EMQ

Data Infrastructure for IoT White Paper

www.emqx.com

Catalogue

Executive Summary

> **Now it's time to think about the business challenges and opportunities for enterprises in the IoT era.**

Over the past decade, EMQ has been developing open–source infrastructure software for IoT applications and solutions. We have delivered EMQX, the world's leading open–source MQTT message broker that solves the challenges of massive device connectivity. We also created HStreamDB streaming database for the storage, processing, and real–time analysis of IoT data.

After years of delivering basic IoT software for enterprises, we have found that most enterprises build IoT solutions with an application–centric approach.

Today, we believe that the core logic of IoT scenarios should be data–centric and obtain business insight from data to create value. In particular, medium and large enterprises should focus on data and think about IoT business from the perspective of data, building diverse and innovative IoT solutions based on a unified data infrastructure.

In this white paper, EMQ formally proposes the architecture paradigm of "**Data Infrastructure for IoT**" to meet the key business challenges of enterprises in the IoT era together with industry to achieve business innovation and value creation.

Digital Transformation Trends in the IoT Era

Cloud–Native

Definition

Cloud–native architecture and technologies are an approach to designing, constructing, and operating workloads built in the cloud and taking full advantage of the cloud computing model.

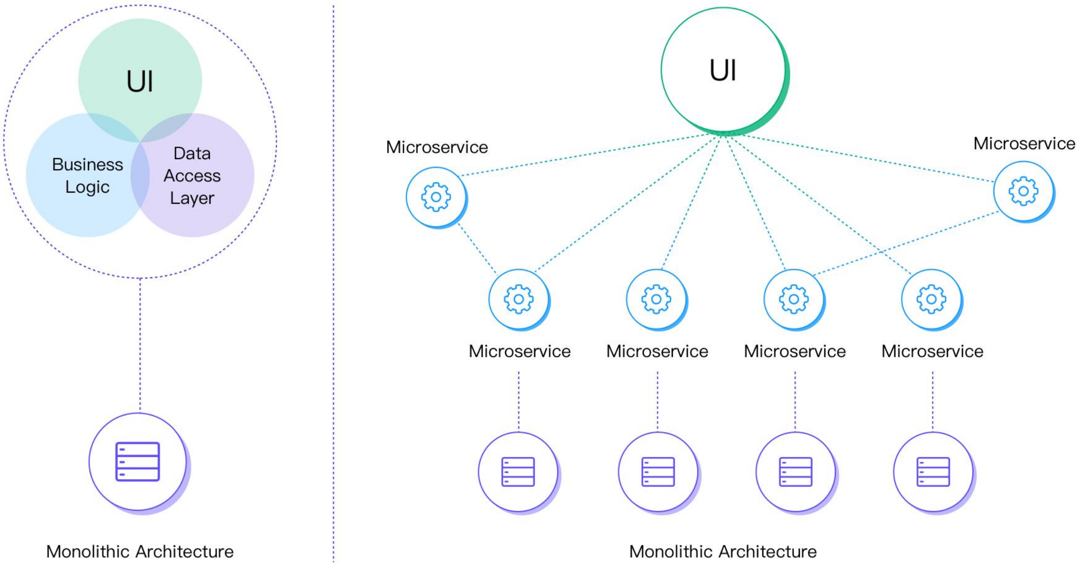
Cloud–native features extreme elasticity, service autonomy, fault self–healing, and replication at scale.

> According to CNCF's official definition, Cloud-native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.
>
> These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.
>
> *Source: https://github.com/cncf/toc/blob/main/DEFINITION.md

Architecture

Cloud-native systems embrace microservices to achieve speed and agility.

Cloud-native architecture is a design methodology that utilizes cloud services to allow dynamic and agile application development techniques that take a modular approach to build, run, and update the software through a suite of cloud-based microservices versus a monolithic application infrastructure. The microservice architecture best reflects the philosophy of cloud-native.



Edge Computing

The classic cloud computing model uploads all the data that needs to be calculated and stored to the cloud data center and uses the cloud data center's supercomputing power to meet the application's computing needs in a centralized way. However, in the era of the Internet of Everything, the centralized processing model of classic cloud computing has three shortcomings:

- **Real–time requirements and network coverage**: As the number of edge devices increases, the amount of data generated by the devices continues to surge, causing network bandwidth to become the bottleneck of the classic cloud computing model gradually. The high cost of ensuring full–scenario network coverage also forces companies to decentralize part of their computing to the edge.

- **Data security and privacy**: With the popularization of data collection equipment, directly uploading the collected data to the cloud data center will increase the risk of leaking the core data assets of the enterprise. In addition, managing customer information on the cloud has also brought about personal privacy leakage. As a result, more and more companies are putting privacy–related data on edge for analysis.

- **Energy consumption**: As cloud servers run more and more applications, it won't be easy to meet the energy consumption requirements of large–scale data centers in the future. Improving the efficiency of energy consumption under the framework of the classic cloud computing model does not solve the fundamental problem by itself, and the issues in the era of the Internet of Everything will become more prominent.

In the edge computing model, the edge device has the processing capacity to perform calculations and data analysis. As a result, part or all of the computing tasks performed by the classic cloud computing model are migrated to the edge device, reducing the computing load of the cloud server, reducing the pressure on the network bandwidth, and improving data processing efficiency in the Internet of Everything era.

IoT–Oriented

Whether cloud–native or edge computing is essentially thinking about software architecture from a technical perspective. Cloud–native brings many benefits, but it also brings new challenges. Edge computing has solved some of these challenges and has become the second force in designing IoT software architecture. Now, it is time to return to the essential needs of the business and think about what software architecture the IoT system needs itself.

In much frequent communication with customers, EMQ found that whether it is on the cloud/edge/end, the software architecture designed for the IoT system is quite different from other web applications and embedded systems due to the following characteristics:

- Integrated system with hardware, software, sensors, connectors, and gateways.

- Various interoperable communication protocols.

- Support for data volume and velocity.

- Support for multiple QoS of communication.

- Live data stream analytics with complex event processing and analysis.

The traditional big data ecosystem cannot meet the above requirements. Therefore, IoT-oriented system architecture requires modern IoT data infrastructure software.

The Paradigm Shift in IoT Data Processing

	From	To
Data generation	human activities	connected things
Data form	transactional data	streaming data
Data volume	TB	PB/EB
Data processing	Batch processing	Stream processing
Data analytics	Schema on Write	Schema on Read

Data generation: from "human activities" to "connected things"

In the past, the generation of data mainly came from activities with people and businesses, such as individuals shopping on Amazon, customer relationship management for businesses.

Nowadays, more and more data is generated from connected IoT devices, such as smart hardware, smart home, industrial Internet, Internet of vehicles, and other scenarios.

Data form: from transactional data to streaming data

Traditional data essentially recorded events, an event with time, place, person, attribute, ID, etc., which is also known as transaction-based data. TP scenarios and aggregated AP scenarios dominate transactional data. TP scenarios emphasize transaction support and consistency, while aggregated AP scenarios are mainly processed in batches.

Data in the IoT is a record of the state of the physical world at each moment in time. The streaming data is generated continuously, and for each specific variable state, it is a series of values at different points in time. Therefore, the processing mode for IoT data is mainly streaming computing.

Data volume: from TB era to PB/EB era

Before the IoT era, data was generated primarily by human activity. In the last 20 years, the global population has grown from 6.3 billion to 7.6 billion, increasing 20%, with billions of mobile terminals. Most IT infrastructures handle data on a terabyte scale.

Data is primarily generated by devices and sensors, and the number of connected devices has increased 100-fold in the last decade from 50 million to 50 billion. Data infrastructure in the IoT era is facing petabytes or even EBs of data volume.

Data processing: from batch to stream processing

Over the past 20 years, databases have evolved from SQL to NoSQL to NewSQL. The BigData ecosystem has led data processing in batches, emphasizing the volume of data processed rather than the latency of data processing.

In the IoT era, data generated from devices and sensors is mainly messaging and streaming. With the release and adoption of the new generation of data infrastructure for streaming computing led by Flink, ksqlDB, and HStreamDB, stream processing is gradually becoming the dominant computing paradigm in data infrastructure.

Data analytics: from Schema–on–Write to Schema–on–Read

Traditional databases usually have a predefined data storage structure, such as a relational model. Then the data from the data source eventually reaches the database or data warehouse through ETL with strict schema.

Modern databases, MongoDB and Elastic, etc., and data warehouses usually store structured and semi–structured data in schema–free mode and apply the schema required to data when processing and analyzing.

Towards a Unified Ontology Namespace Architecture

In the longevity of IoT evolution, data is always the key to driving digital transformation. With the paradigm–shifting on both data volume and morphism, innovations occur in the infrastructure space, speeding up computation and increasing the velocity and volume of IoT data flow. New technology allows us to build more sophisticated architecture for processing large amounts of data and generating insights. However, it also introduces the overwhelming challenge of integrating all these technologies, both on compute side and storage side. Nonetheless, in previous IoT architecture, all data analysis applications require to extract data from storage. To preserve the context of information leads to an even worse problem: adding pieces upon each other and duplicating data. Therefore, you will usually end up in coupled architecture.

We can also elaborate this problem as data spaghetti, which causes complexity and is hard to maintain and scale. Especially when you need to introduce new streaming or storage technology, you will typically focus on merging multiple targets for the workflow to move data to where it should be serving. Therefore, new principles of solution architecture design arise to tackle such problems.

Move computation prior to data

Companies that execute digital transformation always find themselves needing to move data through different layers of traditional systems, such as MES, ERP, CRM, and WMS. Because simply collecting the data cannot produce value alone. We need to put them into the model for calculation and convert data into information. However, the cost of moving a huge volume of data physically is utterly unacceptable due to the limitation of connectivity and power. In contrast, carrying computation tasks to where the data is produced is wiser. Therefore, we can conclude that moving computation firstly, not the data, is one of the principles to build scale architecture.

Datastream reusability

When carrying out IoT solutions, the first step is by deploying sensor devices on the field to collect real–time data streams. The newly established system needs to connect to the previously deployed device's data stream as the business grows. It usually results in a discrete change, extra communication, and refactoring of old architecture.

To avoid extra networking traffic and development expenses, we do not assume how another system will consume the data. Because you never know where the future goes. Making datastream reusable enables the flexibility of data interoperation. Multiple scenarios could share the same data stream source. Then we can isolate the data stream by defining the corresponding namespace or topics.

Edge–cloud orchestration

Previous to Edge Computing, Cloud Computing arrived with many new capabilities, such as on–demand services and applications, scalability, efficient data storage, management, fault–tolerance, easy management. However, the evolution of technology also led to a fast rise in the amount of data generated, processed, and stored. As the number of network–connected devices continues to increase at an astonishing pace, the shortcomings of traditional network architectures, such as undesirable latency and connectivity, bandwidth limitation, the inability of context awareness, and where data is sent and received to/from the core of the network become visible.

To tackle the bottlenecks of traditional Cloud Computing networks, Edge Computing architectures emerged. The goal was to optimize the network and its applications by bringing computation closer to data usage, minimizing the need for long–distance communications between edge clients and servers.

The result is a reduction in latency and bandwidth usage, a significant improvement in connectivity, a more efficient network operation. Nevertheless, edge computing is not perfect in terms of service delivery, scalability, resource management, and maintenance. In most cases, if the user wants faster access to external computation and storage, porting the DevOps solution of the Cloud to the Edge platform is the only way. Dealing with a decentralized network means a heavier maintenance burden and requires more flexible edge–cloud collaboration.

Upon the background, EMQ defines the following principles.

1. The edge–cloud architecture must be decentralized and distributed, including Peer–to–Peer networking topology enabled.

2. Users only sense convenient and flexible computing capabilities like Cloud Computing, the distributed scheduling is its foundation, which is covered by the data infrastructure.

3. It is essential to support computation offloading on the Edge side. Edge service can converge data and identify and dispatch different loads(including communication and storage tasks) to appropriate nodes in the network.

Full–scenario adaptability

Heterogeneity is one of the key factors that hinder the booming of IoT applications. Different protocols and communication mediums are commonly used in different industries, most of which are irreplaceable such as Modbus and OPC UA. The rising number and diversity of edge devices imply an inevitable extra effort for building an open architecture of IoT. Interaction of heterogeneous systems and devices on disparate platforms requires unified data interfaces that facilitate information exchange between them and also act as the one single truth of data source. It is where a unified namespace shines. Unified namespace architecture is consists of a highly scalable MQTT broker being deployed in both edge and cloud, with a protocol gateway to connect heterogeneous devices. As a result, we can achieve full–scenario adaptability from an architecture perspective.

Unified Namespace – New Primitive for IoT Architecture

Upon these principles, we hereby bring up a new concept – unified namespace, which is an essential piece required for modern IoT architecture. In old terms, Unified namespace(UNS) often refers to the digital twin, but we broaden its notion. Despite abiding by the principles above, in short words, UNS is the center of IoT architecture where all data travel through. It is a data rendezvous point that you have complete control of. It is also an engine that pumps the data as a data flow to consumer services. It acts as the coordinators can provide analytics with the trajectory revealing the entire data lifecycle.

To better elaborate on the unified namespace and what could business benefit from the UNS concept:

Seamless data convergence

The data volume and transmission pace are experiencing rapid growth in the Internet of Everything era. Compared to the previous internet era, data is fragmented, connections are massive, and streaming analysis is necessary. They emanate from a broad range of applications areas with data being collected at unprecedented scale and speed. In modern IoE business, data sources generally consist of fragmented hardware, protocols, and various telecommunication modules. However, infrastructure software designed for traditional IT and Internet scenarios is very costly to adapt to the construction and maintenance of large concurrent connection systems. The unified namespace is the modern infrastructure software to resolve the data fragment and seamlessly converge data.

Under such circumstances, UNS is the heart of IoT architecture. The infrastructure software used as UNS must be not only capable of maintaining billions of connections but also ensure message delivery.

Data Model to Dataflow

After resolving the data convergence problem, we managed to collect the data to a namespace. But data alone cannot produce value. It needs to be structuralized and modeled comprehensively. Thus the other software can consume the dataflow seamlessly too. Sparkplug B and ISA-95 standards are the most commonly used primitive when defining the data model in the namespace. It is recommended to use the ISA-95 standard to define the MQTT topic. With a well-defined data model in your unified namespace, it can significantly reduce the integration task and the effort to enhance contextualize the data and add additional abstraction layers.

Furthermore, the traditional way of managing physical data integration via data lake and ETL is too costly to support IoT's dynamic business needs and data trends. The datastream virtualization methodology comes after the dataflow.

Data interoperability

There are already many IoT solutions in different industries. Nonetheless, it is rarely seen solutions take data interoperability into account. Interoperability is a characteristic of an IoT system, whose interfaces are completely understood, to work with other systems in the present or future, in either ingestion or access, without any restrictions.

The IoT devices vary a lot and may move geographically. They produce stateful streaming data. In most cases, IoT scenarios require these data to be processed in real-time. Depending on the conditions above, UNS must fulfill data interoperability as follows:

1. On edge, IoT devices should be able to autonomously communicate with other devices and connect data to the cloud.

2. On the cloud, IoT systems can integrate with multiple databases and applications easily.

3. In terms of data semantic, interoperability represents the capabilities of processing stateful IoT data and streaming analysis. Ideally, UNS provides rich plugins that convert streaming data to events, drive the AI model, and produce value.

4. Allow transformation between messaging protocols such as CoAP, MQTT, HTTP v.1.1, HTTP/2. UNS provides for IoT Interoperability among different protocols.

Exclusive Data source, Multiple user cases

Nowadays, data is the lifeblood of several businesses, Any organization that wants to utilize their data must be able to converge, identify and manipulate the datastream according to their needs proactively first. No matter if it is blockchain, IIoT or AI. These technologies share obliviously in common because they all depend upon each other for a data infrastructure that could provide an exclusive data source with multiple use cases. By applying a UNS architecture, All other system components only retrieve data that are required for its computation task from a unified data source, not make a copy and store them for each consumer. A centrally owned data namespace provides us with lightweight communication costs and enhances the transparency and visibility of datastream.

Ontology is the manifestation of a shared understanding of a domain agreed between several agents. Such agreement facilitates accurate and effective communications of meaning and data model, which leads to other benefits such as data interoperability, datastream virtualization, reusability and sharing. The ontology unified namespace is a place where everything goes; it is one single source of truth; it makes the original datastream reusable and relays it to wherever needs it. In this way, you create a data ecosystem for all other data consumers and scale your digital transformation solution for business.

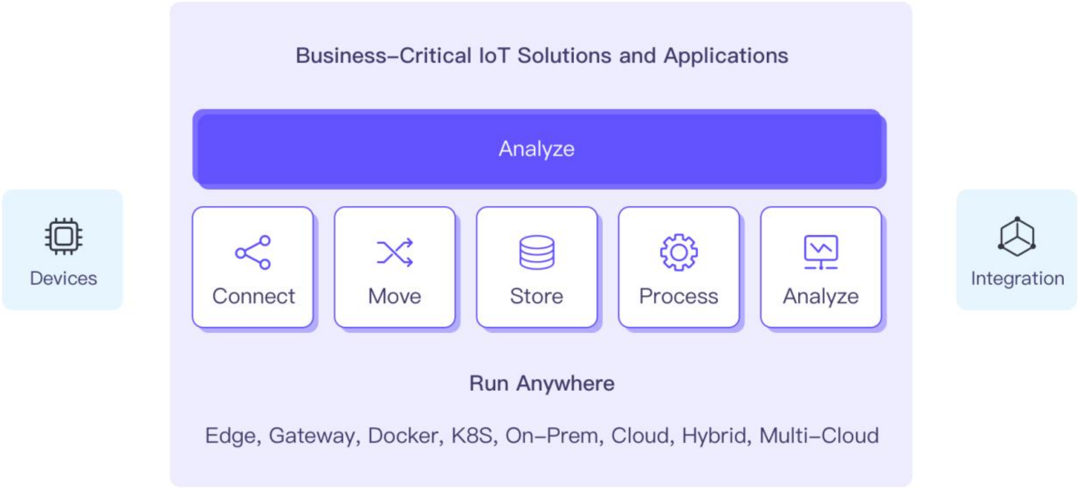
In a word, "Connect once & Integrate everything & Run anywhere & Analyze anytime" is the new philosophy of IoT architecture.

Modern Data Infrastructure for the IoT

Architecture Paradigm

To meet the paradigm shift in data processing and analysis in the IoT era and abide by the UNS concept, EMQ is proud to propose the "**Data Infrastructure for IoT**" architecture paradigm for future-proof IoT solutions.

The "**Data Infrastructure for IoT** "architecture paradigm achieves unified IoT data "connection, movement, storage, processing, and analysis" and integrates the IoT data with transactional data in the enterprise system to gain insights, leverage data, and create business value.


Business–Critical IoT Solutions and Applications
Analyze
Devices
Connect
Move
Store
Process
Analyze
Integration
Run Anywhere
Edge, Gateway, Docker, K8S, On–Prem, Cloud, Hybrid, Multi–Cloud

Connect

Massive–scale

The number of connected devices in the IoT scenario is much larger than traditional IT and Internet scenarios. The data infrastructure for IoT is required to provide a massively scalable **MQTT broker** able to connect hundreds of millions of devices reliably and efficiently.

Multi–protocol

For the diversity of IoT devices, the MQTT broker should have **multi–protocol** support to provide one–stop connectivity, including the open standard IoT protocols MQTT, CoAP, and LwM2M, and proprietary protocols.

Secure

Secure bi–directional communication is a mandate for IoT connectivity. The MQTT broker should support MQTT over TLS, or CoAP over DTLS, to encrypt IoT data and provide various authentication mechanisms using username/password, JWT, PSK, OAuth2, X.509 certificates, etc.

Reliable

In most cases, IoT devices may connect over unreliable networks or low–power networks such as NB–IoT. The MQTT broker should support stable connections, fast reconnections, and continuous, reliable data ingestion.

Move

Bidirectional pub/sub

Unlike enterprise or Internet applications in request-response mode, devices in IoT scenarios mostly communicate in publish-subscribe mode.

This data infrastructure can effectively transfer large amounts of telemetry data from IoT devices to cloud and reliably deliver control messages from cloud to devices via the open standard MQTT protocol.

Real-time message

At the same time, a high-performance real-time MQTT message processing engine is required to process millions of real-time IoT events between devices and cloud services.

End-to-end QoS

The MQTT broker in the data infrastructure should also support end-to-end QoS, including the "at least once" and "exactly once" to ensure IoT data delivery reliability.

Store

Stream-native

Stream is a great fit for IoT data, from the point of view of how IoT data is generated and how it exists. The stream itself is very lightweight and flexible, so not only can it meet the need of writing massive IoT data at high speed, but also it is very suitable for real-time analytics. Therefore, we have created the stream-native storage system in which stream is the basic unit for IoT data management. It can effectively support and manage a large number of streams with different granularity, and it also provides low-latency concurrent writes and high-throughput parallel consumption.

Unlimited Scale

Thanks to the cloud-native architecture design that separates storage and computing, the storage and computing layers can scale independently and elastically.

In the storage layer, we can scale in two dimensions.

On the on hand, it is horizontal scaling by adding more storage nodes. New storage nodes can be added and removed easily without affecting the existing system. The storage system can automatically balance the data distribution according to the change in the number of nodes.

On the other hand, vertical scaling is possible through automatic tiering mechanisms. Cold data can be automatically sunk to cheaper secondary storage, such as object storage. At the same time, the overall system provides a unified data access interface for real–time data and historical data.

Reliable and Consistent

With a powerful consensus engine, the data is reliably persisted to multiple copies while still maintaining strong consistency. We also provide data disaster recovery and reliability across regions, even globally.

Process

Real–time, low–latency data stream processing is at the core of data infrastructure for IoT, and the core components of processing include an SQL–based rules engine and a stream processing engine.

Rules Engine

A powerful rules engine is essential for extracting, filtering, enriching, and transforming stateless IoT data in real–time. Moreover, it accelerates the integration of IoT data with various middlewares, databases, cloud services, and enterprise systems.

Stream Processing

For stateful data, event–time–based stream processing is required for basic extraction, filtering, and transformation operations, as well as complex aggregation based on multiple time windows and even concatenation between multiple streams.

SQL–based Interface

And at the end, the data infrastructure should provide a friendly SQL–based interface to unify the development experience of the rules engine and stream processing engine.

Analyze

Real real–time

Nowadays, market opportunities are fleeting. Risk control and prevention must compete with every second. Business deciding requires speed and precision. All of these require the ability to turn raw data into actionable insights in real–time.

With efficient incremental materialized view maintenance technology, we can realize real real–time analytics. You don't need to run the same analytics task over and over again and tolerate the inevitable wait to get updated results. You just need to run it once, and the analysis results are automatically updated to reflect the latest data changes. This way, you always have the fastest access to real–time data insights.

Support Complex Analytics

You can express your analysis tasks by writing complex SQL statements, including multi–way joins, nested subqueries, and more. We translate these SQL statements into data flow graphs and execute them efficiently with the data flow engine.

Correct and Consistent

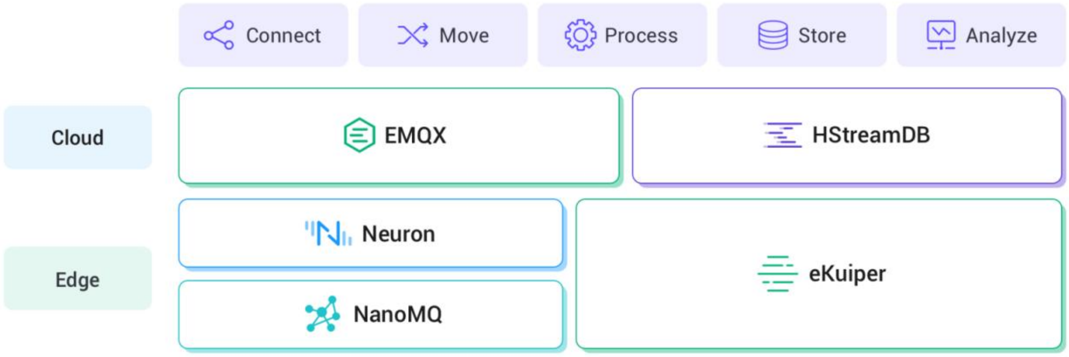
Real–time analytics must be performed with the assurance of correctness and consistency. Otherwise, its value is greatly diminished. Together with our storage system, we are able to provide end–to–end consistency assurance, so your business can rely on our analytics with confidence.

Integrate

A unified integration mechanism is critical to enable rapid application delivery and business innovation. The data infrastructure should ideally support low–code integration of IoT data with Kafka, RDS, various SQL/NoSQL/time–series databases, and enterprise systems such as Oracle and SAP.

EMQ's Portfolio for Data Infrastructure for IoT

EMQ has been working on several open–source projects based on the "Data Infrastructure for IoT" architecture paradigm in the past decade, including the world's leading MQTT messaging broker and stream processing database.



EMQX

Cloud – Connect | Move | Process

EMQX is the world’s leading Cloud–Native IoT Messaging Platform with an all–in–one distributed MQTT broker and SQL–based IoT rule engine, powering high–performance, reliable data movement, processing, and integration for business–critical IoT solutions.

The open-source EMQX broker, initially developed in 2012, is becoming the global De-Facto standard for distributed MQTT Broker.

Learn More

Github Project

HStreamDB

Cloud - Store | Process | Analyze

HStreamDB is a cloud-native streaming database explicitly designed for streaming data, supporting efficient storage and management of large-scale streaming data and complex real-time analytics on dynamically changing data streams. It helps enterprises gather instantaneous insights and stay competitive by empowering a quick implementation of real-time applications.

Learn More

Github Project

NanoMQ

Edge - Connect | Move | Transform

NanoMQ is a lightweight and blazing-fast MQTT 5.0 Broker for IoT edge platforms. It can efficiently connect IoT devices at the edge with standard MQTT protocols and transform and forward IoT data to the cloud.

Learn More

Github Project

Neuron

Edge - IIoT Connect | Ingest | Move

Neuron is an open-source industrial IoT gateway software connecting various industrial equipment with dozens of diverse industrial protocols and then forwarding the data with the standard MQTT protocol to IIoT platforms in the cloud. It requires ultra-low resource consumption and can be deployed on both X86 and ARM architectures.

Learn More

Github Project

eKuiper

Edge - Process | Analyze

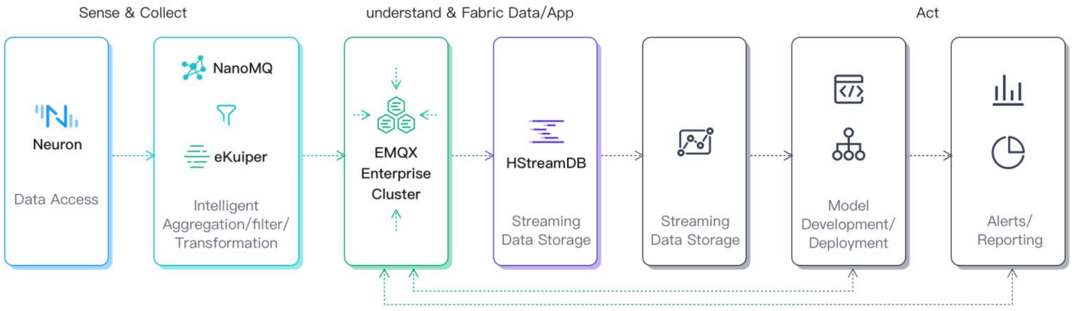
eKuiper is an edge lightweight IoT data streaming/analytics engine, and it runs on all kinds of resource-constrained edge devices. It migrates cloud real-time cloud streaming analytics frameworks such as Apache Spark, Apache Storm, and Apache Flink to the edge.

The eKuiper open source project, initially developed by EMQ, is now contributed to and maintained by the LF-Edge foundation.

Learn More

Github Project

Conclusion



Data is absolutely one of the most valuable assets of enterprises in the IoT era. For most enterprises, establishing their core competitiveness is all about the effectiveness of data use.

It is always EMQ's goal to bring out the greatest value of IoT data. We design and develop basic software based on the characteristics of IoT data and strive to transform the basic machine status and equipment perceived data into insight information with usage value through efficient and reliable connection, movement, processing, and analysis of data to realize data monetization and business innovation.

With the continuous generation of massive IoT data and further convergence with traditional enterprise TP data, more new data–centric and scene–oriented applications will gradually emerge. New business models will inevitably appear with the prosperity of artificial intelligence, cloud computing, and edge computing. EMQ "Data Infrastructure for IoT" will help build a solid IoT database and power the digital transformation of industry and society.

As an open–source software enterprise adhering to the principle of "service–oriented, customer first", EMQ is willing to work with global partners to jointly build "future–oriented" IoT business applications and new business models to serve the human industry and society.

EMQ

EMQ Technologies Co., Ltd.

The World's Leading Provider of Open Source IoT Data Infrastructure

www.emqx.com Email Twitter YouTube