upsolver

Whitepaper:

Data Lake ETL for IoT Data: From Streams to Analytics

Introduction

In 2017 alone, global spend on IoT reached a staggering sum of **$235 billion**, according to research by Bain and Company. The same report expects to see more than double that investment by 2021, with data and analytics poised to be the fastest growing segment.

While the hype around interconnected devices in the consumer space seems to have slowed down (with notable exceptions such as talking to your Samsung fridge), much of the focus has shifted towards the use of IoT devices and data in the manufacturing, healthcare and artificial intelligence sectors. But is the massive spend on these initiatives actually generating the expected returns? Not quite yet.

Taming IoT data proves easier said than done

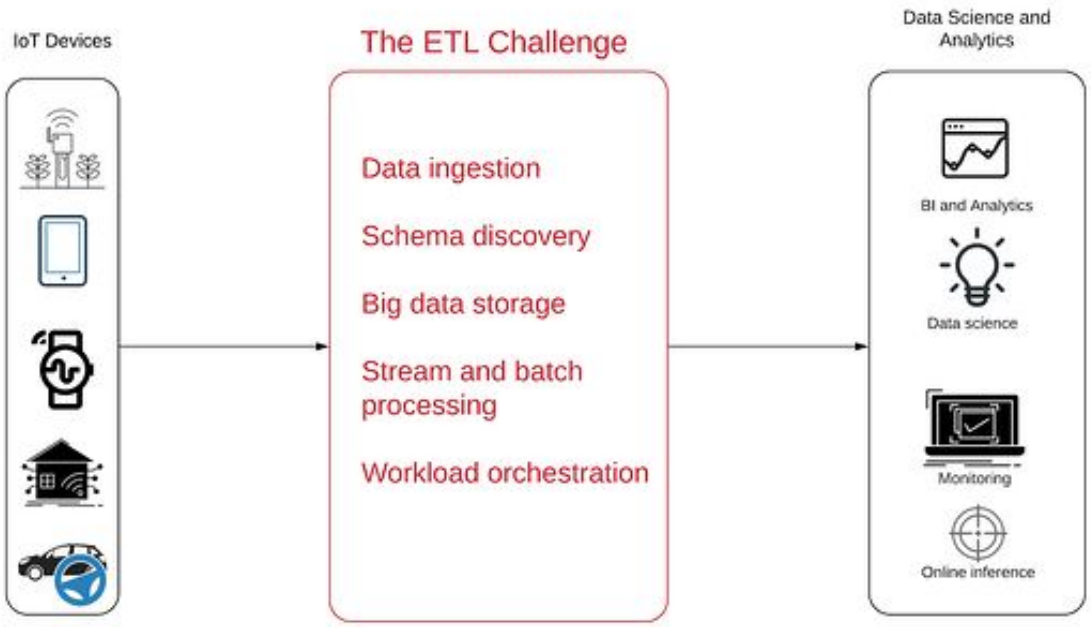
For most enterprises, IoT projects have yet to cross the proof-of-concept stage and are yet to show clear return on investment. Data is meant to play a key role in these projects, as web-enabled devices send a vast wealth of data that can ostensibly be used to improve operational efficiency and accelerate innovation through predictive analytics.

However, as with many big data projects, the challenges and costs can quickly become overwhelming. Sensors generate massive volumes of semi-structured data at high velocity, and traditional data processing and analytics tools struggle to keep up. Operationalizing IoT data requires organizations to develop expertise in emerging ETL technologies such as stream processing and data lakes, leading to IT bottlenecks that derail projects for months or years before any actual data science can be done.

In this whitepaper, we'll look at some of the data integration challenges and opportunities around IoT data, and suggest a reference architecture for simplifying data lake ETL for IoT streams using Upsolver on AWS.

Understanding the ETL challenge in IoT

Before we get into the solution, let's define the problem:



In the diagram above we have our data source on the left: IoT data is generated by innumerous devices with sensors embedded in them. These could be as simple as an antenna or as complex as an autonomous vehicle, but the common denominator is that these devices generate logs that are sent over the WWW in a continuous stream of semi-structured data.

On the right hand side we can see what we're trying to achieve in terms of data consumption. The types of analytic products that we might want to see at the end of an IoT project could include:

- **Business intelligence dashboards:** Product, BI and marketing teams need visibility into product usage trends and patterns
- **Operational monitoring:** Identifying outages and inactive devices in real-time

- **Anomaly detection:** Proactive alerting on peaks or precipitous drops in the data
- **Embedded analytics:** Giving customers the ability to see and understand their own usage data
- **Data science:** Advanced analytics and machine learning for predictive maintenance, route optimization or AI development

However, to reach that goal, data must first be transformed from its raw streaming form into analytics-ready tables that can be queried using SQL and familiar analytics tools. This is the process of extract-transform-load (ETL), which is frequently the most complex part of analytics projects - and especially so in light of the following:

Unique aspects of IoT data

If traditional databases, ETL and BI tools are built around structured tables, it's easy to see why they're not

immediately useful in the context of IoT data, which comes with a different set of characteristics, including:

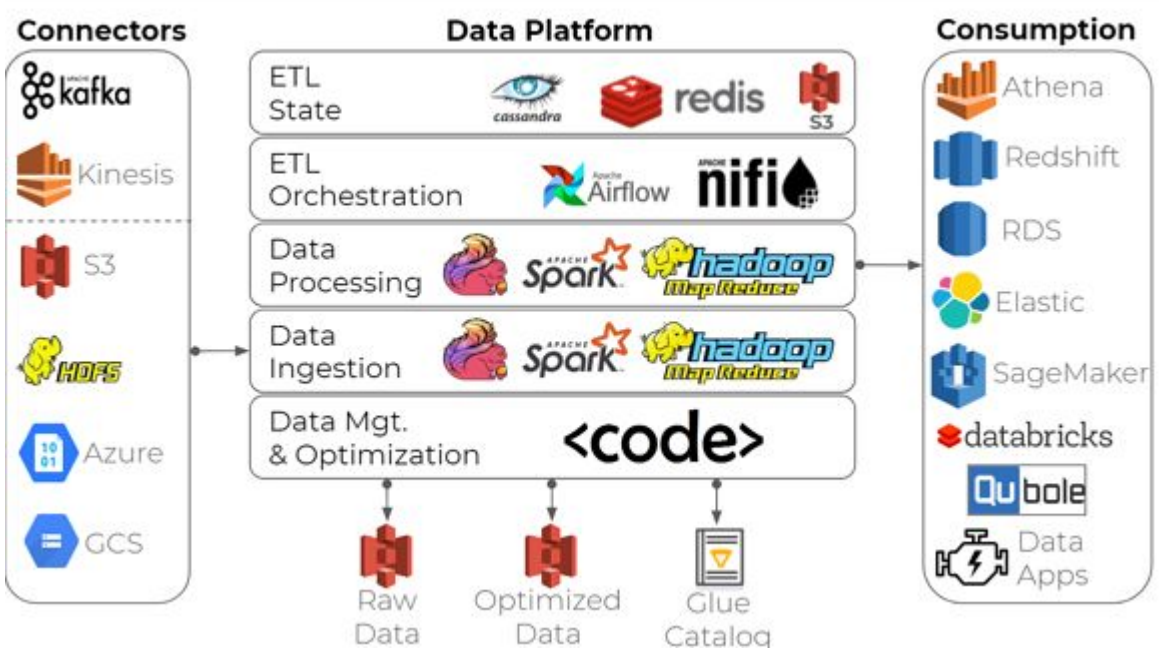
- **Streaming data:** Almost by definition, IoT data is streaming data. It is generated continuously in small files that combine to form massive, sprawling datasets, which makes it very different from traditional tabular data (read more about streaming data architecture), necessitating more complex ETL for joins, aggregations and data enrichment.
- **Store now, analyze later:** Since many of the use cases are exploratory rather than pre-defined, we will want to store large volumes of data in its original format to maintain flexibility in future analysis and support for historical replay. This means a cloud or on-premise data lake.
- **Unordered events due to multitude of devices:** Data-generating devices might move in and out of

internet connectivity areas, meaning logs arrive at our servers at different times and not necessarily in the ‘correct’ order.

- **Low-latency access is often required:** As we’ve seen above, for operational use cases it is often necessary to be able to identify anomalies or specific devices in real time or near-real time, which means we often can’t tolerate the latencies caused by batch processing.

Building a data platform using open-source frameworks

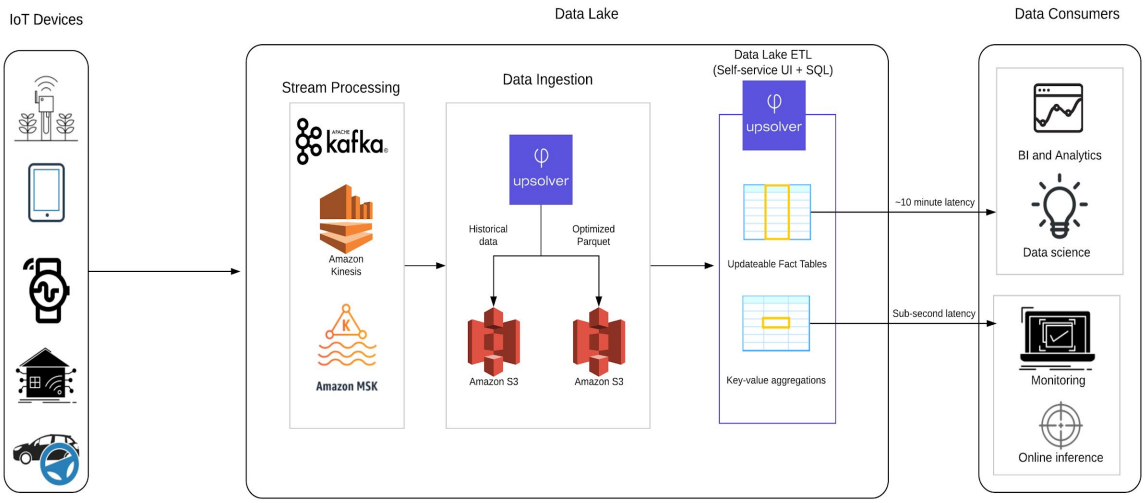
One common approach to building an enterprise data platform for IoT analytics is to create a data lake architecture using open-source stream processing frameworks and time-series databases such as Apache Spark/Hadoop, Apache Flink, InfluxDB, and many additional building blocks:



This open-source tool-set will get the job done, but implementing it correctly can be an overbearing task for all but the most data-savvy organizations. Building and orchestrating this type of data platform requires specialized big data engineers and a strong focus on data infrastructure, which is often not a core competency for the companies that actually work with IoT data in industries such as manufacturing and consumer electronics - which in turn leads to delayed delivery, exorbitant costs and thousands of engineering hours going to waste.

Building the IoT data platform with Upsolver on AWS

Reference Architecture



Upsolver is a data lake ETL platform that was built for turning event streams into analytics-ready datasets. The solution shown above provides high performance and a full breadth of functionality and use cases - including operational reporting, ad-hoc analytics and data preparation for machine learning.

However, the Upsolver platform is not limited by the same rigidity and complexity of Spark/Hadoop data platforms, and

is configured strictly through a self-service user interface and SQL, rather than intense coding in Java/Scala. This makes the solution fully useable by data consumers (analysts, data scientists, product managers) as well as data providers (DevOps, data engineering).

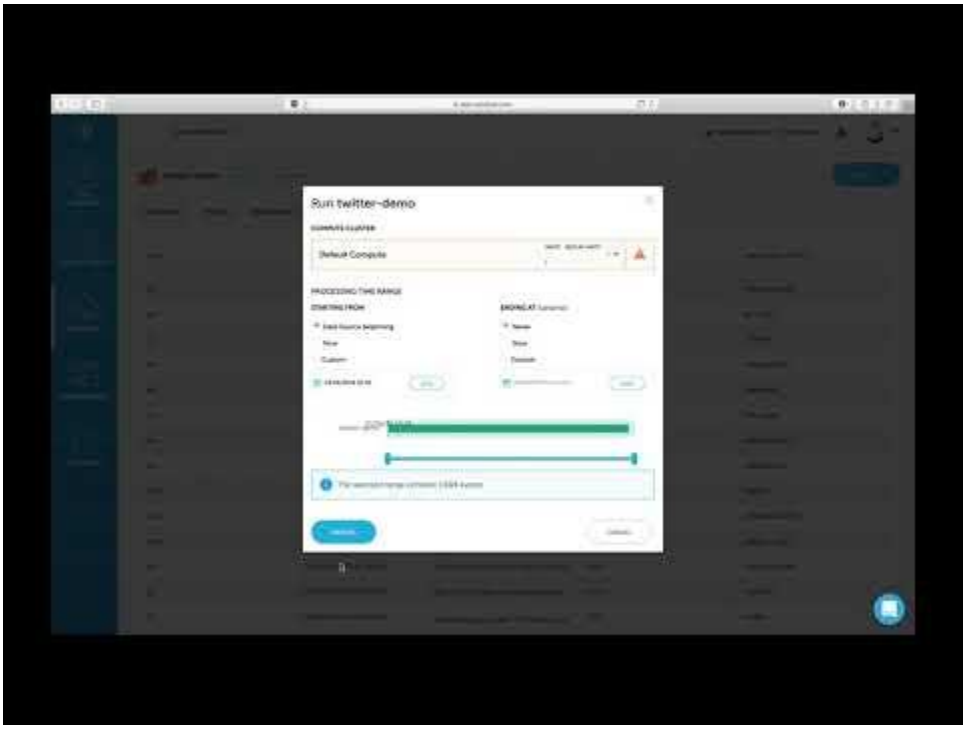
How it works (in a nutshell)

- Raw data is generated by IoT devices seen on the left, and then sent to a message broker such as Apache Kafka or Amazon Kinesis.
- Upsolver connects directly to said broker, automatically identifies schema and key statistics, and ingests the data into cloud object storage on Amazon S3 while ensuring exactly-once processing and enforcing data lake best practices such as data partitioning.
- Upsolver stores a copy of the raw event data for historical replay on S3, as well as optimized Parquet files that are ready for consumption using SQL engines such

as Amazon Athena.

- Users can then run ETL flows for joins between multiple streams, aggregations and enrichments using Excel-like functions or SQL notebooks. Upsolver can then write structured data to a database such as Amazon Redshift, to S3 or send table data and metadata to Amazon Athena (via the Glue Data Catalog).
- For near-real time stream processing, Upsolver Lookup Tables enable users to run sub-second queries against time series data by any key/aggregate combination, which can then be used for online machine learning as well as embedded analytics.

To get a better idea of how you can use Upsolver to build streaming data pipelines, check out this video:



Key benefits

- Self-service for data consumers without over-reliance on IT and data engineering
- Reduce infrastructure costs by optimizing ETL flows and big data storage
- Fully managed service that enables organizations to focus on features rather than infrastructure

- Single platform for both stream and batch processing - no need to maintain multiple systems for real-time data, ad-hoc analytics and reporting
- Fully secure in any VPC - data never leaves the customer's AWS account

Want to learn more about Upsolver for IoT?

- Grab a copy of our comprehensive technical white paper
- Pencil some time for a chat with one of our data experts
- Read our previous article on IoT analytics