# Big Data Real-Time Analytics

Knut Nesheim · @knutin

**GameAnalytics**

# Topics

- What we do
- What we want
- Our solution

# Me

- Klarna
- Wooga
- Game Analytics
- github.com/knutin

# Analytics SaaS for games

# Game Analytics

- Startup, venture funded, ~2 years old
- HQ in Copenhagen, engineering in Berlin
- Need to move fast
- Willing to take on technical debt
- Be ready for traffic growth
- Big games are *big*, millions of DAU

```
GA.Event.Business(“sheep”, “gold”, 200);
```

# Metrics

- Daily Active Users, Monthly Active Users
- Revenue
- Histogram of event values

# Idea: *some* real-time metrics

*Suitable subset*

# Scope creep: real-time *everything*

# MapReduce architecture, v1.0

iOS SDK

Android SDK

Unity SDK

HTTP

Collectors

S3

**Online**

MapReduce

Work queue

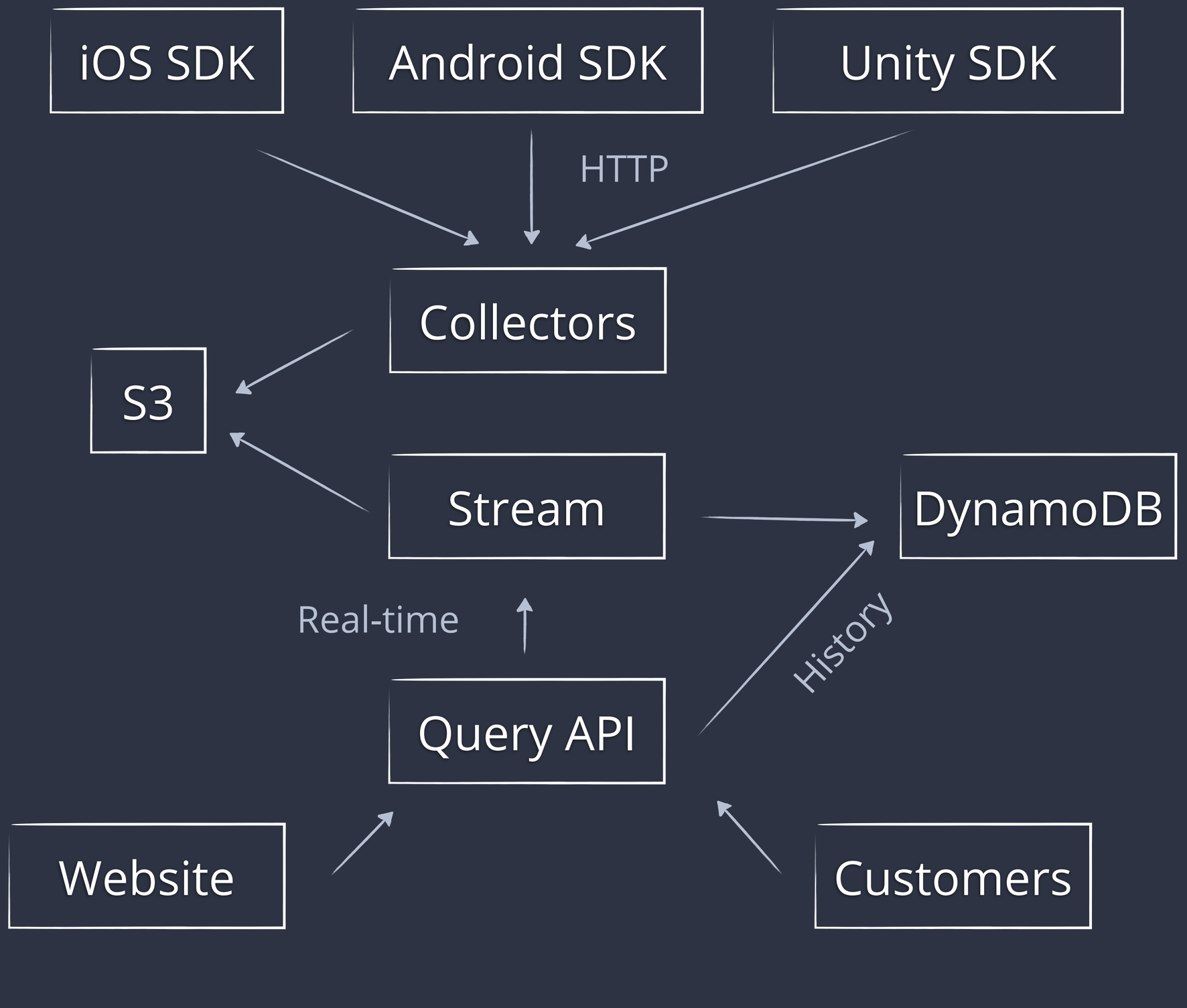
MongoDB

Special stuff

Website

**Offline**

# Streaming architecture, v2.0


iOS SDK
Android SDK
Unity SDK
HTTP
Collectors
S3
Stream
DynamoDB
Real-time
History
Query API
Website
Customers

Today

2013-10-14

2013-10-15

DynamoDB

RAM

# Streaming

- Partition on game
- One process per game, autonomous
- Prototype very promising

# Implementation

# Game process

- Process files sequentially
- Keep running results in RAM
- Flush to DB when window closes
- Answer real-time queries
- Put all in one node

# “Metric DSL”

```
handle_event(E) ->
    [{set_add, <<"DAU">>, {country, country(E)}, user_id(E)}].
```

```
apply_update({set_add, Name, Dimension, Value}, Metrics) ->
    HLL = case dict:find({Name, Dimension}, Metrics) of
              {ok, H} -> H;
              error   -> hyper:new()
          end,
    dict:store({Name, Dimension}, hyper:insert(Value, HLL), Metrics).
```

# HyperLogLog

- Estimate cardinality
- Clever hash tricks
- Millions unique with <1% error in ~40k words
- Unions
- “hyper”: Erlang HLL++ from Google paper[0]
- Will open source Soon (TM)

[0]: “HyperLogLog in Practice: Algorithmic Engineering of a State of The Art Cardinality Estimation Algorithm”

# Recordinality[0]

- Estimate frequency of values
- User session count
- Keeps a reservoir, clever hash tricks
- Will open source Soon (TM)

[0]: “Data Streams as Random Permutations: the Distinct Element Problem”

# Next step: More parallelization

## Game #123

```
loop(State) ->
    NewState = process(next_file(), State),
    loop(NewState).
```

**Worker #1**

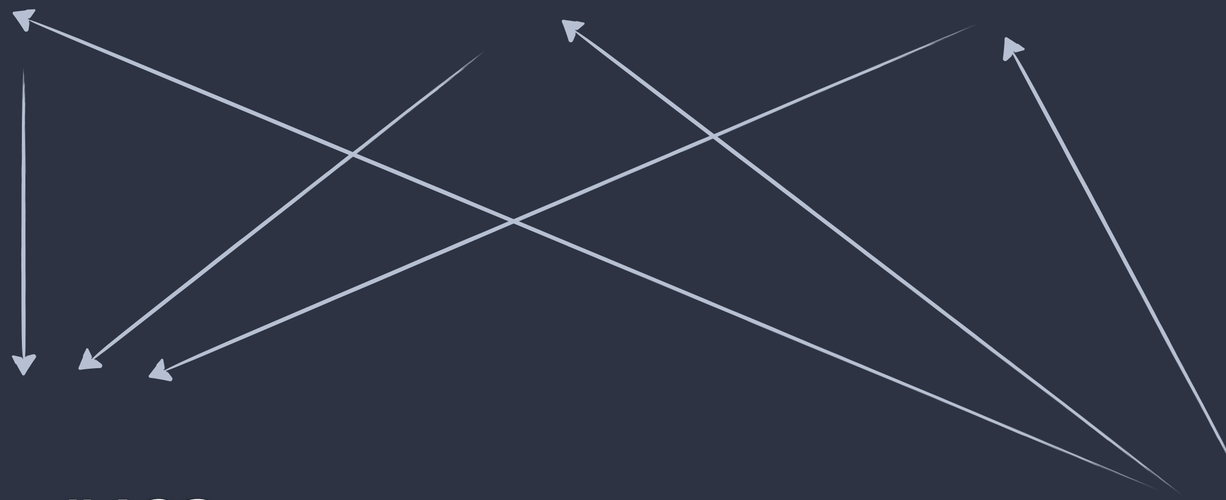
```
Part = process(next_file(), empty()),
game_123 ! Part.
```



**Game #123**

```
loop(State) ->
    receive
        Part ->
            NewState = merge(Part, State)),
            loop(NewState)
    end.
```

Worker #1 Worker #2 Worker #N



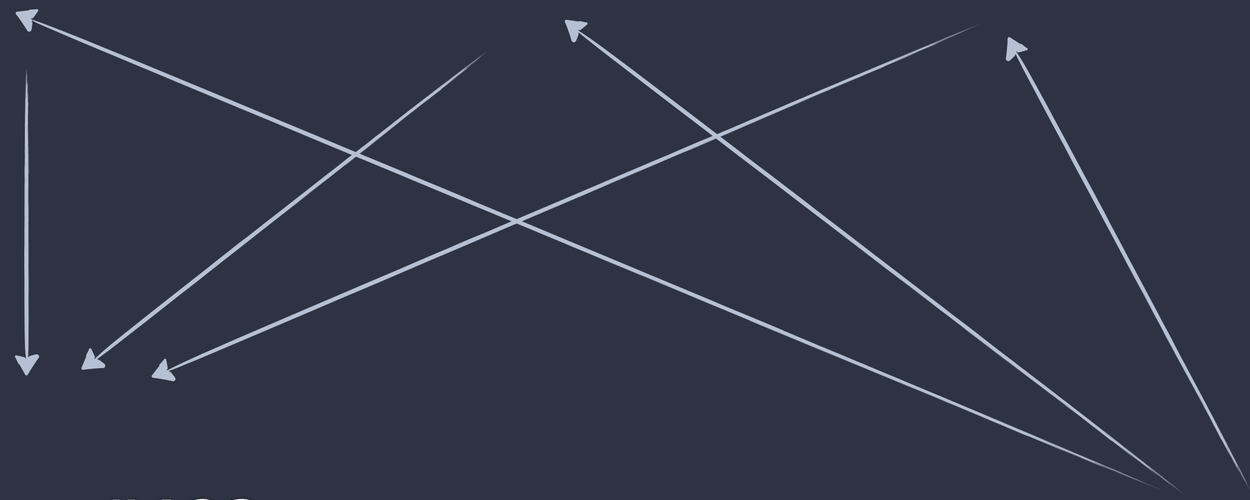
Game #123

Manager

Mapper #1

Mapper #2

Mapper #N



Reducer #123

Scheduler

# Scheduler

- Take ideas from Hadoop, Riak Pipe
- Manage limited resources
- Distribute work across nodes
- Colocate processing with state storage

# Implementation

- DIY?
- riak_core for state processes?
- riak_pipe for managing work?

# Conclusion

# Erlang: The bad parts

- Big process state not great
- Lots of updates, lots of garbage

# Erlang: The good parts

- Total allocated RAM >30GB
- Per process heap is gold
- Easy parallelization, distribution
- Looking forward to maps!

# Questions?

Knut Nesheim · @knutin

**GameAnalytics**

GameAnalytics

www.gameanalytics.com