

NÁSKOK  
DÍKY  
ZNALOSTEM

PROFINIT

# B0M33BDT

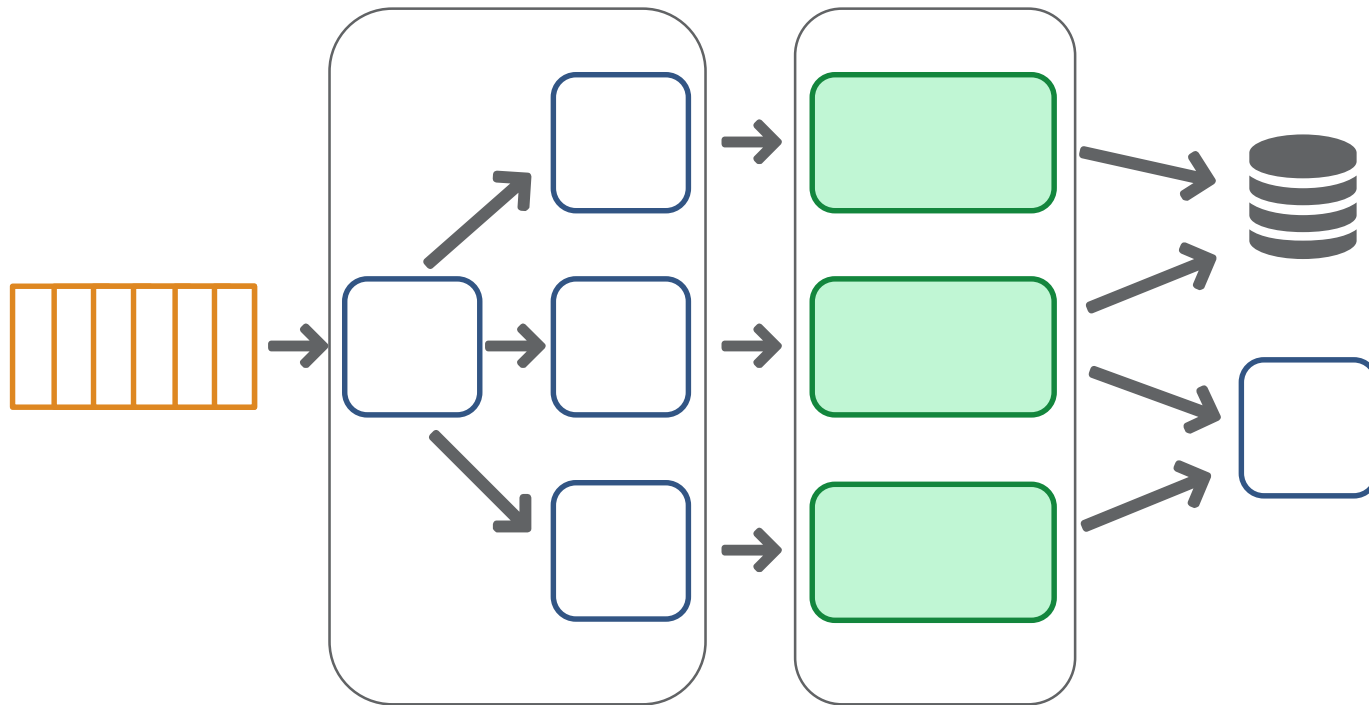
## Stream processing

Milan Kratochvíl

12. prosinec 2018

# Stream processing

- › Průběžné zpracování trvalého toku zpráv



Vstupní stream

Message processor

Stream processor

Úložiště

# Stream processing

- › Hlavní komponenty
  - Message processor – vstup dat do streamového zpracování
  - Stream processor – jádro zpracování údajů
  - Úložiště výstupů – persistentní úložiště zpracovaných dat
  
- › Životní cyklus zpráv
  1. Příjem zpráv
  2. Rozdělení zpráv do partitions
  - 3. Zpracování zpráv (logika aplikace)**
  4. Uložení, notifikace výsledků atd.

# Stream processing vs batch processing

## Batch processing

- › Zpracování velkého množství dat najednou
- › **Pořadí nutno vyčíst z dat**
- › Zpracování s velkým zpožděním (denní, hodinové...)
- › Výsledky z principu nelze poskytovat „online“
- › Efektivní využití zdrojů (paměť, CPU)
- › Zpracuje enormní množství libovolných dat (petabajty)
- › Hadoop založen na batch zpracování

## Stream processing

- › Zpracování záznam za záznamem
- › **Zaručené pořadí**
- › Zpracování ihned po příchodu zprávy nebo s malým zpožděním
- › Výsledky jsou často dostupné „online“
- › Náročnější na zdroje (**paměť**, CPU)
- › Zpracování složitých dotazů na velkém množství dat nemusí být efektivní
- › Pro Hadoop relativně nové



Messaging processor

# Apache Kafka

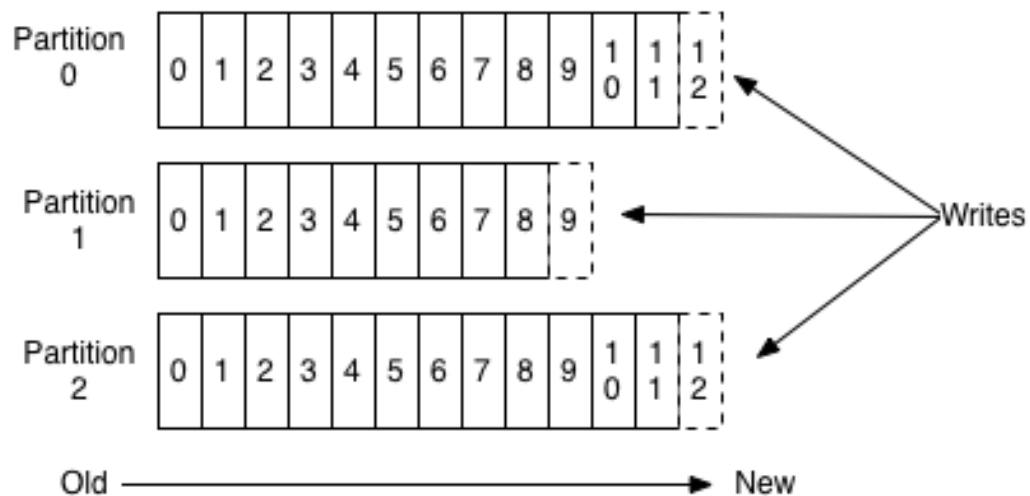
- › Distribuovaný systém pro zpracování datových streamů
- › Typicky se používá jako messaging systém
- › Transakční log
  
- › Základní vlastnosti
  - vysoce výkonný (zpracuje i miliony zpráv za sekundu)
  - distribuovaný
  - zajišťuje replikaci dat
    - nevyžívá HDFS, má vlastní způsob ukládání dat
  - schopnost řešit výpadky v clusteru
  - škálovatelný
    - lze snadno přidat nový node pro zvýšení propustnosti



# Základní pojmy

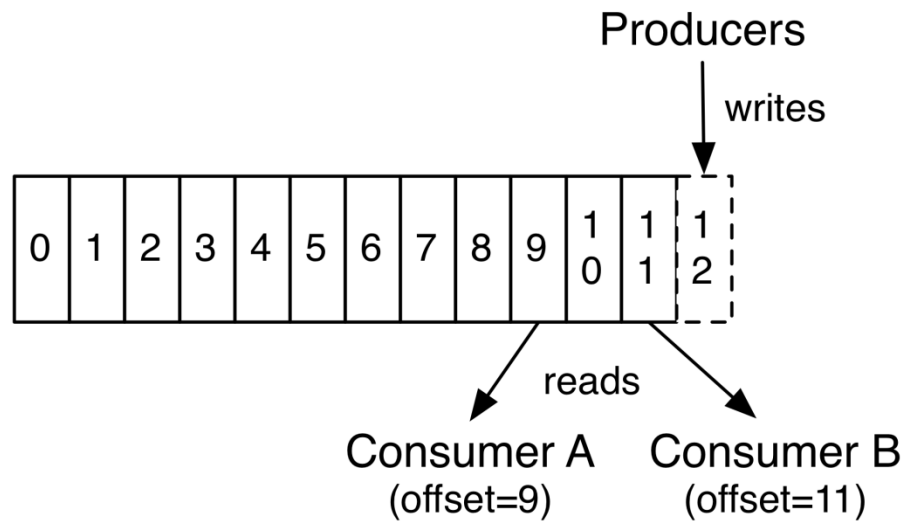
- › Topic
  - pojmenovaná „fronta zpráv“
- › Partition
  - dělení topiku na menší části
- › Offset
  - aktuální pozice v topiku/partition (orientace, kde se v topiku nacházíme)
- › Consumer
  - odběratel dat
- › Consumer Group
  - skupina odběratelů dat
- › Producer
  - tvůrce dat

## Anatomy of a Topic

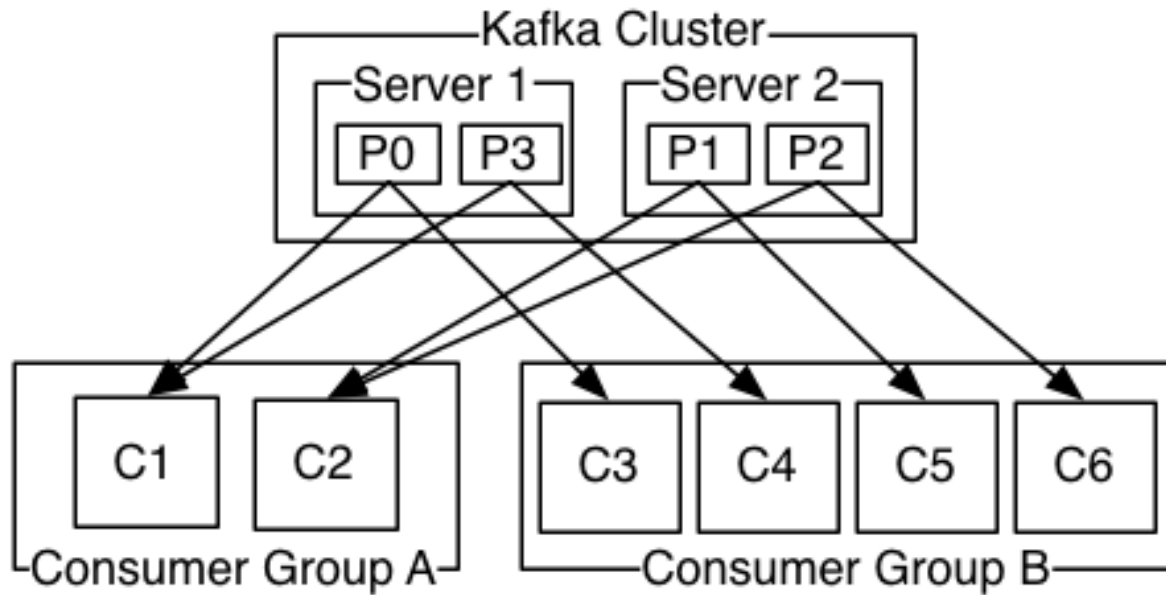




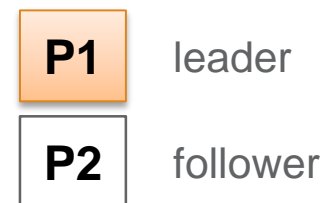
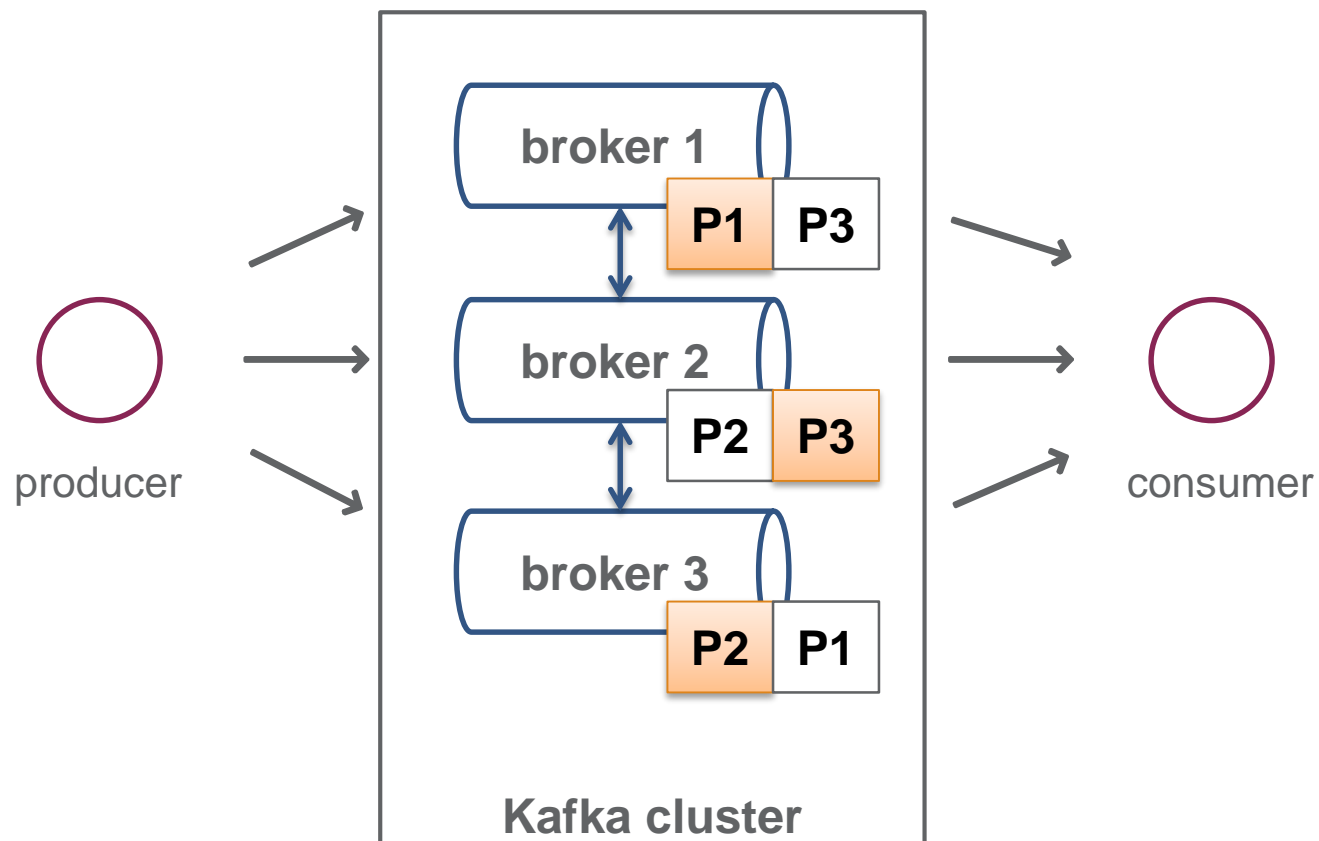
# Základní pojmy



# Základní pojmy



# Zápis a čtení



**Mathias Verraes**

@mathiasverraes

 Follow

There are only two hard problems in distributed systems: 2. Exactly-once delivery 1. Guaranteed order of messages 2. Exactly-once delivery

RETWEETS

6,775

LIKES

4,727



10:40 AM - 14 Aug 2015



69



6.8K



4.7K



# Pořadí zpráv v Kafce

- › Pořadí zpráv
  - garance doručení zpráv v pořadí, v jakém byly zapsány **per partition**
  - ale ne v rámci všech zpráv daného topiku!
- › Standardně se zprávy rozdělují rovnoměrně mezi jednotlivé partitions (náhodně)
- › Typicky ale potřebuji zajistit pořadí zpráv jen např. pro klienta, zařízení... → možnost definovat vlastní pravidla partitioningu

## Další možnosti Kafky

- › Komprese zpráv
- › Automatická retence zpráv
  - staré zprávy automaticky maže po uplynutí definovaného období
- › Obsahuje vlastní systém pro zpracování streamů – Kafka Streams
- › KSQL
  - SQL-like jazyk pro přístup k datům Kafky
- › Základní transakční zpracování
  - consumer commituje poslední zpracovaný offset

# Použití Kafky

- › Všude tam, kde se komunikuje prostřednictvím zpráv, tj. skoro všude
  - senzorická data
  - finanční transakce
  - burzovní informace
  - logy
- › Kappa architektura
- › Lambda architektura
- › Kafka nabízí velkou propustnost a robustnost, ale někdy za cenu vyšších latencí
  - obecně je třeba počítat s desítkami ms latencí jako minimum
  - lze optimalizovat na úkor propustnosti a bezpečnosti (konzistence) dat

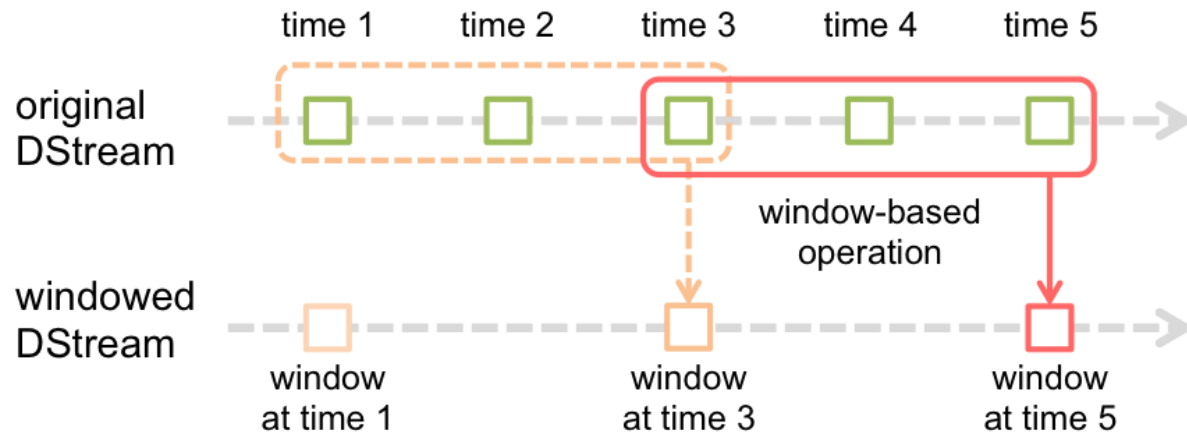
# Stream processor



# Charakteristiky streamového zpracování

- › Stream je analogie („nekonečné“) tabulky
- › Streamy lze partitionovat
  - paralelizace
- › Streamy je možné
  - číst
  - zapisovat
  - joinovat
- › Často je potřeba udržovat stav (typicky aggregate)
  - např. suma obrátů na účtu, průměrná hodnota konkrétního senzoru...
- › Práce s časovými okny
  - vyhodnocování úseků dat

# Časová okna



- › Délka okna
- › Interval posunu
  - jak často se okno posouvá
- › Mohou se překrývat!
- › Příklad
  - délka okna 3s
  - interval posunu 2s

# Časová okna

- › Většina transformací probíhá právě s oknem (typicky nás nezajímají libovolně stará data)
  
- › Příklady
  - akciové/bitcoinové trhy – výpočty klouzavého průměru, např.
    - délka okna 5min
    - posun intervalu 1min
  - detekce událostí – zjišťuji, zda v intervalu nenastaly události společně, např. zjišťujeme, zda se průměr a medián liší v daném okně o více než 10%
    - délka okna 5min
    - posun intervalu 5min

# Druhy streamového zpracování

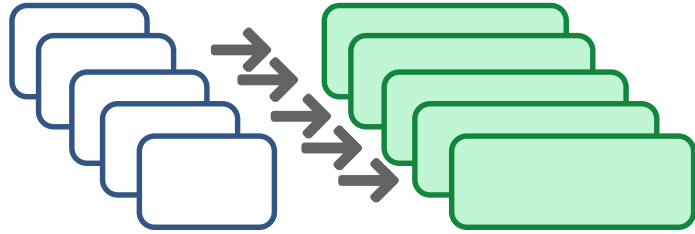
## › Podle doby zpracování

- Real time
  - reakce na vstupní zprávu je typicky dokončena v řádu jednotek až stovek milisekund
- Near-real time
  - reakce na vstupní zprávu je typicky dokončena v řádu jednotek až desítek sekund

## › Podle technologie zpracování zpráv

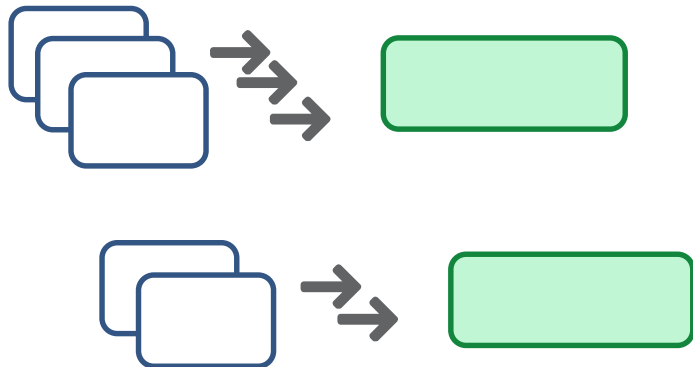
- Real time streaming
  - skutečné zpracování jednotlivých zpráv, jak přicházejí jedna za druhou
- Micro-batches
  - sekvenční spouštění malých dávek,
  - tj. nezpracovávají se zprávy ihned po příchodu, ale nejprve se nahromadí malá množina dat a ta se zpracuje jako celek

# Real time streaming vs micro batches



## Real time streaming

- každá zpráva je zpracována nezávisle



## Micro batches

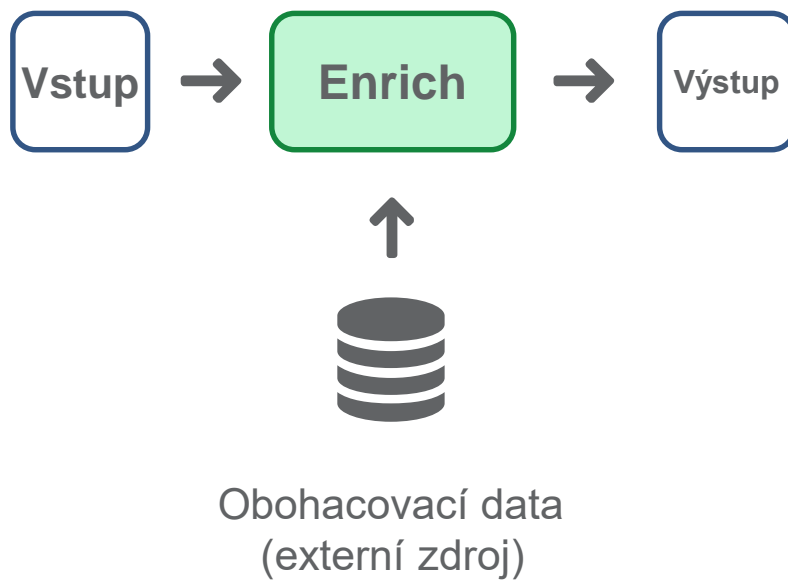
- zprávy jsou zpracovány v (malých) dávkách najednou

# Real time streaming vs micro batches

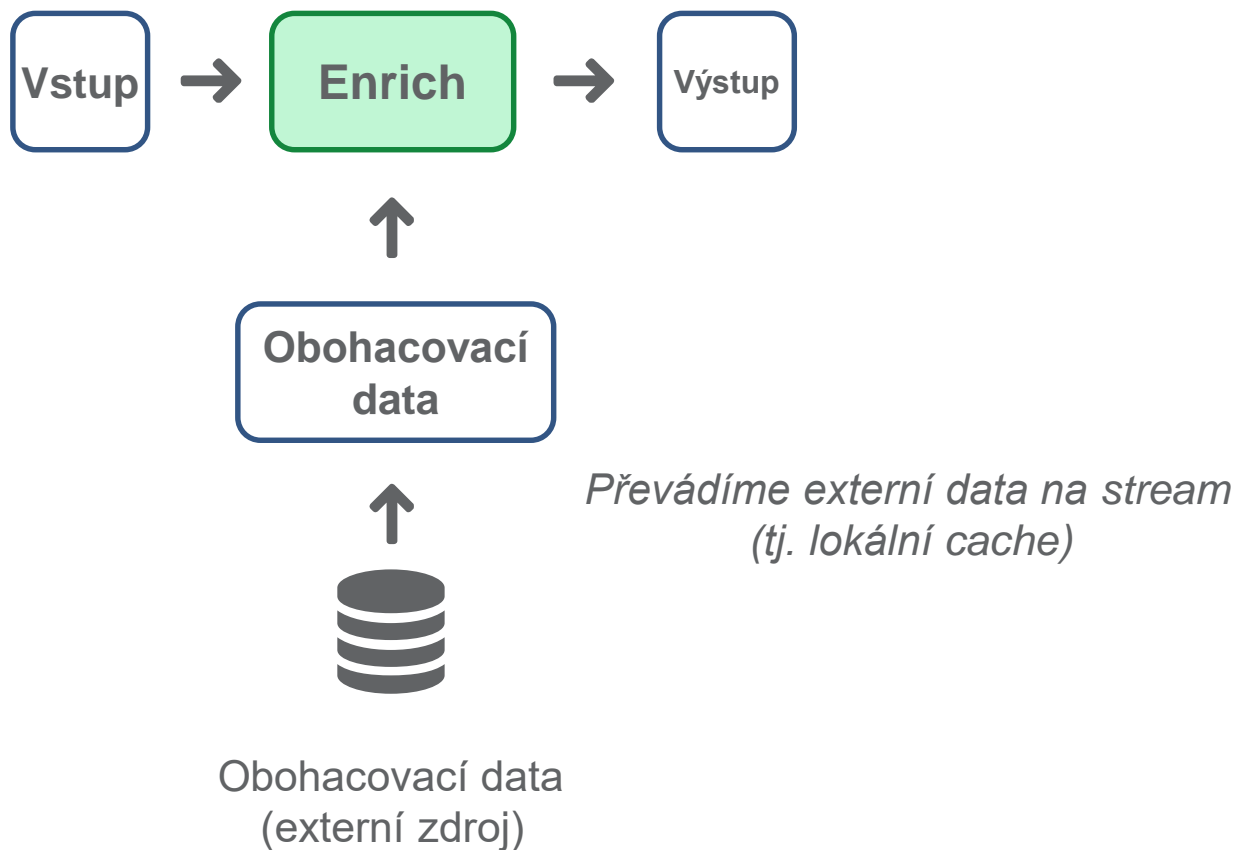
- › Zpracování záznamu po záznamu
- › Minimální latence
- › Menší prostupnost (průměrný počet zpráv za sekundu)  
→ neustále se zlepšuje
- › Vyžaduje pro realtime zpracování a batch zpracování samostatný kód
- › Zpracování množství záznamů najednou
- › Latence nejméně délka batch
- › Typicky vyšší prostupnost
- › Lze použít stejný kód pro streamové zpracování micro batches i pro „velké“ batches



# Streamové zpracování - architektura



# Streamové zpracování - architektura





# Sémantiky streamového zpracování

- › Exactly once
  - za každých okolností zajistíme, že zpráva bude doručena
  - vyžaduje nějaký způsob checkpointování, aby se dalo zjistit, jaké zprávy byly zpracovány, v případě, že dojde k havárii procesu
- › At least once
  - zpráva se může doručit více než jednou
  - velmi častý kompromis
- › At most once
  - zprávu nikdy nedoručíme opakovaně, může také nastat, že zpráva bude zcela ztracena
  - pouze pro nedůležitá data/data, která brzy ztrácejí cenu

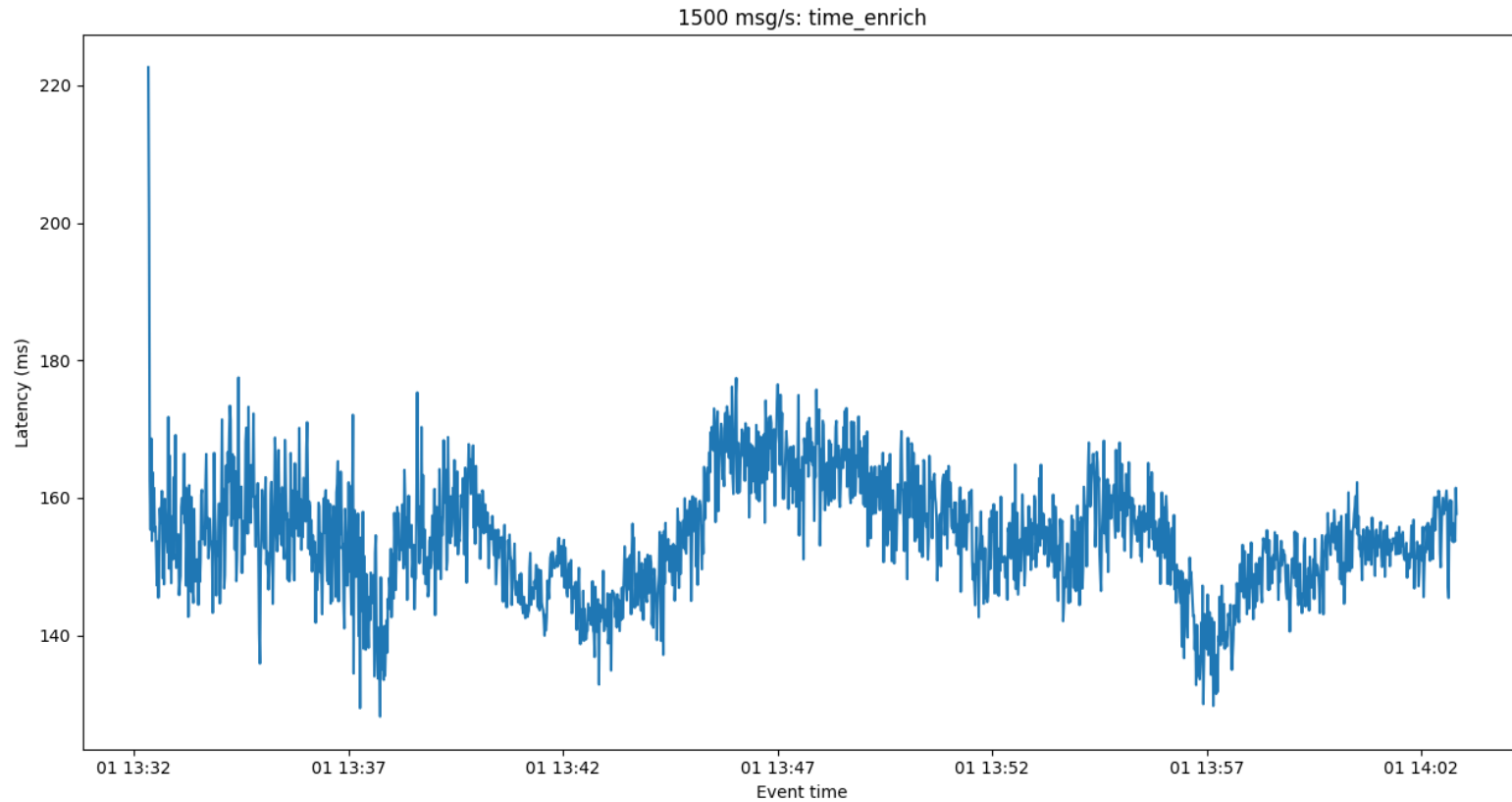
# Idempotence

- › Nástroje pro stream zpracování obsahují možnosti, jak zajistit exactly once sémantiku v rámci jednoho procesu
- › Ale je velmi složité zajistit toto v rámci více návazných procesů!
- › **Idempotence je takové chování, kdy opětovné doručení totožné zprávy nezmění stav systému**
- › Jak to řešit?
  - Např. pokud jediný výstup je databáze (HBase)
  - Doručení již existující zprávy způsobí uložení identických dat pod stejným klíčem (tj. systém není dotknut)

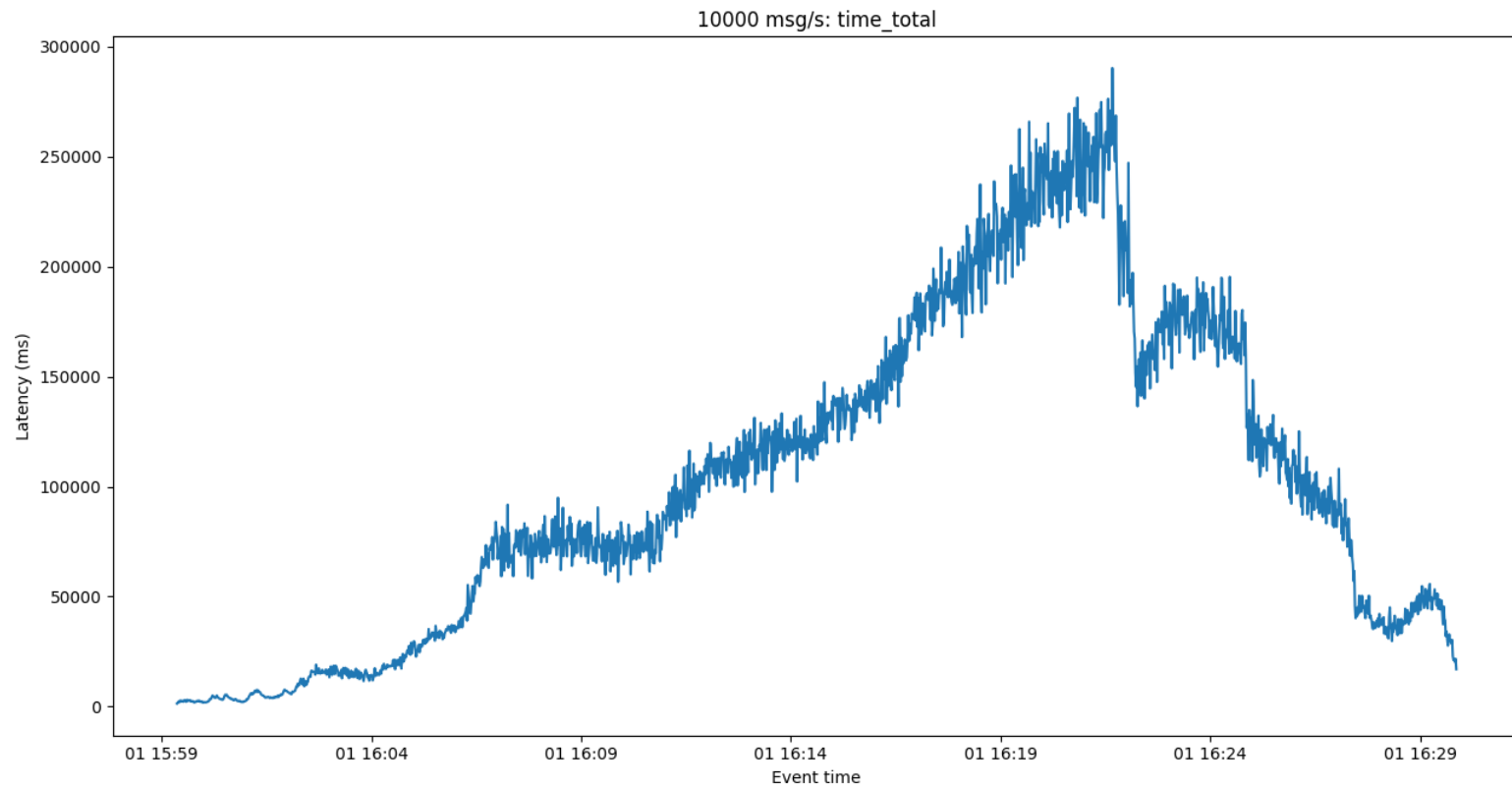
# Používané nástroj

- › Apache Spark Streaming
  - pouze microbatche a velké batche
- › Apache Flink
  - real time streaming,
  - podporuje i čistě batchové zpracování
- › Apache Storm
  - real time streaming
- › Kafka Streams

# Korektní průběh zpracování



# Kumulace zpoždění



# Úložiště - HBase

# HBase

- › <https://hbase.apache.org/>
- › NoSQL „databáze“
  - dotazování de facto programaticky (Java), žádný vhodný „SQL“ jazyk nemá
- › Key/value storage
  - vhodná konstrukce klíče je základ pro použití HBase!
- › Ukládá data na HDFS
- › Velmi rychlý přístup k datům podle klíče
  - na rozdíl Impala a Hive random access!
- › Velmi pomalý full scan
- › Velmi dobrá horizontální škálovatelnost
  - cca 4000-5000 dotazů za sekundu per node
- › Je třeba detailní znalost, při velkém množství dat je třeba správně nakonfigurovat
- › ~~Zajímavost: Využívá Facebook pro messaging~~

# HBase data model

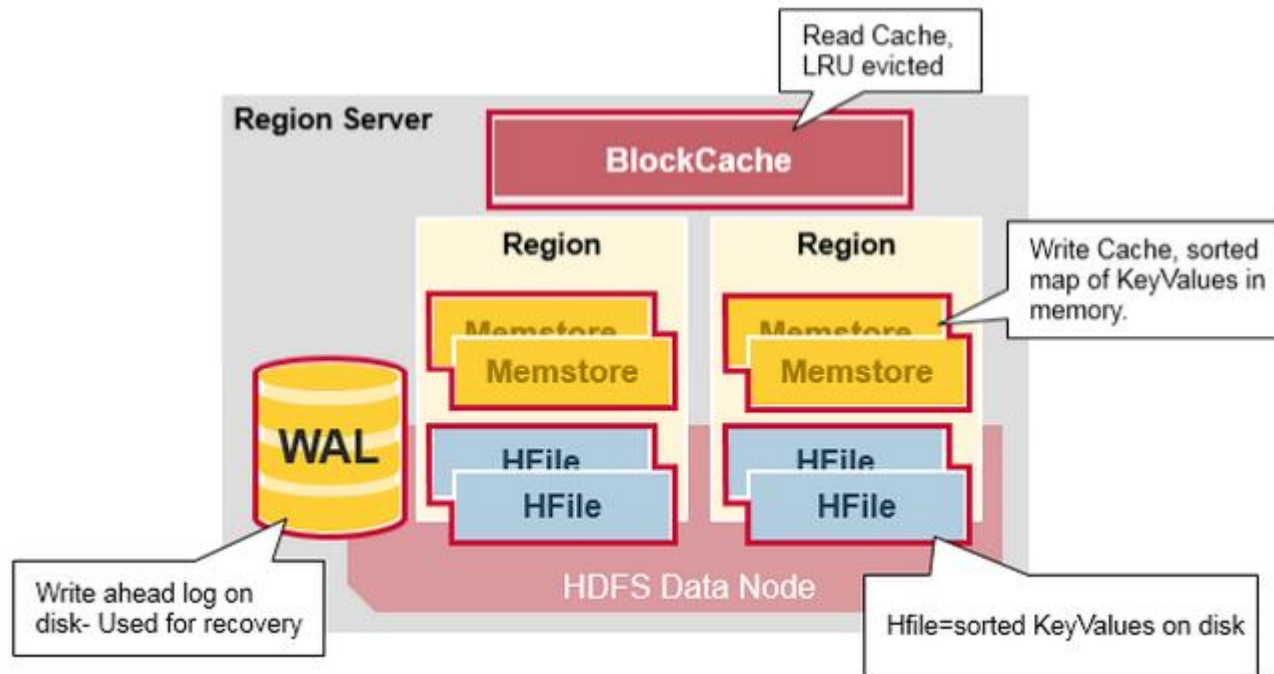
- › Table
- › Row
  - pro každý jednoznačný klíč
- › Column Family
  - sloupce v jedné Column Family jsou vždy ukládány společně
  - naopak, velká data (např. obrázky), která se načítají zřídka, lze dát do jiné Column Family a omezit tím množství čtených dat
- › Column
- › Version
- › HBase neobsahuje datové typy!



# Operace

- › Get
  - načtení záznamu podle klíče
- › Put
  - uložení záznamu
- › Scan
  - sekvenční načítání dat daného rozsahu klíčů nebo všech dat

# HBase – architektura

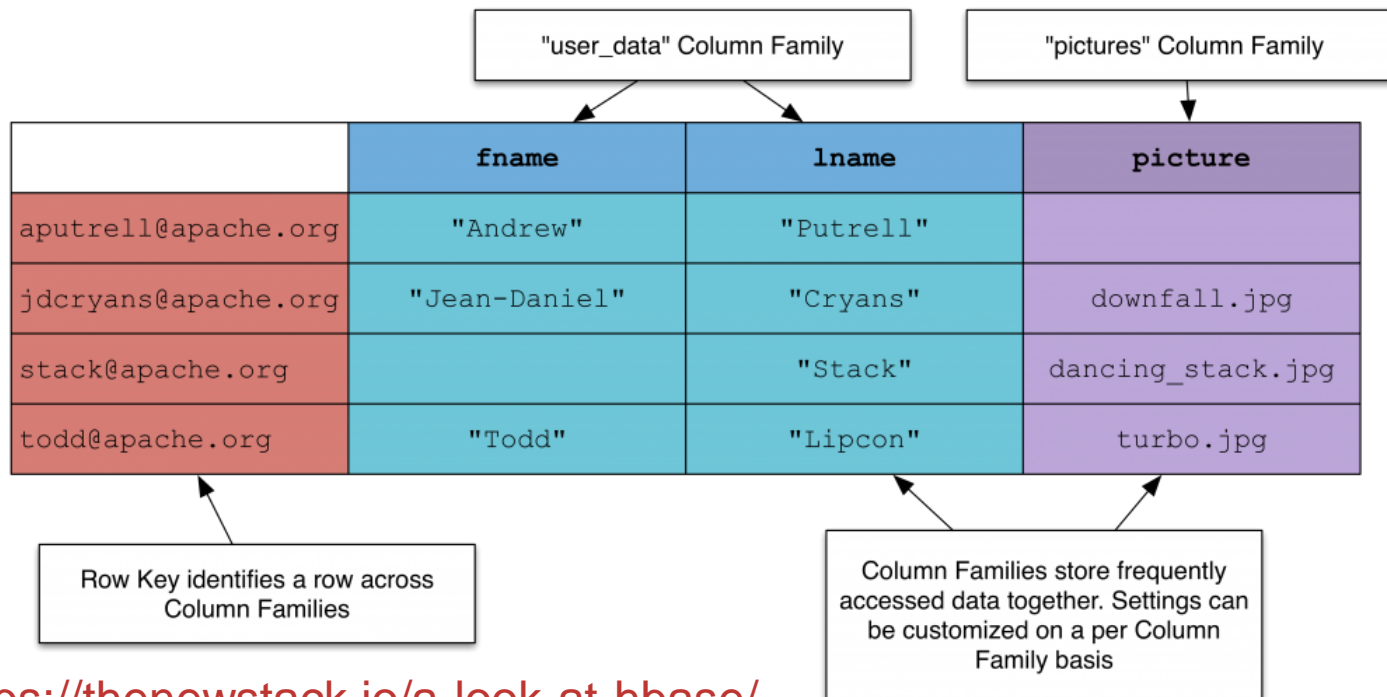


› <https://www.mapr.com/blog/in-depth-look-hbase-architecture>

# HBase – použití

- › Vhodné použití
  - jednoduché dotazy, hodně jednoduchých dotazů
  - více se čte, než zapisuje
  - odpověď je třeba velmi rychle (stovky ms)
  - k datům se přistupuje jen podle klíče, příp. počáteční části klíče
  - není třeba načítat velké množství dat sekvenčně
- › Nevhodné
  - analytické dotazy
  - průchod daty/scan
  - pouze zápisy (nebo nepoměrně mnoho zápisů vůči čtení)
- › Základem je **konstrukce klíče**, podle kterého se dotazuje
- › Lze použít tam, kde je jasně definovaný use-case
  - na HBase nelze snadno stavět obecná/flexibilní řešení

# HBase – příklad



<https://thenewstack.io/a-look-at-hbase/>

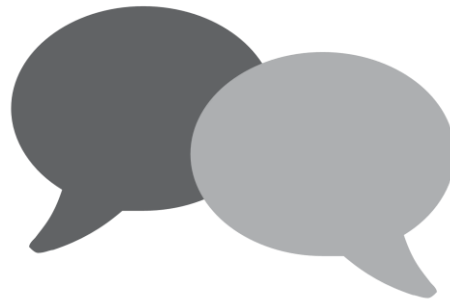
## › Column family

- skupina sloupců, které spolu souvisí – často se načítají společně
- zajištěno, že v HDFS jsou uloženy společně

# HBase – příklad

```
public class RetriveData{
    public static void main(String[] args) throws IOException,
Exception{
    // Instantiating Configuration class
    Configuration config = HBaseConfiguration.create();
    // Instantiating HTable class
    HTable table = new HTable(config, "cli");
    // Instantiating Get class
    Get g = new Get(Bytes.toBytes("stack@apache.org"));
    // Reading the data
    Result result = table.get(g);
    // Reading values from Result class object
    byte [] value =
result.getValue(Bytes.toBytes("user_data"),Bytes.toBytes("lname"));
    }
}
```

# Diskuze



`milan.kratochvil@profininit.eu`

# Díky za pozornost

**PROFINIT**

NÁSKOK DÍKY ZNALOSTEM

Profinit EU, s.r.o.

Tychonova 2, 160 00 Praha 6 | Telefon + 420 224 316 016



Web  
[www.profinet.eu](http://www.profinet.eu)



LinkedIn  
[linkedin.com/company/profinet](https://linkedin.com/company/profinet)



Twitter  
[twitter.com/Profinit\\_EU](https://twitter.com/Profinit_EU)



Facebook  
[facebook.com/Profinit.EU](https://facebook.com/Profinit.EU)



Youtube  
Profinit EU