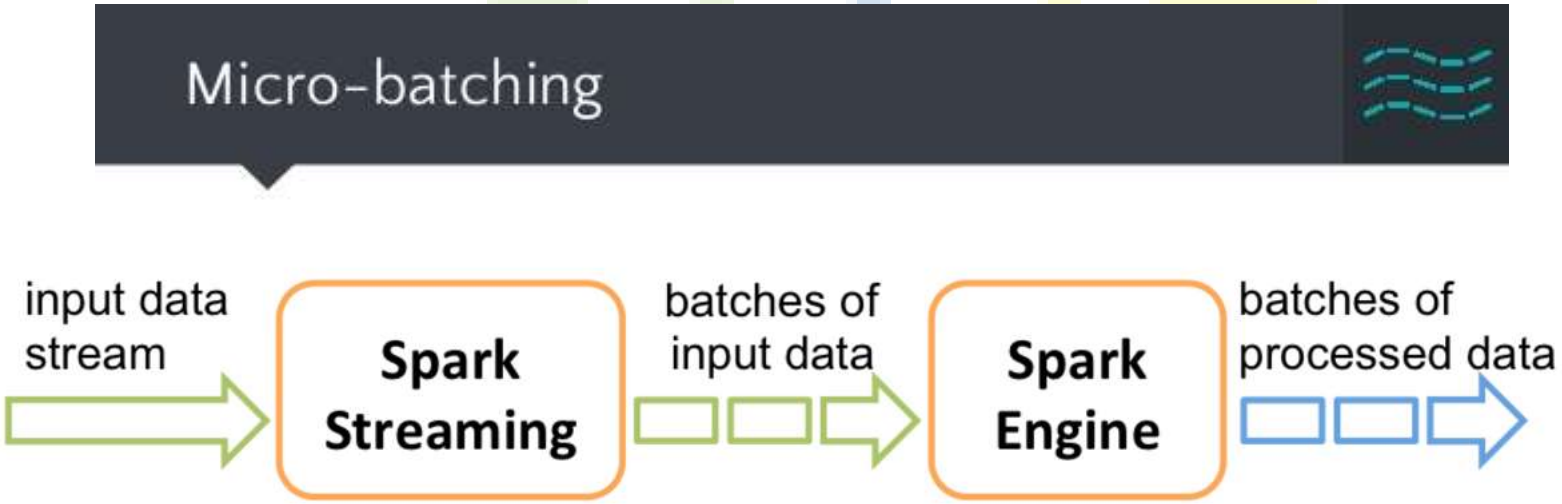
Apache Spark Streaming



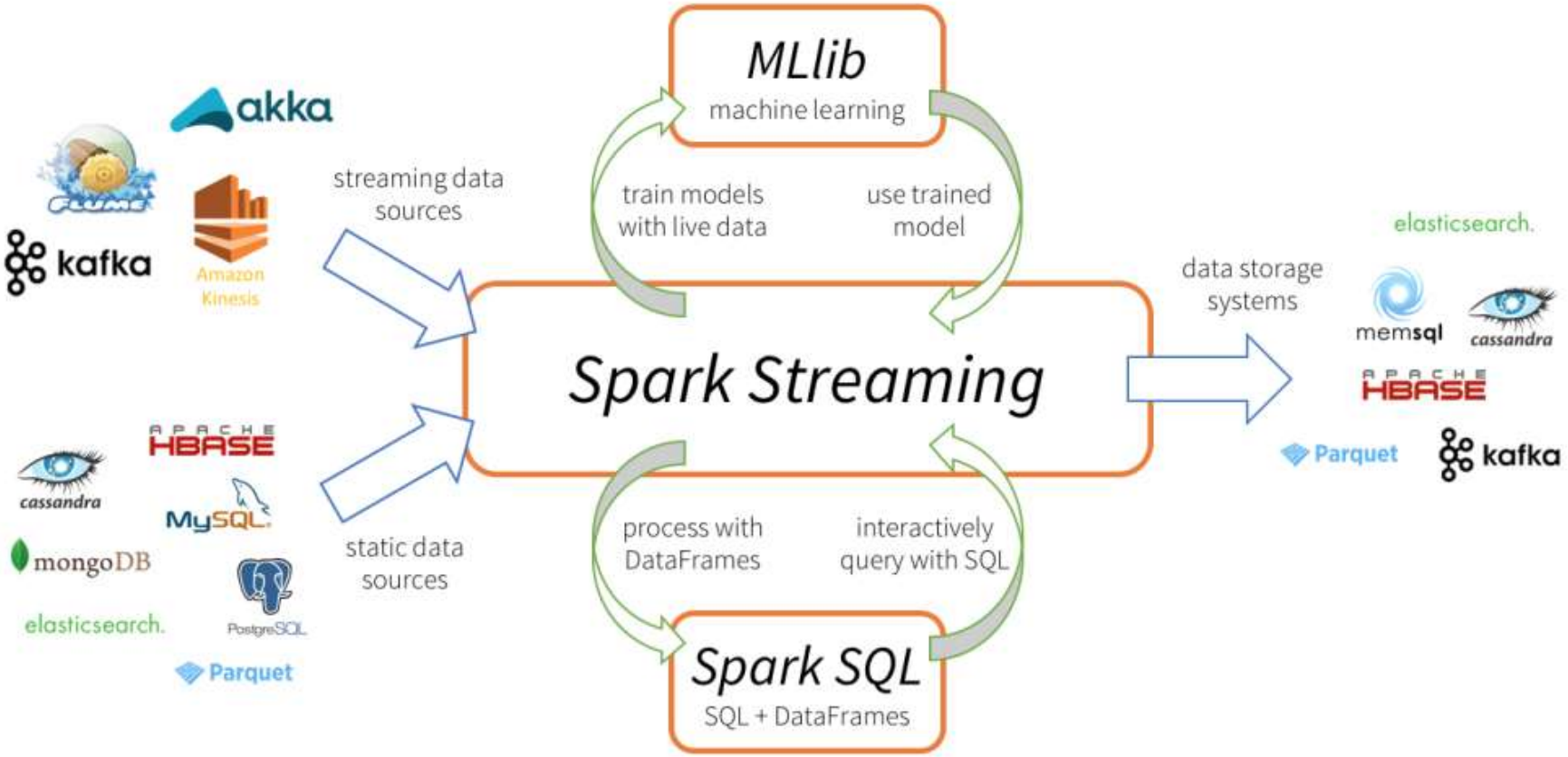
Apache Spark Streaming

Quer dizer que o Spark Streaming não é ”real” real-time?

Apache Spark Streaming



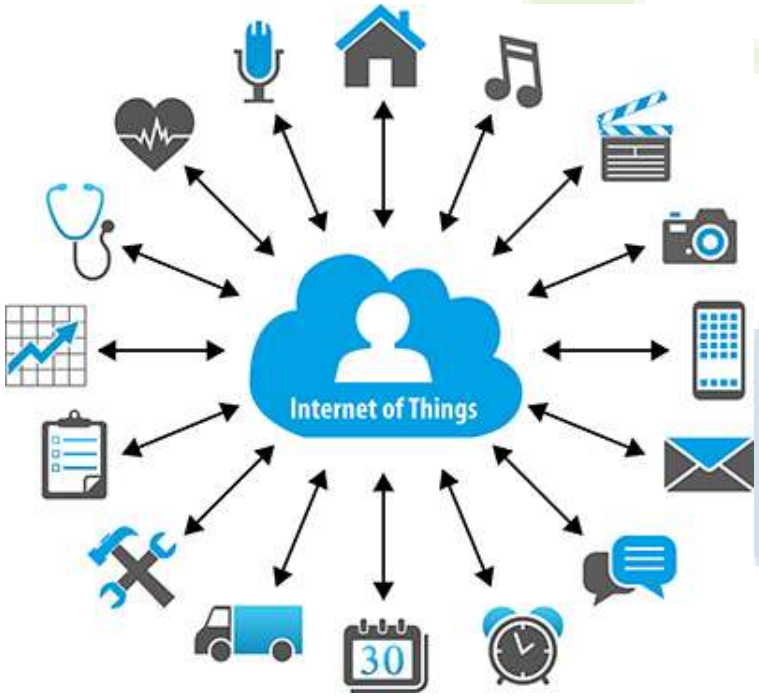
Apache Spark Streaming



Apache Spark Streaming

- Flat Files (à medida que são criados)
- TCP/IP
- Apache Flume
- Apache Kafka
- AWS Kinesis
- Mídias Sociais (Facebook, Twitter, etc...)
- Bancos NoSQL
- Bancos Relacionais

Apache Spark Streaming



Apache Spark Streaming

- Streaming ETL
- Detecção de Anomalias
- Enriquecimento de Dados
- Sessões Complexas e Aprendizado Contínuo

Apache Spark Streaming

- Detecção de Fraudes em Tempo Real
- Filtro de Spam
- Detecção de Invasão de Redes
- Análise de Mídias Sociais em Tempo Real
- Análise de Stream de Cliques em Sites, gerando Sistemas de Recomendação
- Recomendação de Anúncios em Tempo Real
- Análise do Mercado de Ações

Apache Spark Streaming

Coleta e análise dos dados direto da fonte e à medida que são gerados

Transformação, sumarização e análise

Machine Learning

Previsões em tempo real

Apache Spark Streaming

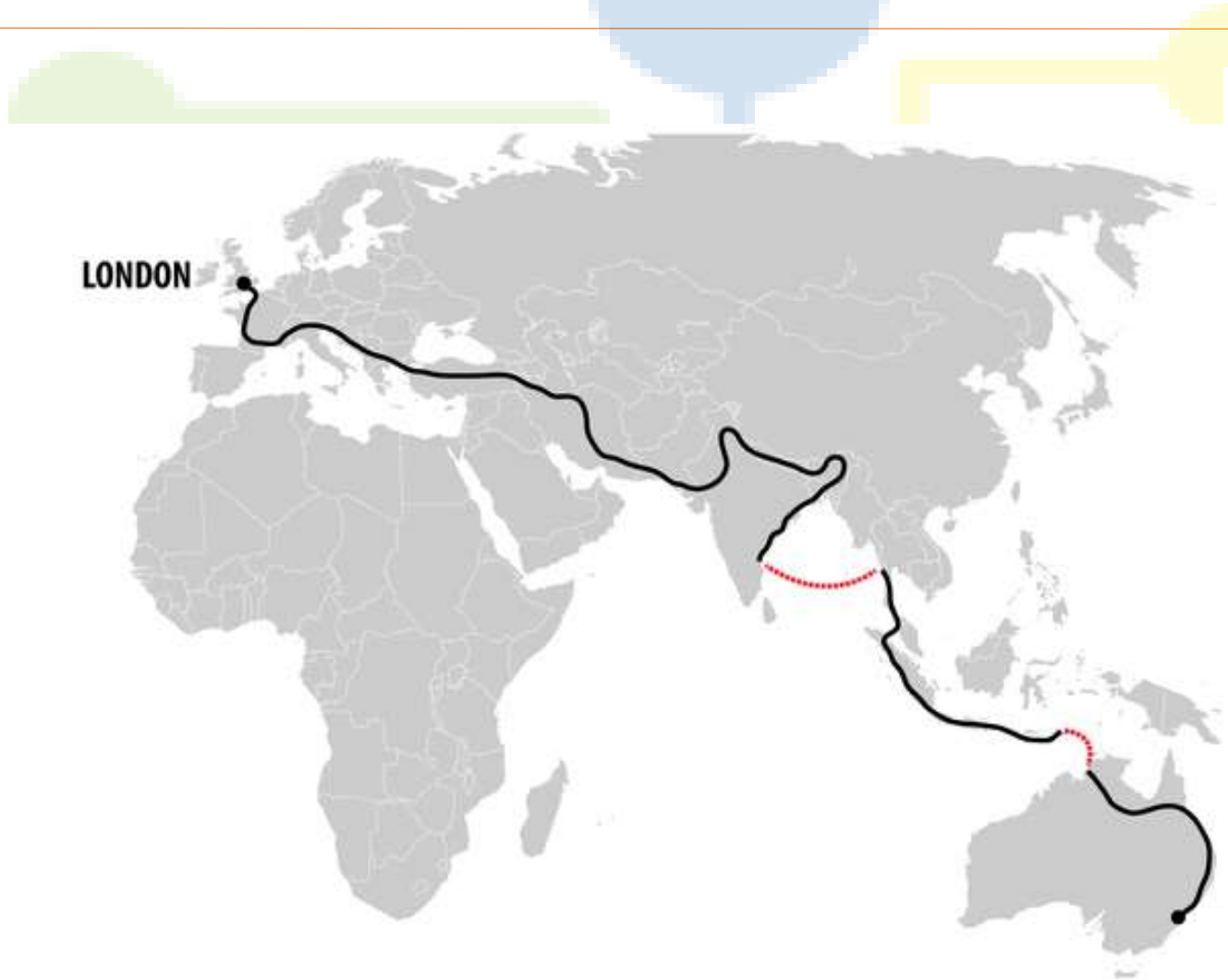
Uma importante vantagem de usar o Spark para Big Data Analytics é a possibilidade de combinar processamento em batch e processamento de streaming em um único sistema.

Big Data Real-Time Analytics com Python e Spark

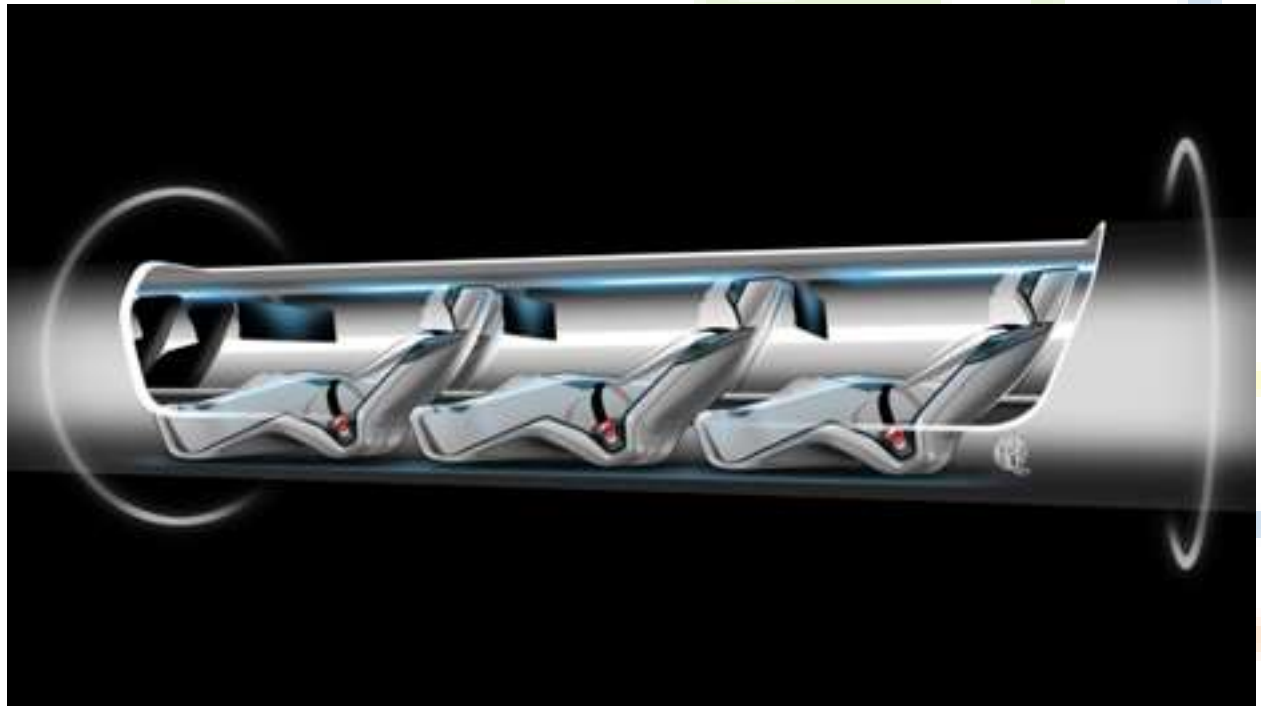
Spark Streaming
A velocidade com que você passa o cartão de crédito

Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito

Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito



Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito

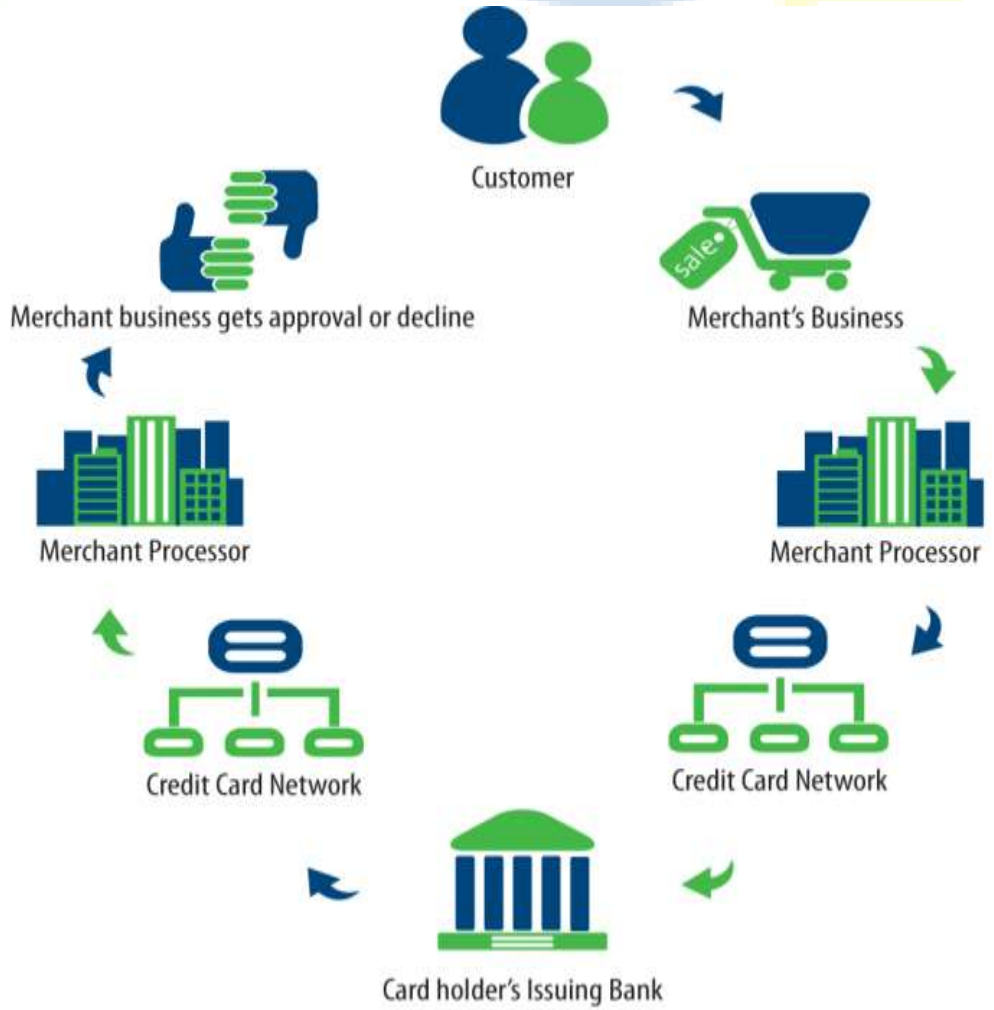


Apenas a título de curiosidade, a Tesla está trabalhando em um projeto de cápsula de transporte supersônica!!

Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito

No combate a fraudes com cartão de crédito, uma propriedade tem sido usada com frequência. A velocidade com que o cartão é passado na máquina leitora é usado como um indicador da probabilidade de atividade fraudulenta. A ideia por trás da velocidade do cartão é bastante simples.

Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito



Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito

Objetivo 1	Quando um cliente utiliza o cartão de crédito para uma transação, o vendedor precisa saber em tempo real a resposta para a frase: É uma fraude?
Objetivo 2	Precisamos manter um histórico de decisões de fraude feitas pelo sistema. Esse histórico deve estar disponível para outros sistemas da empresa.

Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito

Um Streaming de dados é como um dataset contínuo, infinito. Vamos retirando porções deste dataset, analisando, tomando decisões e armazenando para análise futuras. Trabalhar com Streaming de dados é sem dúvida um grande desafio.

Streaming de Dados - A Velocidade com que Você passa o Cartão de Crédito

Big Data Real-Time Analytics com Python e Spark

Arquitetura Apache Spark Streaming

Apache Spark

Processamento **iterativo** – várias tarefas em sequência
Processamento **interativo** – análise exploratória de dados

Arquitetura Apache Spark Streaming

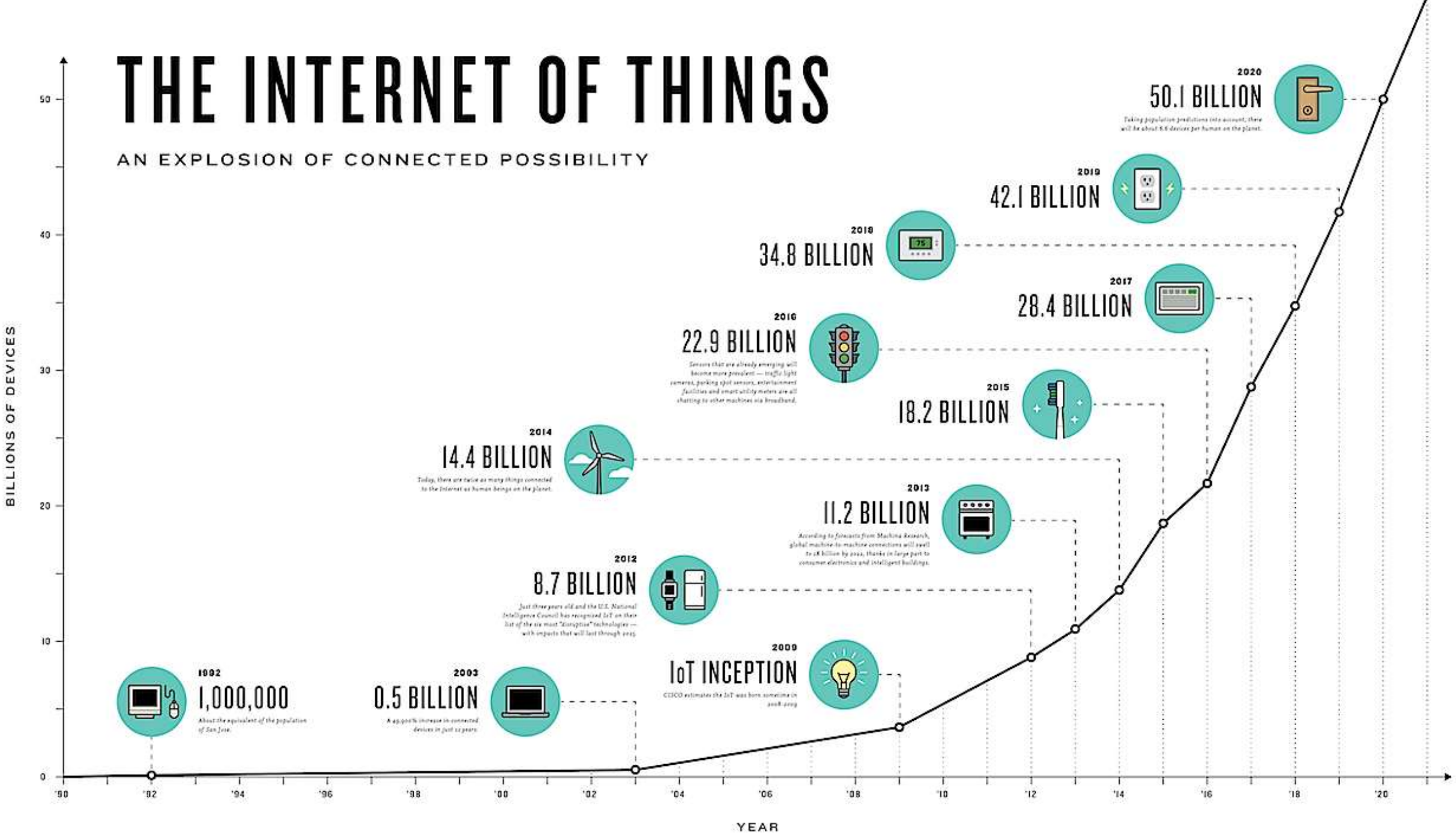
UBER

NETFLIX

amazon

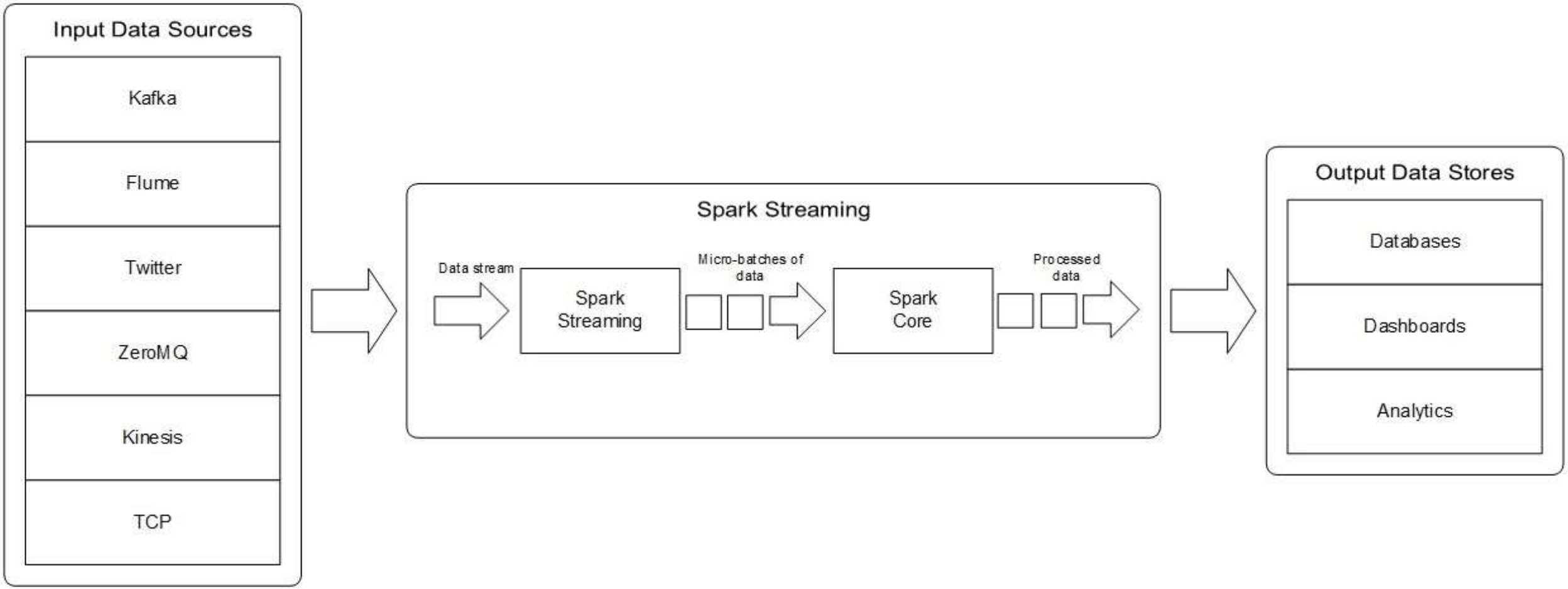
THE INTERNET OF THINGS

AN EXPLOSION OF CONNECTED POSSIBILITY

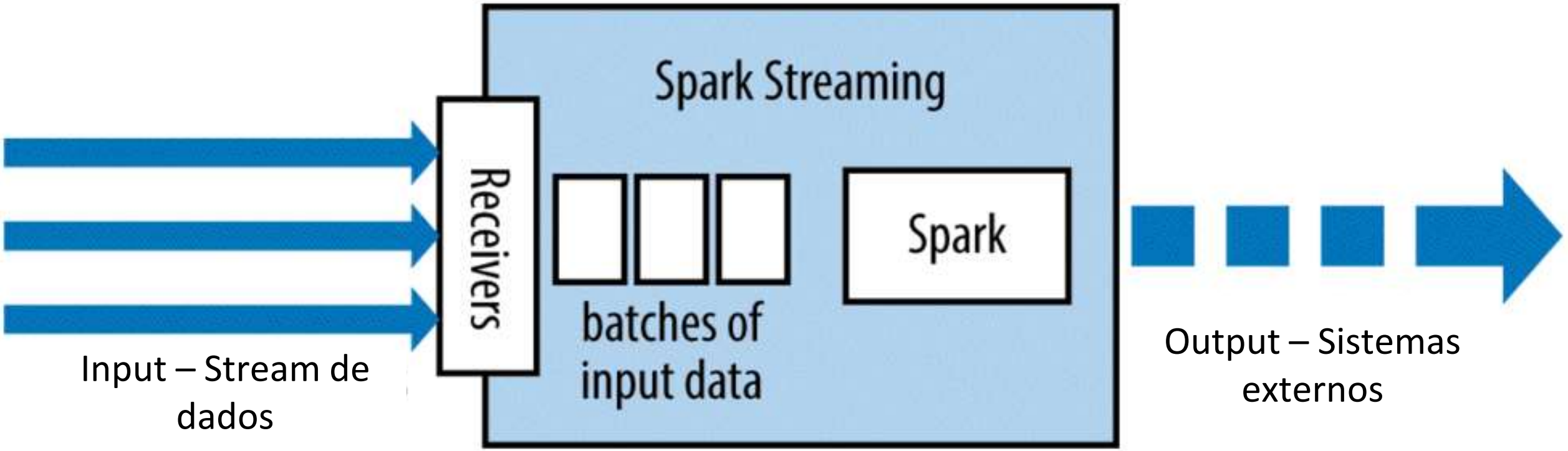


Arquitetura Apache Spark Streaming

Arquitetura Apache Spark Streaming

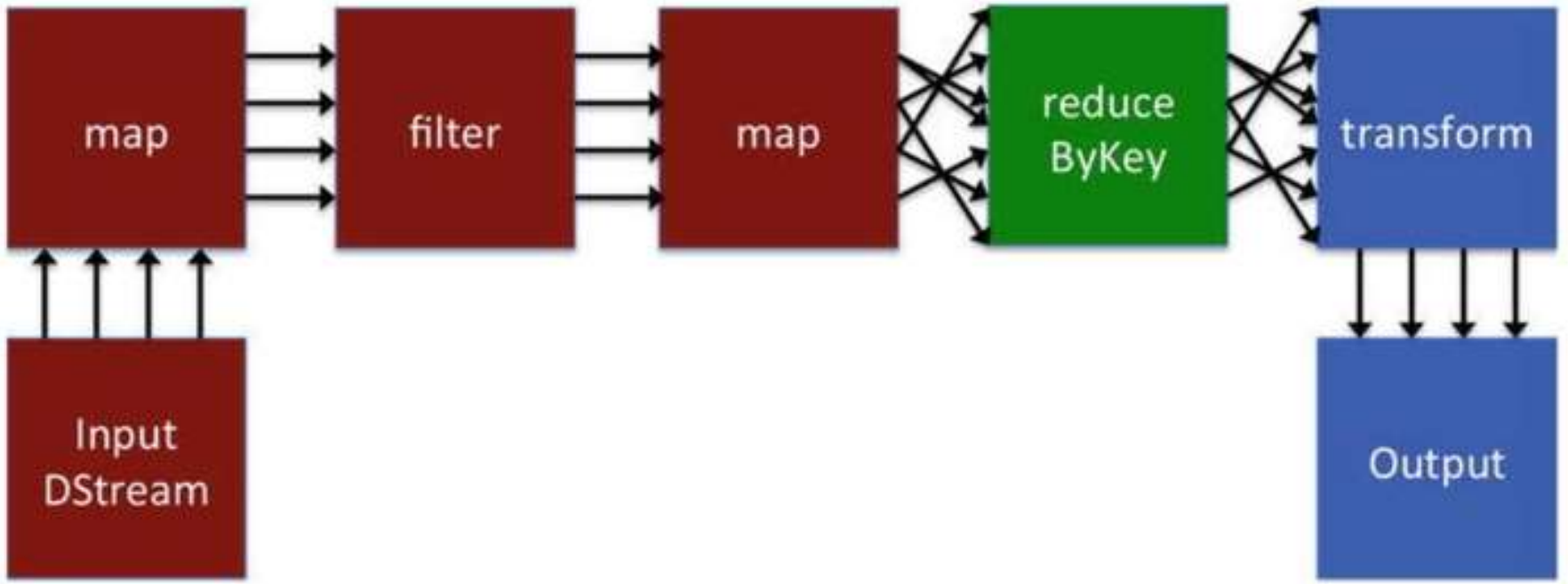


Arquitetura Apache Spark Streaming



Arquitetura Apache Spark Streaming

Arquitetura Apache Spark Streaming



Arquitetura Apache Spark Streaming

Arquitetura Apache Spark Streaming

Spark Streaming ➔

Arquitetura Apache Spark Streaming

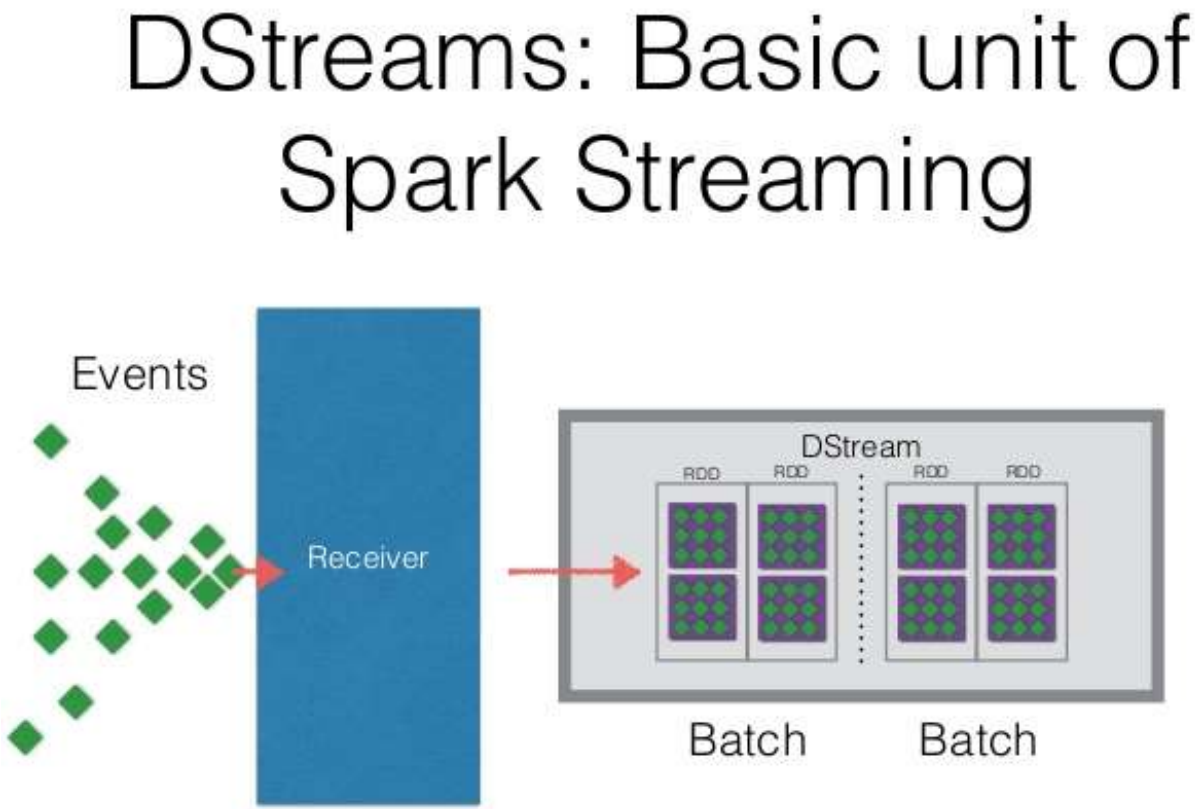
Principais Frameworks para processamento de Streaming de Dados:

- Apache Samza
- Apache Storm
- Apache Flink
- Apache Spark Streaming
- AWS Kinesis (tem custo associado)

Big Data Real-Time Analytics com Python e Spark

O que são DStreams (Discretized Streams)?

O que são DStreams (Discretized Streams)?

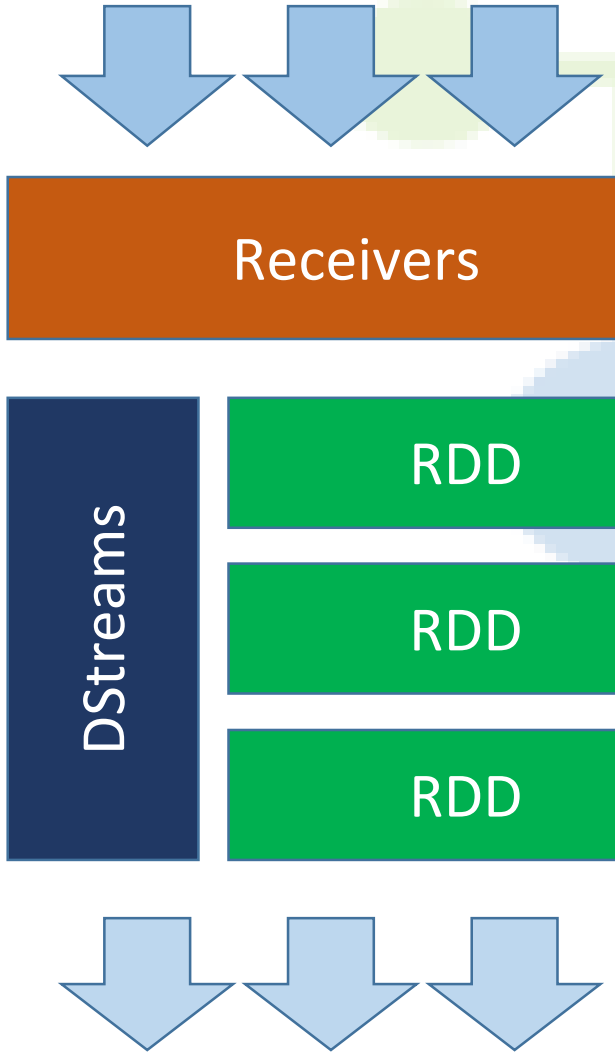


The DStream is (Discretized) into batches, the timing of which is set in the Spark Streaming Context. Each Batch is made up of RDDs.

O que são DStreams (Discretized Streams)?

Os DStreams oferecem muitas das operações que podem ser realizadas com os RDD’s, mais as operações relacionadas ao tempo, como sliding windows.

O que são DStreams (Discretized Streams)?

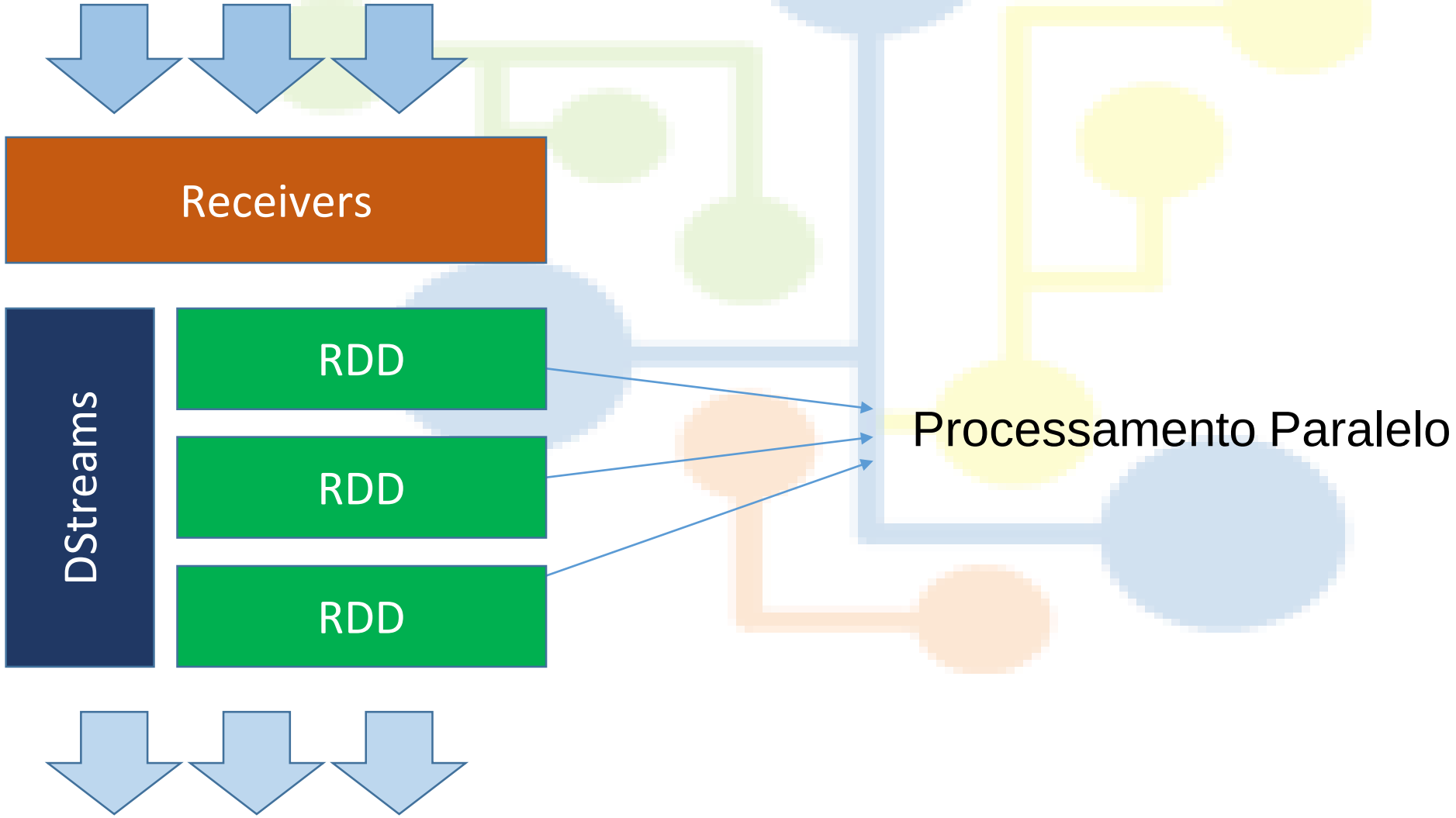


Streams

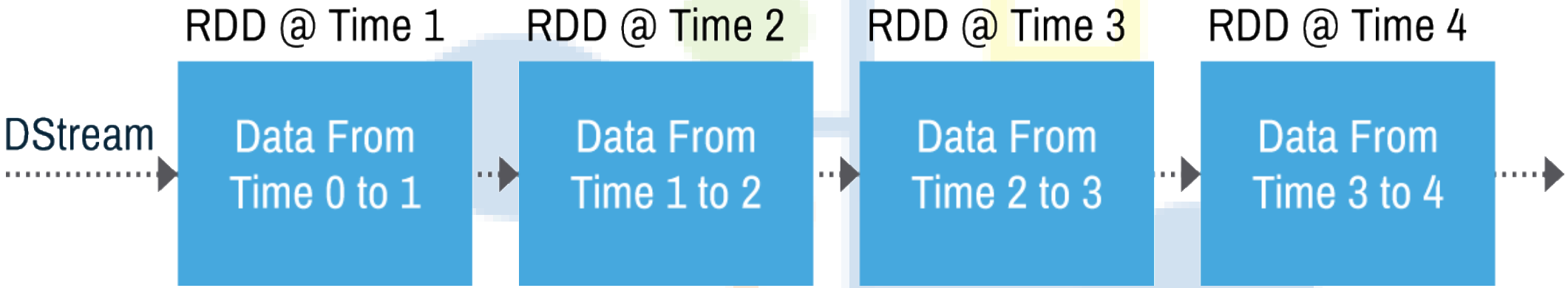
DStreams – batches de dados para um determinado período de tempo

Transformação e output para outros sistemas

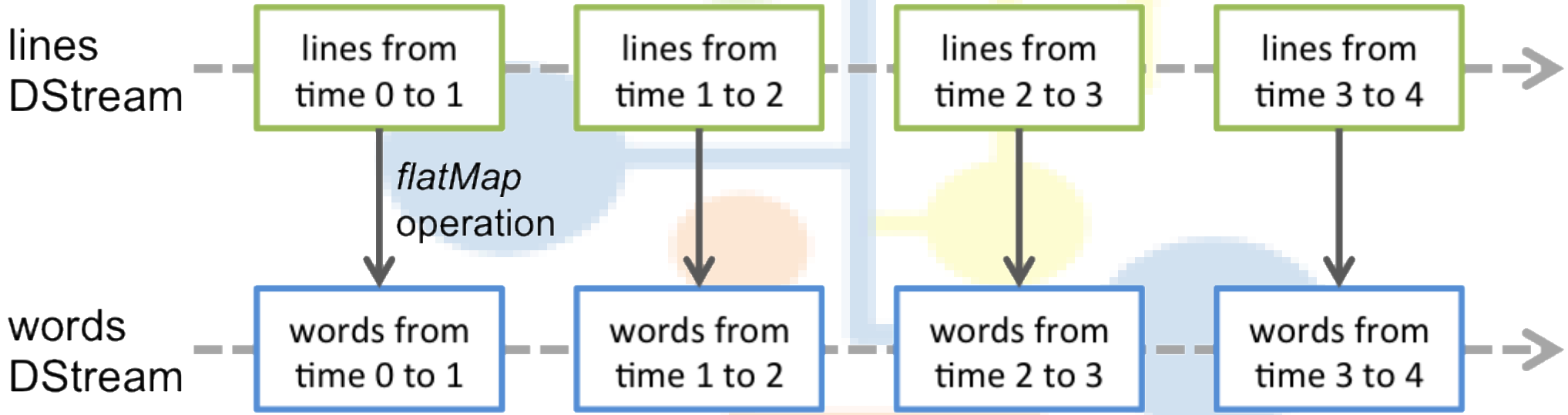
O que são DStreams (Discretized Streams)?



O que são DStreams (Discretized Streams)?



O que são DStreams (Discretized Streams)?



O que são DStreams (Discretized Streams)?

Podemos aplicar aos Dstreams as operações:

- Map
- FlatMap
- Filter
- reduceByKey
- Join
- Window

Mas precisamos tomar alguns cuidados especiais, como manter o controle de estado dos dados (Stateful Data).

Exemplo: coletar streams de cliques em um web site e manter a associação dos dados com a sessão ou ip do usuário.

O que são DStreams (Discretized Streams)?

O DStream permite converter um conjunto de dados contínuo em um conjunto discreto de RDD’s (DStream significa Discretized Streams).

Big Data Real-Time Analytics com Python e Spark

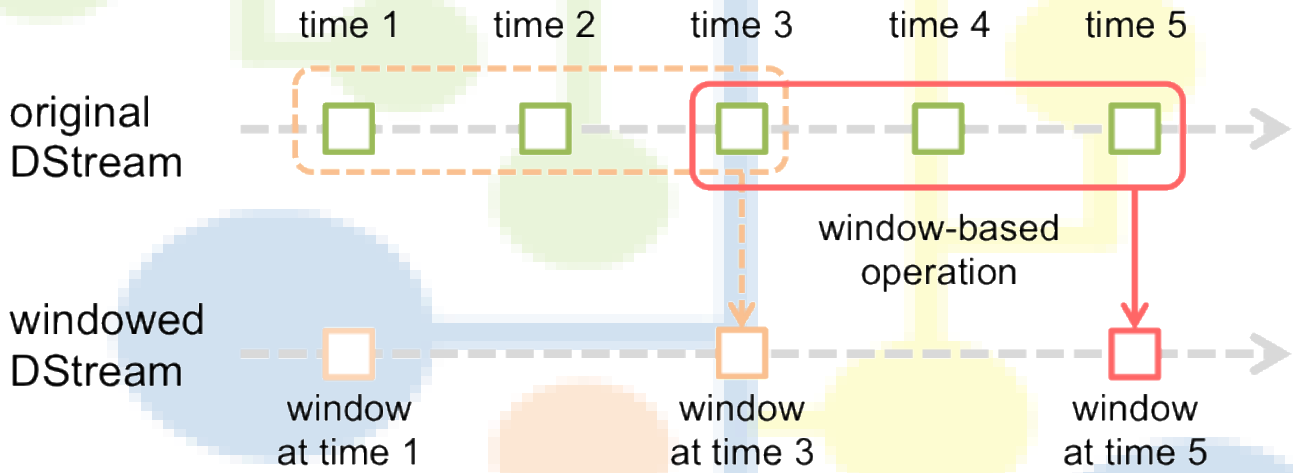
Windowing

Windowing

Computing Windowed

Windowing

Windowing



Window length – duração da window (3 unidades de tempo nesse caso)
Sliding interval – intervalo entre as windows

Windowing

ssc = StreamingContext(sc, **INTERVALO_BATCH**)

window(windowDuration: Duration, slideDuration: Duration): DStream[T]

Windowing

Windowing permite computar os resultados ao longo de períodos de tempo maiores que o batch interval.

Windowing

Batch interval = 1 segundo

Window length = 1 hora

Windowing

Batch Interval

Frequência com que os dados são capturados em um DStream

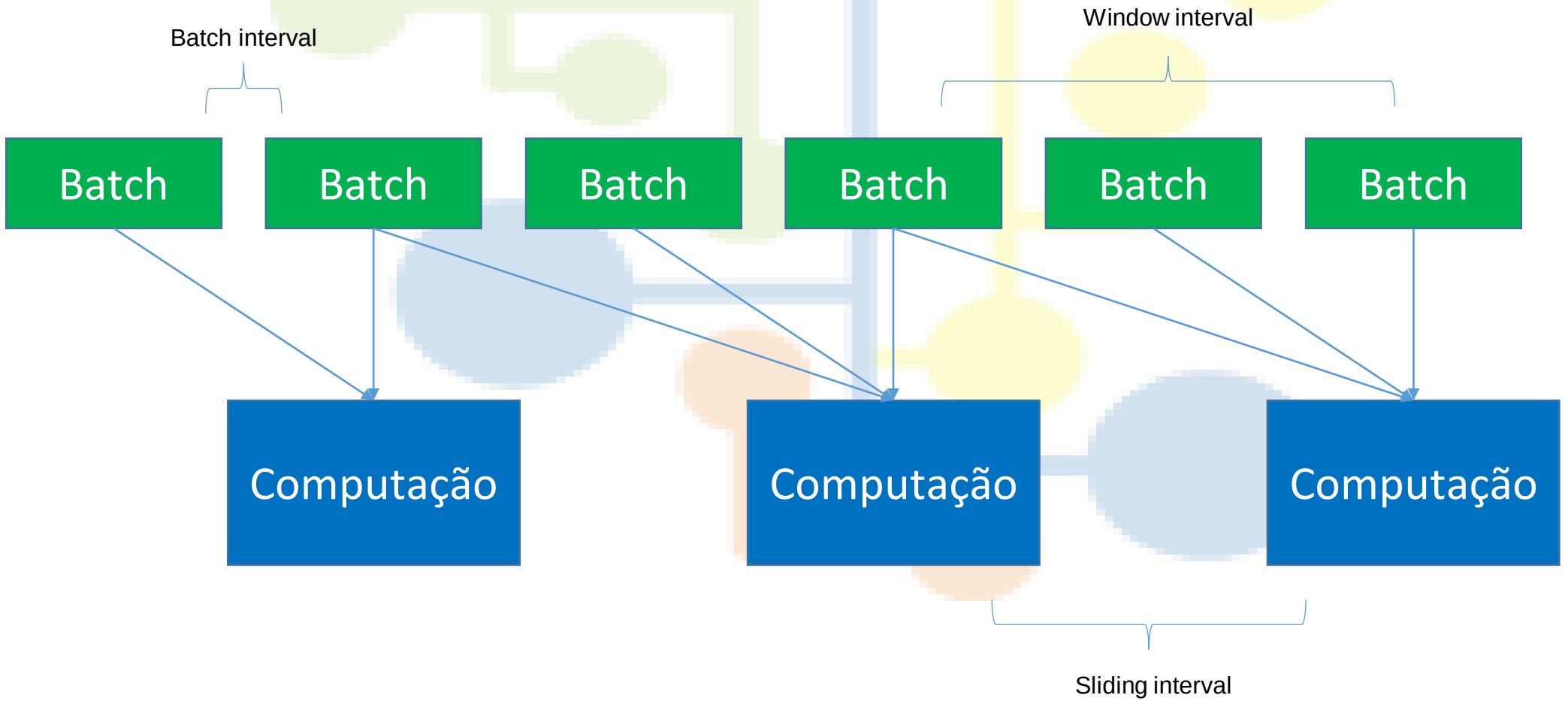
Sliding Interval

Frequência com que uma window é aplicada

Window Interval

Intervalo de tempo capturado para computação e geração de resultados

Windowing


Batch interval
Window interval
Batch
Batch
Batch
Batch
Batch
Batch
Computação
Computação
Computação
Sliding interval

Windowing

Transformation	Meaning
window(*windowLength*, *slideInterval*)	Return a new DStream which is computed based on windowed batches of the source DStream.
countByWindow(*windowLength*, *slideInterval*)	Return a sliding window count of elements in the stream.
reduceByWindow(*func*, *windowLength*, *slideInterval*)	Return a new single-element stream, created by aggregating elements in the stream over a sliding interval using *func*. The function should be associative and commutative so that it can be computed correctly in parallel.
reduceByKeyAndWindow(*func*, *windowLength*, *slideInterval*, [*numTasks*])	When called on a DStream of (K, V) pairs, returns a new DStream of (K, V) pairs where the values for each key are aggregated using the given reduce function *func* over batches in a sliding window. **Note:** By default, this uses Spark's default number of parallel tasks (2 for local mode, and in cluster mode the number is determined by the config property `spark.default.parallelism`) to do the grouping. You can pass an optional `numTasks` argument to set a different number of tasks.
reduceByKeyAndWindow(*func*, *invFunc*, *windowLength*, *slideInterval*, [*numTasks*])	A more efficient version of the above `reduceByKeyAndWindow()` where the reduce value of each window is calculated incrementally using the reduce values of the previous window. This is done by reducing the new data that enters the sliding window, and "inverse reducing" the old data that leaves the window. An example would be that of "adding" and "subtracting" counts of keys as the window slides. However, it is applicable only to "invertible reduce functions", that is, those reduce functions which have a corresponding "inverse reduce" function (taken as parameter *invFunc*). Like in `reduceByKeyAndWindow`, the number of reduce tasks is configurable through an optional argument. Note that checkpointing must be enabled for using this operation.
countByValueAndWindow(*windowLength*, *slideInterval*, [*numTasks*])	When called on a DStream of (K, V) pairs, returns a new DStream of (K, Long) pairs where the value of each key is its frequency within a sliding window. Like in `reduceByKeyAndWindow`, the number of reduce tasks is configurable through an optional argument.

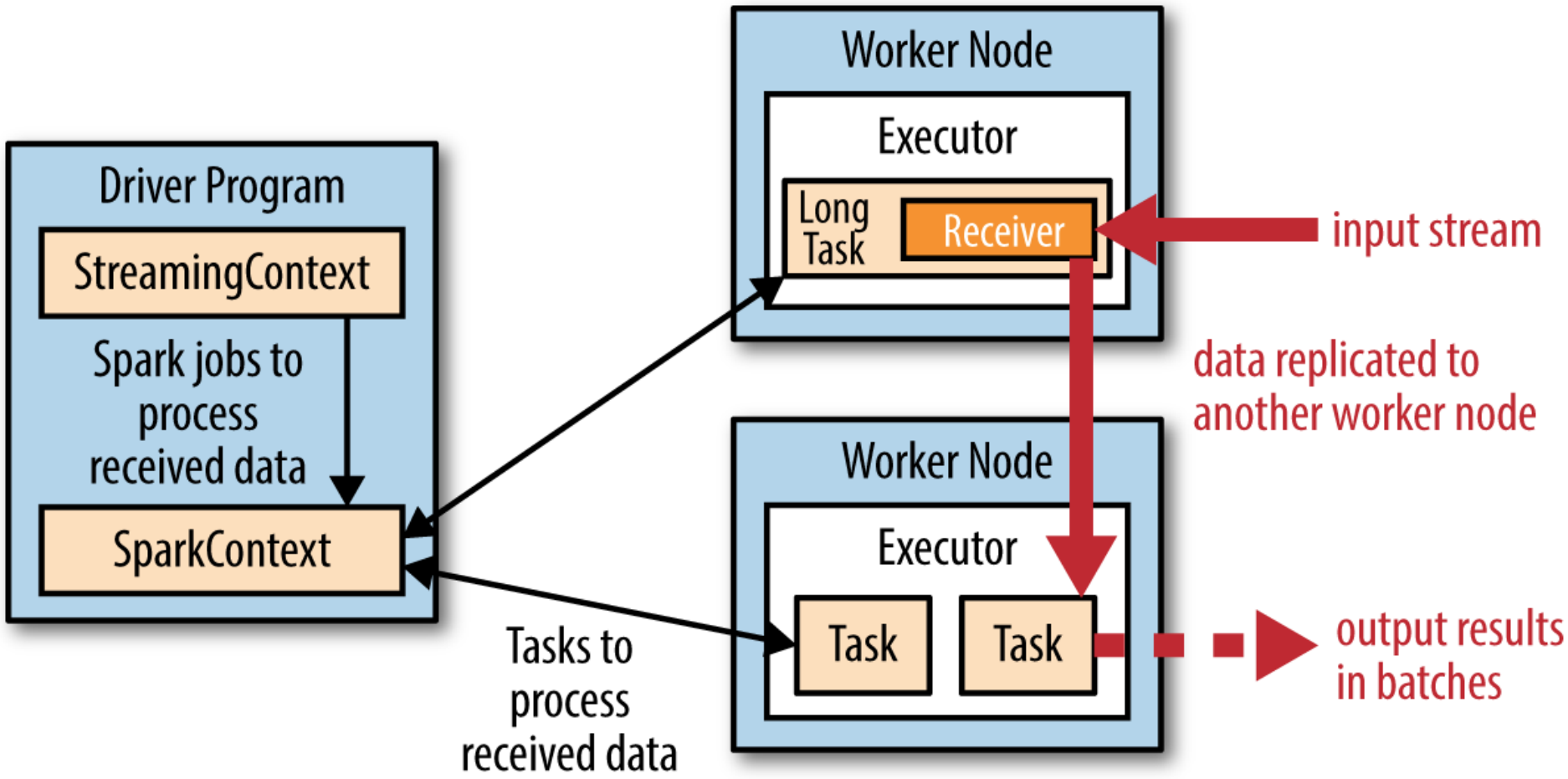
E existem diversas funções de transformação específicas para se trabalhar com Window.

Big Data Real-Time Analytics com Python e Spark

Tolerância a Falhas

Tolerância a Falhas

Tolerância a Falhas



Tolerância a Falhas

Todos os dados são replicados para no mínimo 2 worker nodes.

Tolerância a Falhas

Um diretório de checkpoint pode ser usado para armazenar o estado do streaming de dados, no caso em que é necessário reiniciar o streaming.

ssc.checkpoint()

Tolerância a Falhas

Falha no Receiver

Falha no Driver Context (script)

Tolerância a Falhas

Falha no Receiver

- Alguns receivers são melhores que outros.
- Receivers como Twitter, Kafka e Flume são permitem recuperação de dados. Se o receiver falha, os dados do streaming são perdidos.
- Outros receivers garantem a recuperação dos dados em caso de falhas: HDFS, Directely-consumed Kafka, Pull-based Flume.

Tolerância a Falhas

Falha no Driver Context

- Embora os dados sejam replicados para os worker nodes, o DriverContext é executado no node Master e este pode ser um ponto único de falha.

- Podemos usar checkpoint() para recuperar dados em caso de falhas e usamos a função StreamingContext.getOrCreate() para continuar o processamento de onde ele foi interrompido em caso de falha.

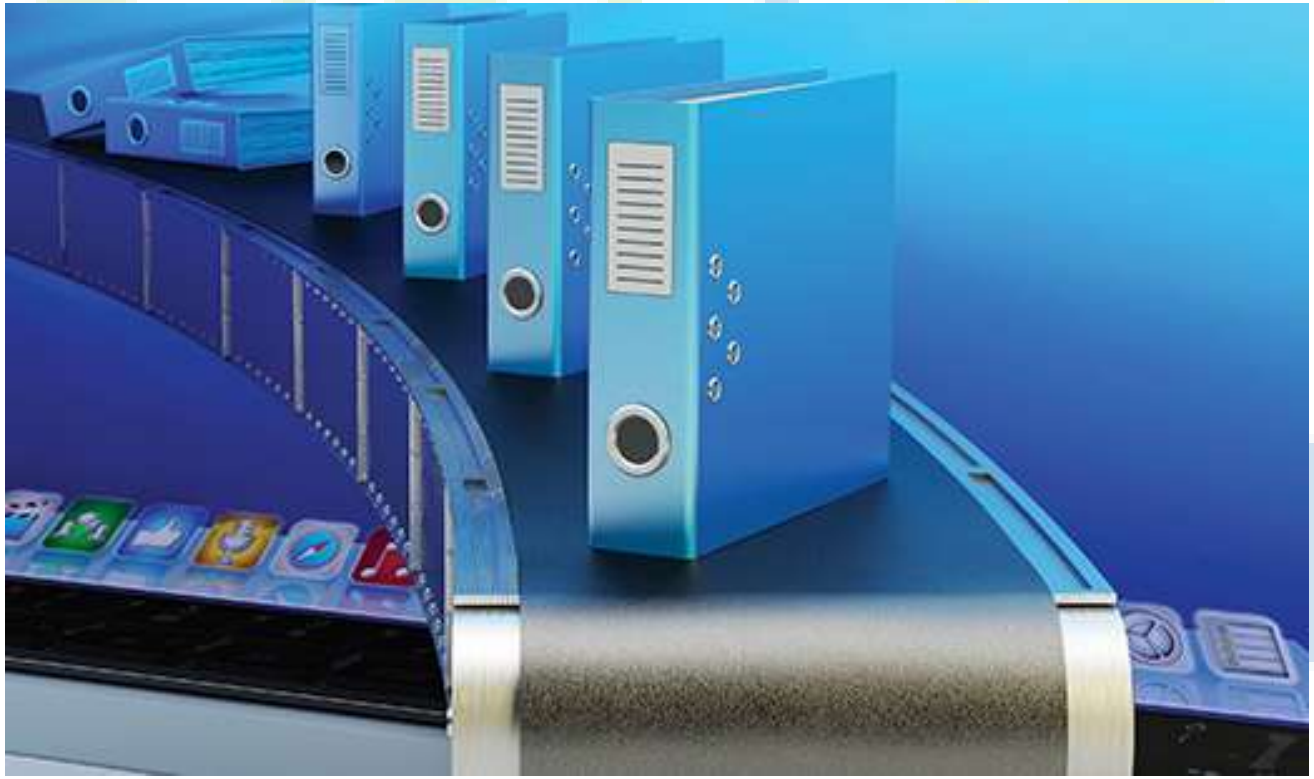
- Em caso de falha no seu script sendo executado no DriverContext, podemos reiniciar automaticamente o processo de streaming, usando o Zookeeper (no modo supervise). O Zookeeper é um cluster manager usado pelo Spark.

Tolerância a Falhas

Big Data Real-Time Analytics com Python e Spark

Integração com Outros Sistemas
Kafka, Flume, Kinesis

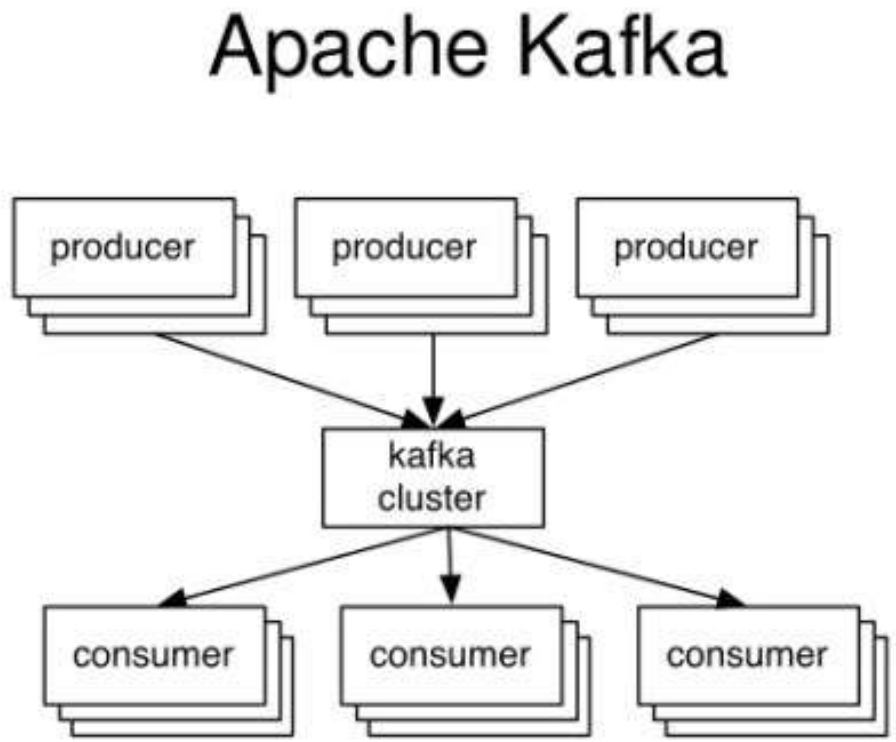
Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis




BIG
DATA
& ANALYTICS

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

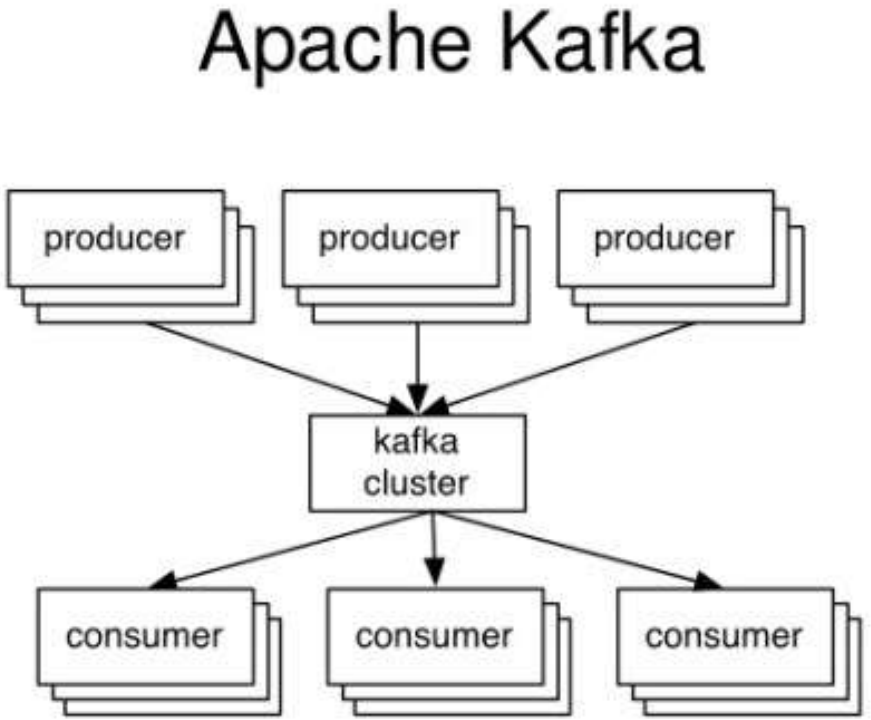
Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis



Servidores web gerando logs

Spark Streaming Receiver

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis



Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

Transportar dados entre diversos sistemas de dados

Enriquecer a análise de dados

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

Soluções Similares ao Kafka:

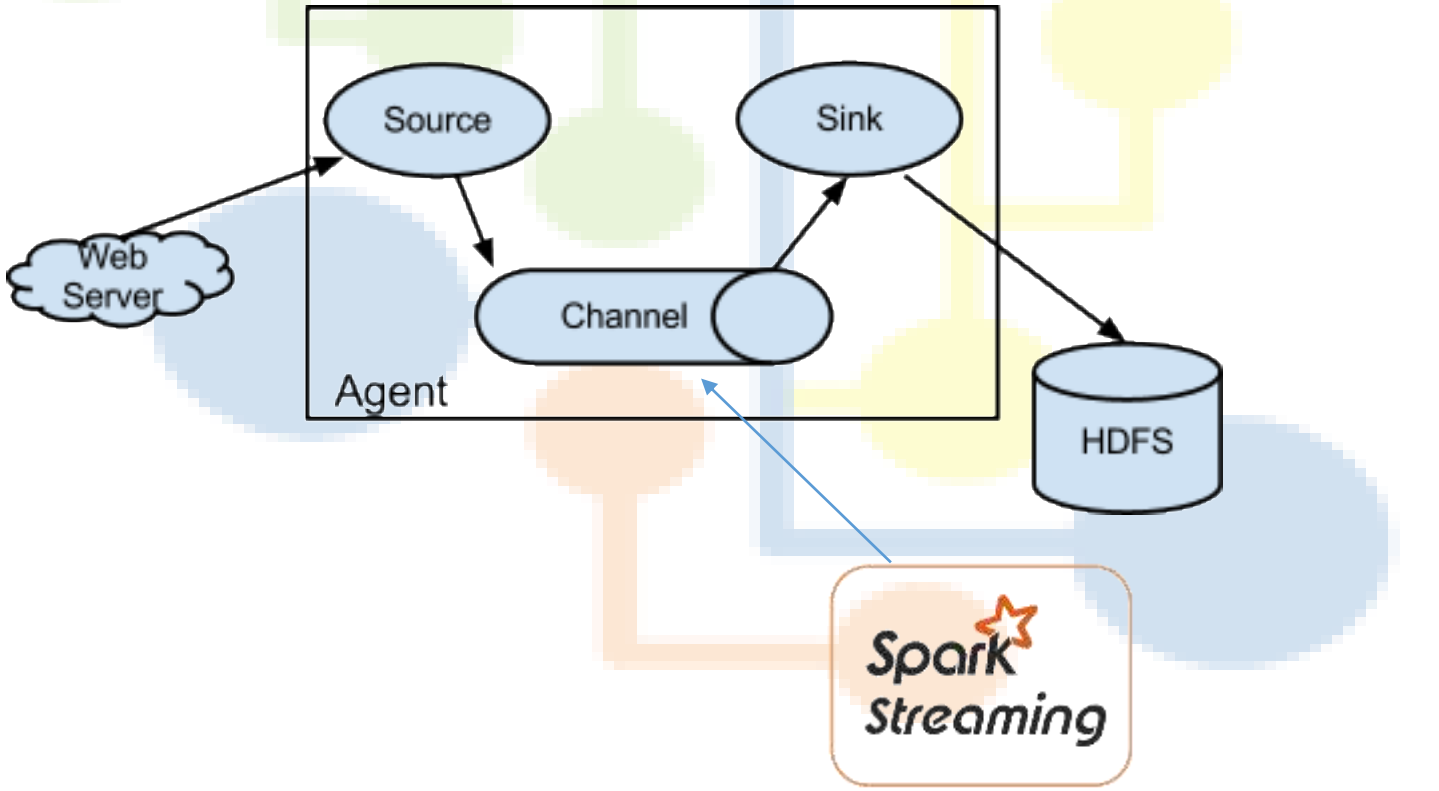
- IBM InfoSphere Streams
- Informatica’s Ultra Messaging Streaming Edition
- SAS’s Event Stream Processing Engine (ESP)
- Tibco’s StreamBase
- DataTorrent
- Splunk
- Loggly
- Logentries
- Glassbeam

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

É necessário instalar o pacote spark-streaming-kafka!

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis



Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

Push-Based Flume X Pull-Based Flume

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

É necessário instalar o pacote spark-streaming-flume – Push-Based

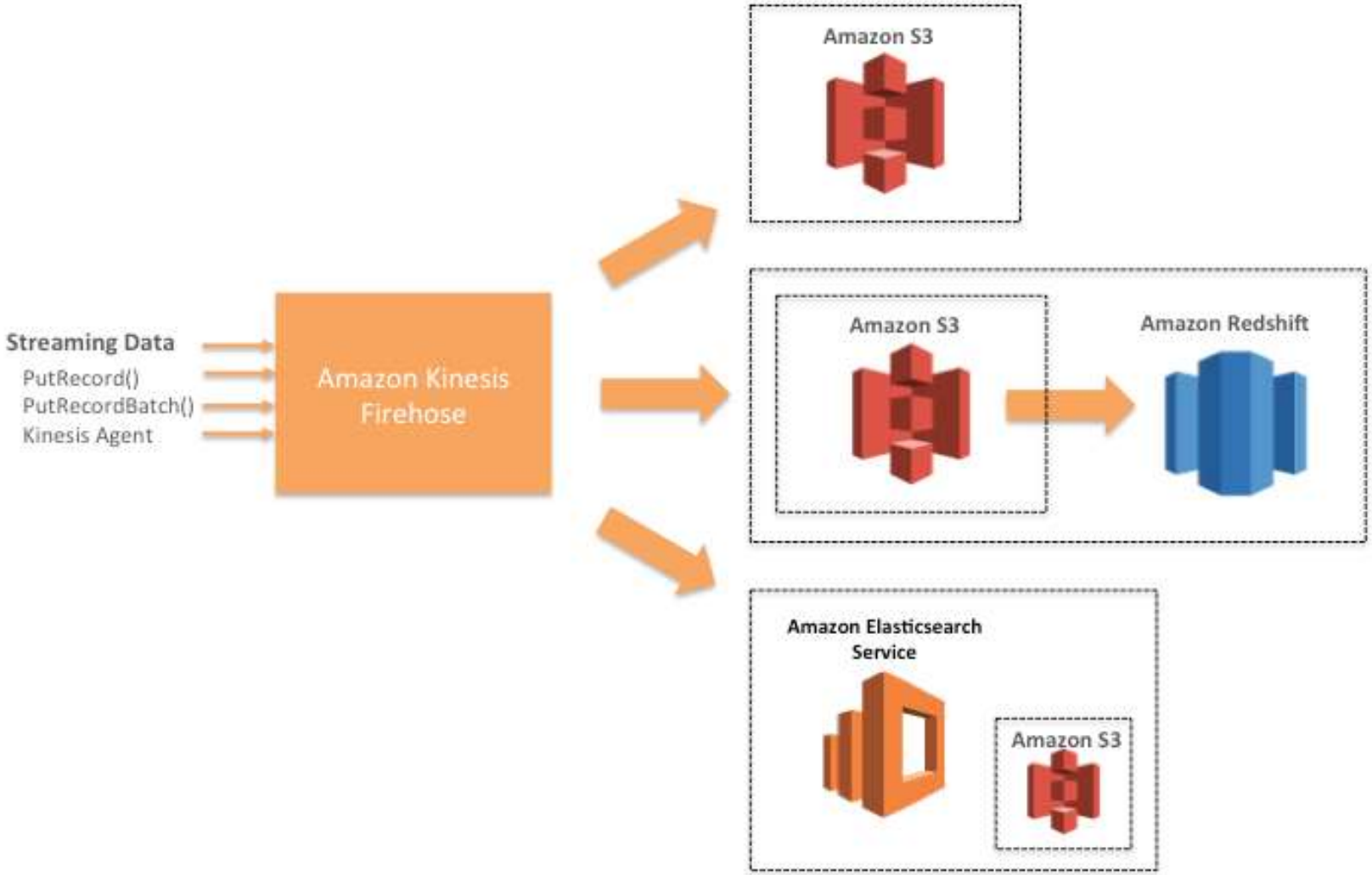
É necessário instalar o pacote spark-streaming-flume-sink – Pull-Based

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

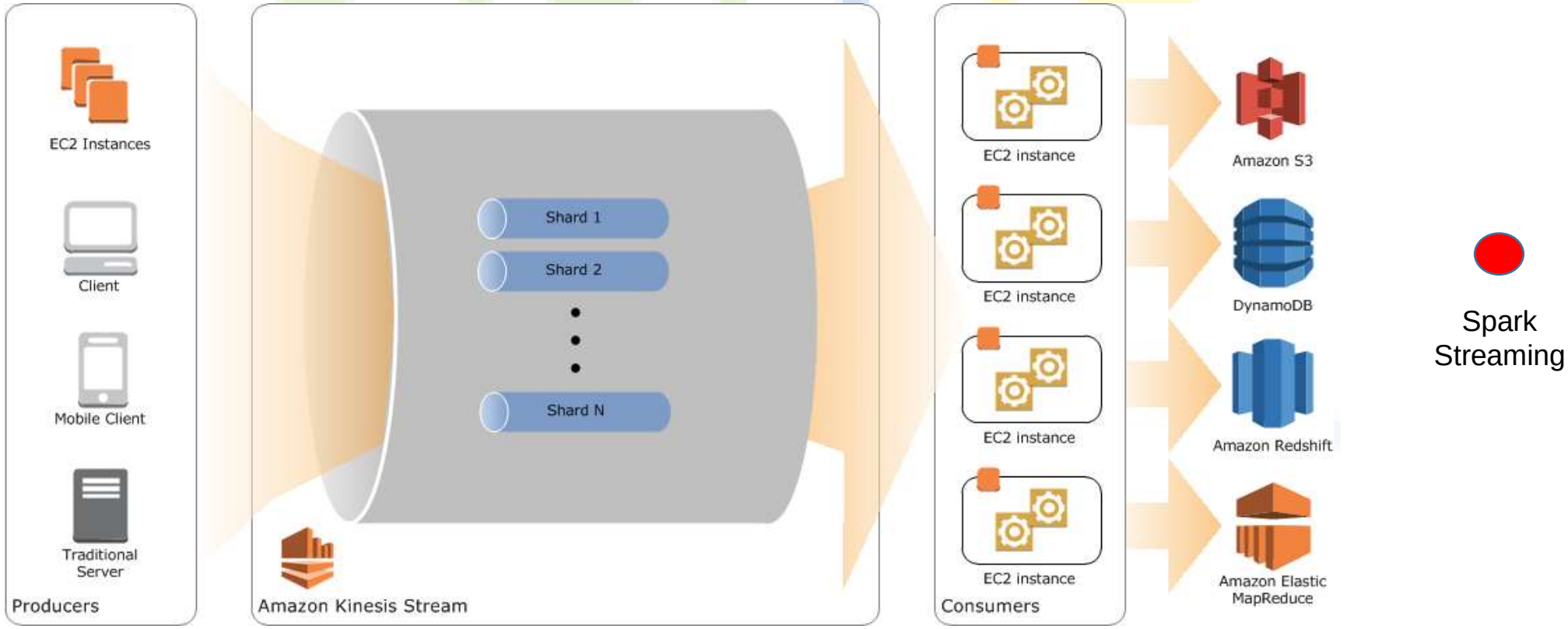
Ou

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis



Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis



Spark Streaming

Integração com Outros Sistemas - Kafka, Flume, Amazon Kinesis

É necessário instalar o pacote spark-streaming-kinesis-asl

Requer Licença da Amazon!

Big Data Real-Time Analytics com Python e Spark

Introdução ao Processamento de Linguagem Natural

Introdução ao Processamento de Linguagem Natural

Introdução ao Processamento de Linguagem Natural

Processamento de Linguagem Natural

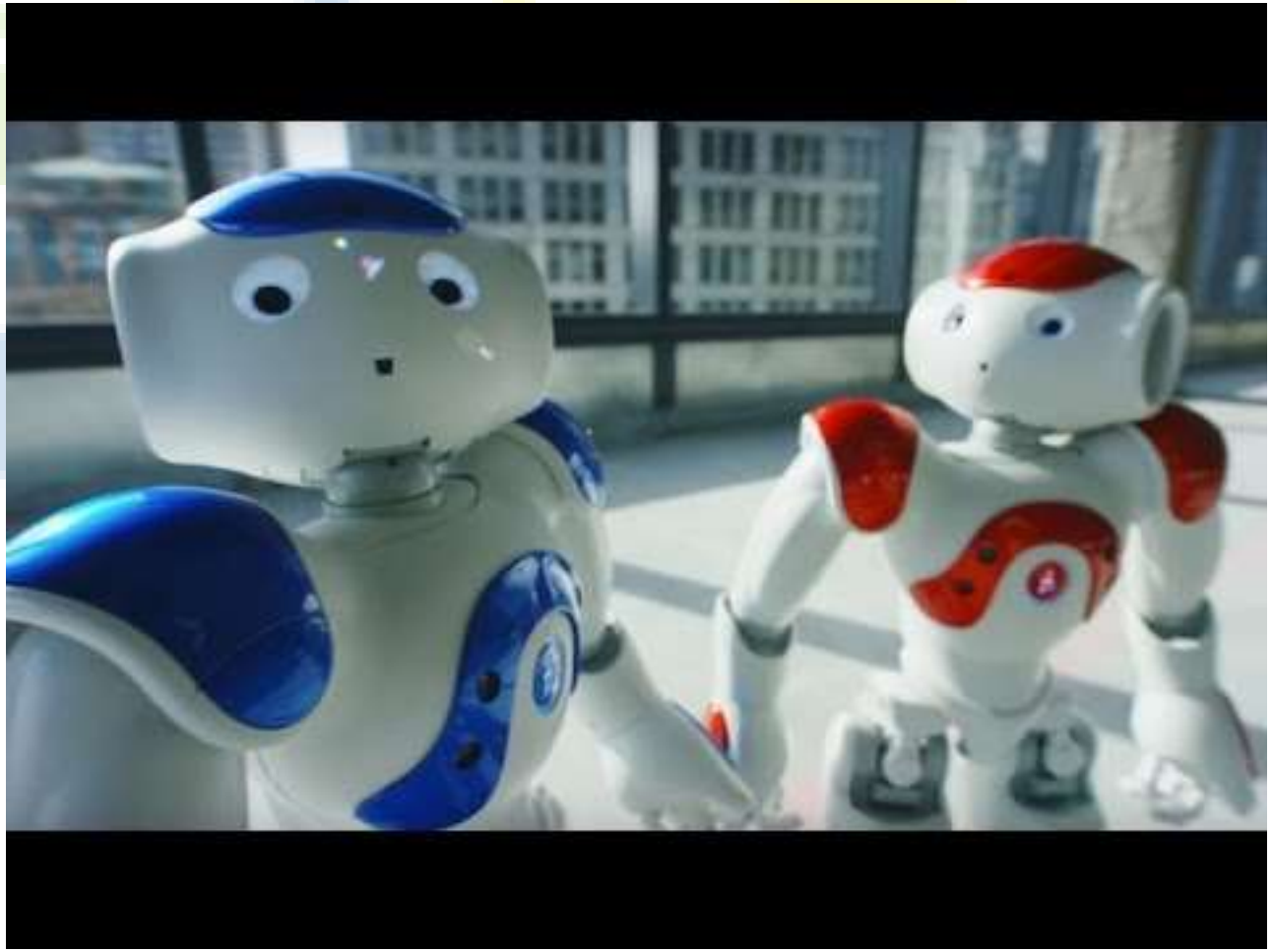
Introdução ao Processamento de Linguagem Natural

O computador espera que a linguagem humana seja precisa, não ambígua e altamente estruturada.

Introdução ao Processamento de Linguagem Natural

PLN

Processamento de
Linguagem Natural



Introdução ao Processamento de Linguagem Natural

PLN

Processamento de
Linguagem Natural

Introdução ao Processamento de Linguagem Natural

Talvez você não saiba, mas você utiliza PLN em aplicações como:

- Corretores Ortográficos (Microsoft Word)
- Engines de Reconhecimento de Voz (Siri, Google Assistente, Cortana)
- Classificadores de Spam
- Mecanismos de Busca (Google, Bing)
- IBM Watson

Introdução ao Processamento de Linguagem Natural

Introdução ao Processamento de Linguagem Natural

Análise de Sentimentos

Introdução ao Processamento de Linguagem Natural

Principais Frameworks para PLN:

- GATE (General Architecture for Text Engineering)
- Mallet (Machine Learning for Language Toolkit)
- OpenNLP
- UIMA
- Gensim
- SpaCy
- Natural Language Toolkit (NLTK)

Introdução ao Processamento de Linguagem Natural

Natural Language
Tool Kit (NLTK)

Tenha uma Excelente Jornada de Aprendizagem.

Muito Obrigado por Participar!

Equipe Data Science Academy