AWS

SUMMIT

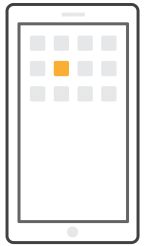
Real-time Streaming Applications on AWS – Patterns and Use Cases

Christian Deger, Chief Architect, AutoScout24
Dr. Steffen Hausmann, Solutions Architect, AWS

May 18, 2017

amazon web services

Most data is produced continuously

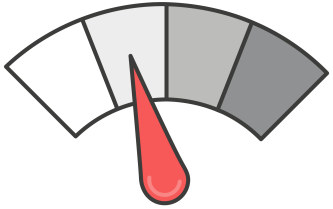


Mobile apps

Web clickstream

```
[Wed Oct 11 14:32:52
2000] [error] [client
127.0.0.1] client
denied by server
configuration:
/export/home/live/ap/h
tdocs/test
```

Application logs



Metering records

IoT sensors

Smart buildings

Simple Pattern for Streaming Data

Data Producer

Continuously creates data

Continuously writes data to a stream

Can be almost anything

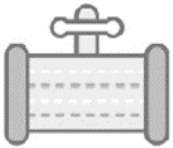
Mobile Client

Streaming Storage

Durably stores data

Provides temporary buffer

Supports very high-throughput



Amazon Kinesis

Data Consumer

Continuously processes data

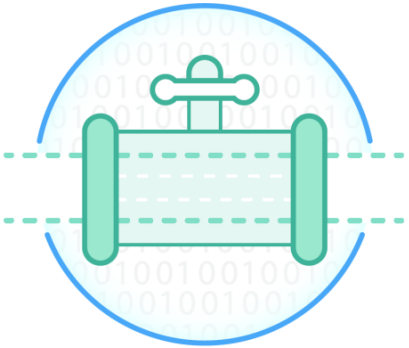
Cleans, prepares, & aggregates

Transforms data to information

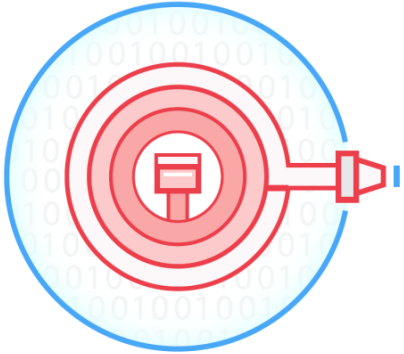
Amazon Kinesis app

Amazon Kinesis: Streaming Data Made Easy

Services make it easy to capture, deliver, process streams on AWS



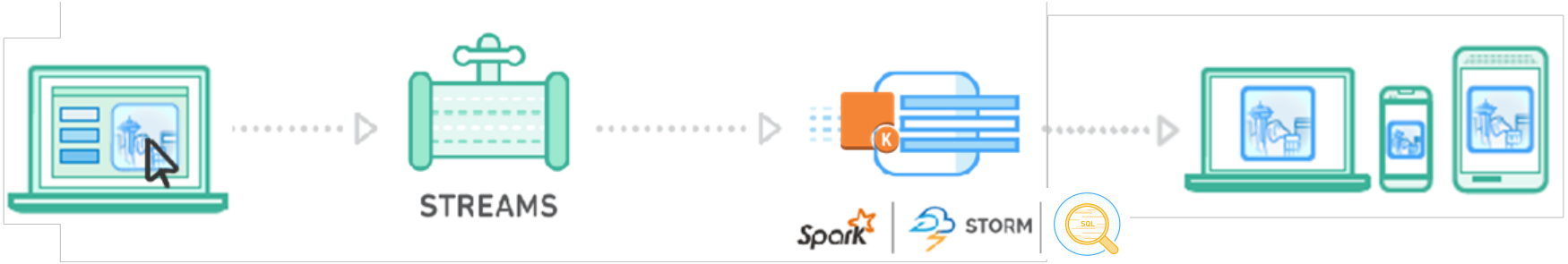
Amazon Kinesis Streams



Amazon Kinesis Firehose

Amazon Kinesis Analytics

Amazon Kinesis Streams



Send clickstream data to Kinesis Streams

Kinesis Streams stores and exposes clickstream data for processing

Custom application built on Kinesis Client Library makes real-time content recommendations

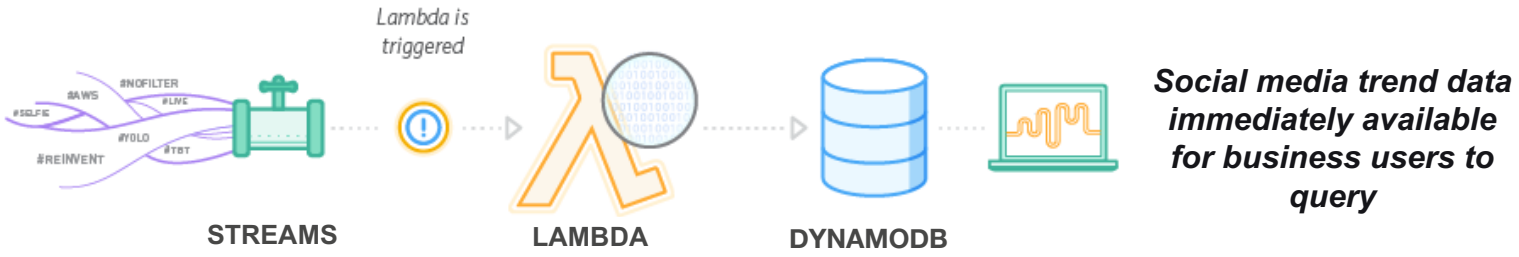
Readers see personalized content suggestions

- Easy administration
- Build real time applications with framework of choice
- Low cost

Amazon Kinesis Client Library

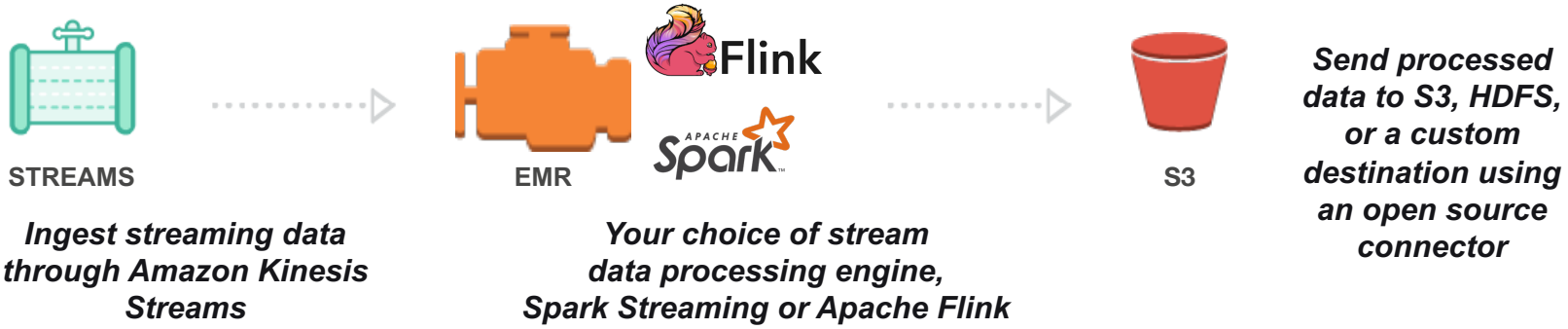
- Build applications with Kinesis Client Library (KCL) in Java, .NET, Ruby, Python, or Node.JS
- Deploy on your EC2 instances
- Two primary components:
  1. **Record Processor** – Processor unit that processes data from a shard in Amazon Kinesis Streams
  2. **Worker** – Processing unit that maps to each application instance
- Key features include load balancing, check pointing and CloudWatch metrics

AWS Lambda



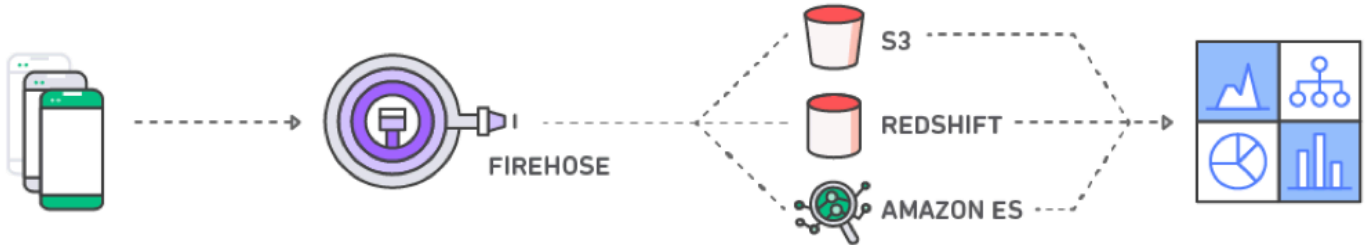
- Function code triggered from newly arriving events
- Simple event-based processing of records
- Serverless processing with low administration

Amazon Elastic Map Reduce (EMR)



- Ingest streaming data from many sources
- Easily configure clusters with latest versions of open source frameworks
- Less underlying performance management

Amazon Kinesis Firehose



Capture and submit streaming data

Firehose loads streaming data continuously into Amazon S3, Redshift and Elasticsearch

Analyze streaming data using your favorite BI tools

- Zero administration and seamless elasticity
- Direct-to-data store integration
- Continuous data transformations

Easily capture, process, and deliver data

- Write data to a Firehose delivery stream from a variety of sources
- Transform, encrypt, and/or compress data along the way
- Buffer and aggregate data by time and size before it is written to destination
- Elastically scales with no resource provisioning

Amazon Kinesis Analytics

Connect to Kinesis streams, Firehose delivery streams

Run standard SQL queries against data streams

Kinesis Analytics can send processed data to analytics tools so you can create alerts and respond in real-time

- Apply SQL on streams
- Build real-time, stream processing applications
- Easy scalability

Stream Processing Use Cases

Use Case	Characteristics
Streaming Ingest-Transform-Load	• Ingest and store raw data at high volume • Atomic transformations • Simple data enrichment
Continuous Metric Generation	• Windowed analytics (count X over 5 minutes) • Event correlation like sessionization • Visualization
Actionable Insights	• Act on the data by triggering events or alerts • Machine learning • Real-time feedback loops

Try these use cases yourself

Many variations of these use cases have sample code on the AWS Big Data Blog. Follow the blog!

- Analyzing VPC Flow Logs with Amazon Kinesis Firehose, Amazon Athena, and Amazon QuickSight
- Build a Real-time Stream Processing Pipeline with Apache Flink on AWS
- Real-time Clickstream Anomaly Detection with Amazon Kinesis Analytics
- Writing SQL on Streaming Data with Amazon Kinesis Analytics | Part 1, Part 2

AWS

SUMMIT

Streaming Use Cases at AutoScout24

Christian Deger, Chief Architect, AutoScout24

amazon web services

STRATEGIC GOALS

Goals of the business side

REDUCE TIME TO MARKET

Establish fast feedback loops to learn, validate and improve. Remove friction, hand-offs and undifferentiated work.

SUPPORT DATA-DRIVEN DECISIONS

Provide relevant metrics and data for user and market insights. Validate hypothesis for problems worth solving.

MOBILE FIRST

Start small and use device capabilities.

BEST TALENT

Autonomy, purpose and mastery: We know why we do things, we decide how to approach them and deliberately practice our skills.

COST EFFICIENCY

Run your segment in the right balance of cost and value.

ONE SCOUT IT

Foster collaboration. Harmonize and standardize tools. Pull common capabilities into decoupled platform services.

ARCHITECTURAL PRINCIPLES

High-Level Principles

ORGANIZED AROUND BUSINESS CAPABILITIES

Build teams around products not projects. Follow the domain and respect bounded contexts. Make boundaries explicit. Inverse Conway Maneuver.

ELIMINATE ACCIDENTAL COMPLEXITY

Strive to keep it simple. Don't over-engineer. Focus on necessary domain complexity.

LOOSELY COUPLED

By default avoid sharing and tight coupling. No integration database. Don't create the next monolith.

MACRO AND MICRO ARCHITECTURE

Clear separation. Autonomous micro services within the rules and constraints of the macro architecture.

SECURITY, COMPLIANCE AND DATA PRIVACY

Build with least privilege and data privacy in mind. Know your threat model. Limit blast radius.

AWS FIRST

Favor AWS platform service over managed service, over self-hosted OSS, over self built solutions.

DESIGN AND DELIVERY PRINCIPLES

Tactical measures

YOU BUILT IT, YOU RUN IT

The team is responsible for shaping, building, running and maintaining its products. Fast feedback from live and customers helps us to continuously improve.

CROSS-FUNCTIONAL TEAMS

Engineers from all backgrounds work together in collaborative teams as engineers and share responsibilities. No silos.

AUTONOMOUS TEAMS

Make fast local decisions. Be responsible. Know your boundaries. Share findings.

BE BOLD

Go into production early. Value monitoring over tests. Fail fast, recover and learn. Optimize for MTTR not MTBF.

DATA-DRIVEN/METRIC-DRIVEN

Collect business and operational metrics. Analyze, alert and act on them.

INFRASTRUCTURE AS CODE

Automate everything: Reproducible, traceable, auditable and tested. Immutable servers.

STRATEGIC GOALS

Goals of the business side

REDUCE TIME TO MARKET

Establish fast feedback loops to learn, validate and improve. Remove friction, hand-offs and undifferentiated work.

SUPPORT DATA-DRIVEN DECISIONS

Provide relevant metrics and data for user and market insights. Validate hypothesis for problems worth solving.

MOBILE FIRST

Start small and use device capabilities.

BEST TALENT

Autonomy, purpose and ma… things, we decide how to ap… practice our skills.

COST EFFICIENCY

Run your segment in the rig…

ONE SCOUT IT

Foster collaboration. Harmonize and standardize tools.
Pull common capabilities into decoupled platform services.

ARCHITECTURAL PRINCIPLES

High-Level Principles

ORGANIZED AROUND BUSINESS CAPABILITIES

Build teams around products not projects. Follow the domain and respect bounded contexts. Make boundaries explicit. Inverse Conway Maneuver.

ELIMINATE ACCIDENTAL COMPLEXITY

Strive to keep it simple. Don't over-engineer.
Focus on necessary domain complexity.

LOOSELY COUPLED

By default avoid sharing and tight coupling.
No integration database. Don't create the next monolith.

MACRO AND MICRO ARCHITECTURE

Clear separation. Autonomous micro services within the rules and constraints of the macro architecture.

…MPLIANCE AND SAFE…

Know your threat model. Limit blast radius.

AWS FIRST

Favor AWS platform service over managed service, over self-hosted OSS, over self built solutions.

DESIGN AND DELIVERY PRINCIPLES

Tactical measures

YOU BUILT IT, YOU RUN IT

The team is responsible for shaping, building, running and maintaining its products. Fast feedback from live and customers helps us to continuously improve.

CROSS-FUNCTIONAL TEAMS

Engineers from all backgrounds work together in collaborative teams as engineers and share responsibilities. No silos.

AUTONOMOUS TEAMS

Make fast local decisions. Be responsible. Know your boundaries. Share findings.

BE BOLD

Go into production early. Value monitoring over tests. Fail fast, recover and lea…rn. Optimize for MTTR not MTBF.

…-DRIVEN/METRIC-DRIVEN

…usiness and operational metrics. Analyze, alert and act on them.

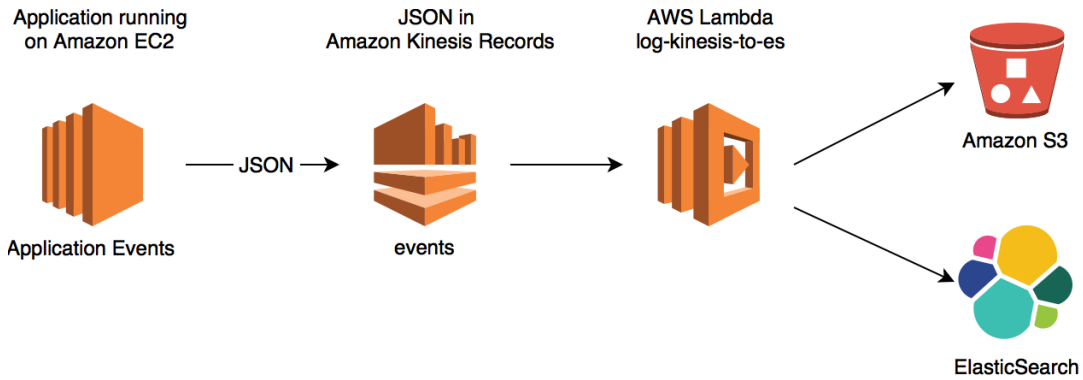
INFRASTRUCTURE AS CODE

Automate everything: Reproducible, traceable, auditable and tested. Immutable servers.

AWS FIRST

Favor AWS platform service over managed service, over self-hosted OSS, over self built solutions.

AutoScout24 Logging Infrastructure



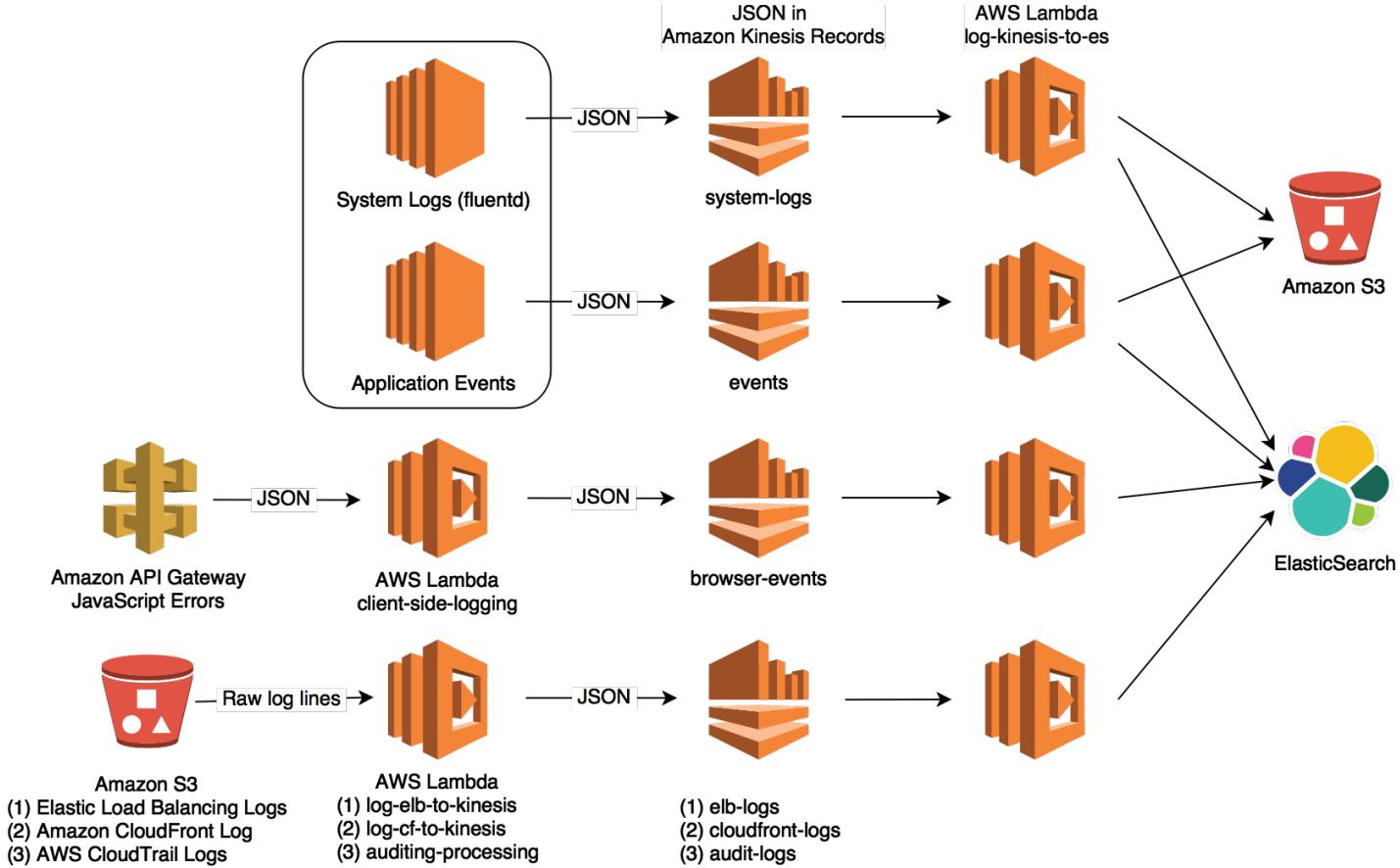
Why Amazon Kinesis Streams?

- Decouple producer from consumer
  - No events lost, when a consumer is not available
  - Replay past events
  - Buffer peak loads
  - Fan out to multiple consumers
- AWS Lambda integration
- Reliable and scalable managed service

Managed Service != No Ops

- Manage capacity (shards)
  - Cost optimization
  - Batching
- Manage limits
  - Kinesis shard limit
  - Lambda concurrent executions
- Monitor metrics
  - Errors
  - Throttling/ Lag

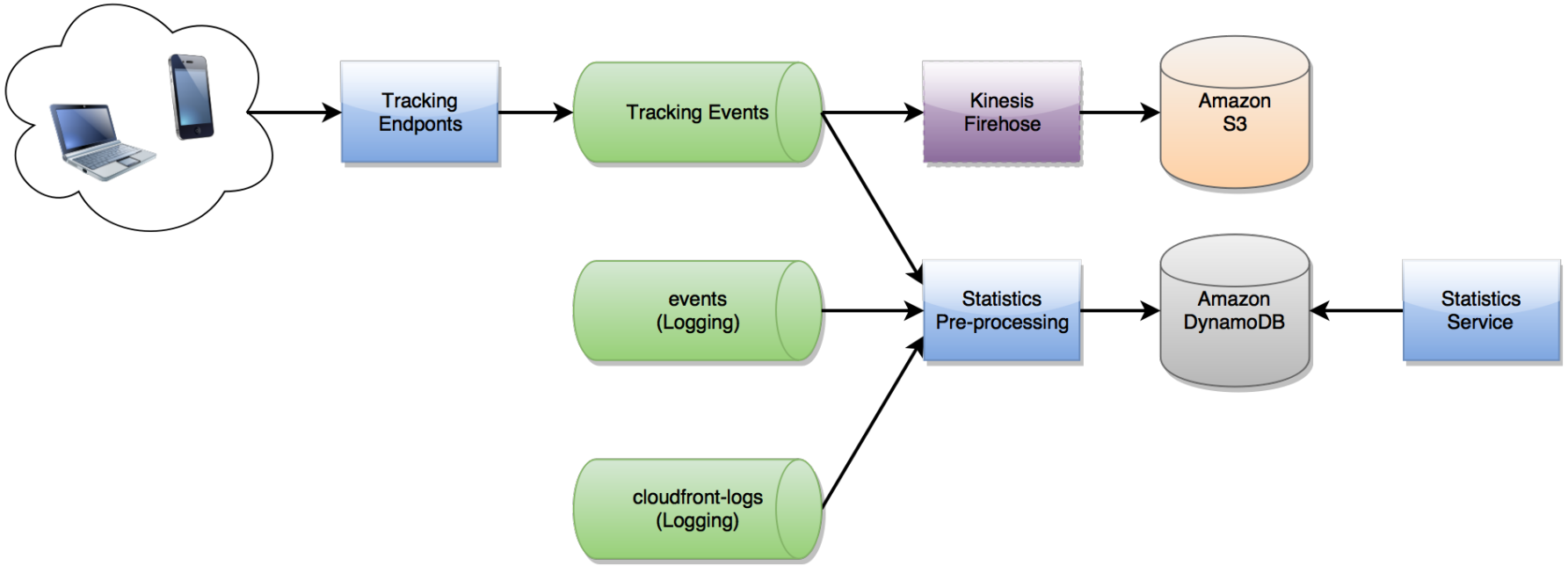
Expanded concept



Numbers ~

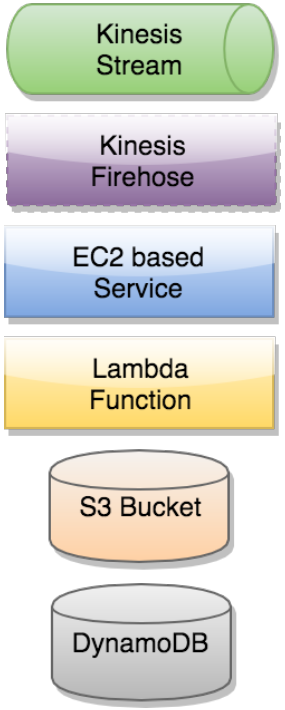
- Log size per day: 4 TB
- Number of events per day: 1.7 billion
- Lambda invocations per second: 3,500
- Kinesis cost per month: $1,700
- Lambda cost per month: $1,700

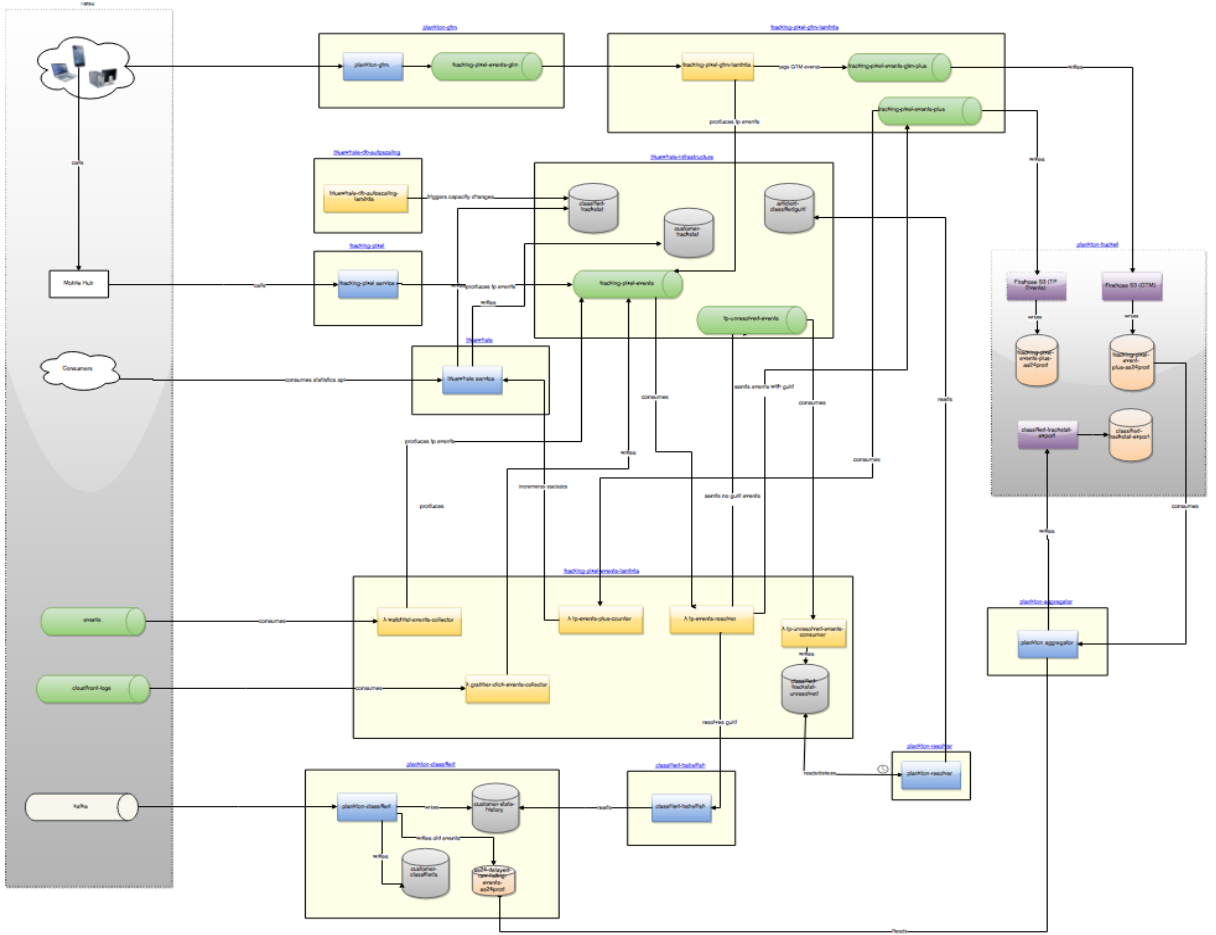
AutoScout24 Real-time Statistics - Concept



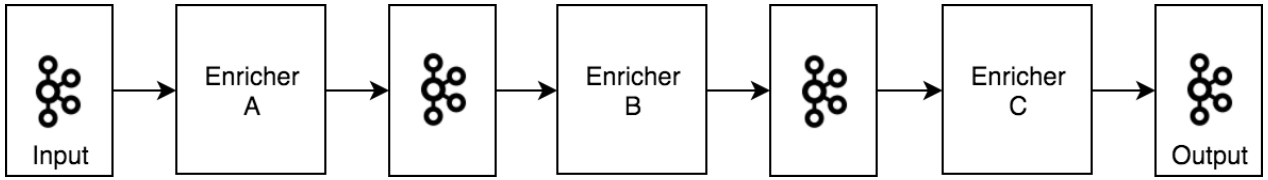
AutoScout24 Real-time Statistics - Architecture

Service Legend:





AutoScout24 Listings pipeline



Use case:

- Stream listing changes.
- Only interested in latest version a listing.
- Time-based retention not useful.

Solution Kafka

- Record based retention: Log compaction
- No managed service: Need to operate a cluster.

AWS

SUMMIT

Thank you!

amazon web services