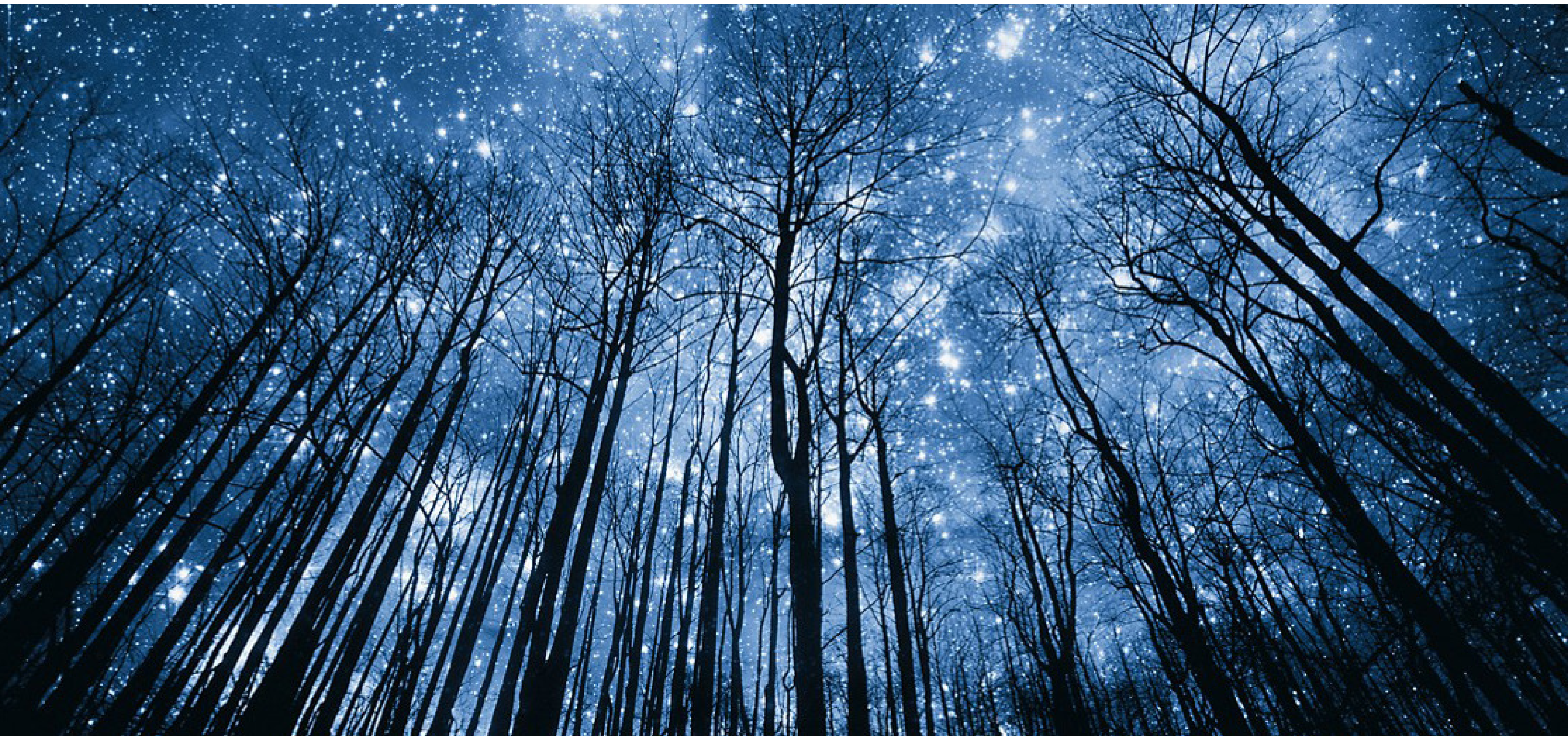
HOW EXPLORATORY DATA ANALYSIS FITS IN CLOUD AND EDGE ENVIRONMENTS



Poorna Prashanth
Manager, Inside Sales
Dell Technologies
Poorna.prashanth@dell.com

Knowledge Sharing Article

The Dell Technologies Proven Professional Certification program validates a wide range of skills and competencies across multiple technologies and products.

From Associate, entry-level courses to Expert-level, experience-based exams, all professionals in or looking to begin a career in IT benefit from industry-leading training and certification paths from one of the world's most trusted technology partners.

Proven Professional certifications include:

- Cloud
- Converged/Hyperconverged Infrastructure
- Data Protection
- Data Science
- Networking
- Security
- Servers
- Storage
- Enterprise Architect

Courses are offered to meet different learning styles and schedules, including self-paced On Demand, remote-based Virtual Instructor-Led and in-person Classrooms.

Whether you are an experienced IT professional or just getting started, Dell Technologies Proven Professional certifications are designed to clearly signal proficiency to colleagues and employers.

Learn more at www.dell.com/certification

Table of Contents

Introduction

High speed, throughput, resiliency, and scalability – Digital leaders in today's world have mastered these capabilities to be more competitive, and they were able to achieve this by adopting microservices and event driven architecture. **Event Driven Architecture** (EDA) is an approach to build and develop applications that asynchronously communicate or integrate with other applications and systems via **events**.

An event may be any change in state defined by the application, which is used to establish communication between applications and systems. A good example of an event is the addition of a new banking feature on a banking website. An application designated as a producer detects the event and sends out relevant data as a message. The intermediary, an even processor, directs the message to an application designated as an end user, who receives the message. Multiple relevant end users can receive the same message and process the associated data in their own way.

Audience

This paper is for Dell Technologies sales and presales field personnel interested in understanding the components of EDS and its benefits on a high level. This article provides insights on the relevance of EDA in Edge and Cloud Environment. This should equip sales teams to be able to drive initial conversation on Edge and Cloud solution associated with decoupled microservices

Scope

This paper intends to cater to the audience with more awareness about EDA while capturing below information,

1) Fundamental knowledge about EDA
2) EDA Architecture
3) Benefits of EDA
4) Used Cases:
    a. EDA for Edge Environment
    b. EDA for Cloud Environment

Overall, this article should give a high-level insight into modern application architecture.

What is Event Driven Architecture?

Today’s world relies on revolutionary technologies like micro services and cloud native development that make distributed systems and applications possible, where a system or application is built from components physically located in separate places. For the distributed components to work seamlessly there should be seamless communication between each component

Traditional synchronous architecture cannot handle the performance and scale needs of distributed systems and consequently applications, face challenges in distributed environment with latency issues, system degradation, unpredictable failures.

EDA helps developers overcome these challenges. EDA requires minimal coupling between services, while we can still communicate with each other, which makes Event Driven Architecture apt for modern and distributed applications. EDA is an ideal architecture for enterprise applications that derive the most benefit from scalable and reliable real-time communication such as data integration, website activity tracking and stream processing.

EDA creates an environment where there is no waiting for requests. Events are identified when there is a change in the state that demands a response and that is when the response is executed. Events occur when resources are not available to respond. EDA stores such event notifications until resources become available

EDA is not a software language but is an approach to programming and a model for an efficient solution to capture, communicate, process the persistence of events. Event drive applications can be created in any programming language

EDA Maturity Level

There are 5 levels of EDA Maturity

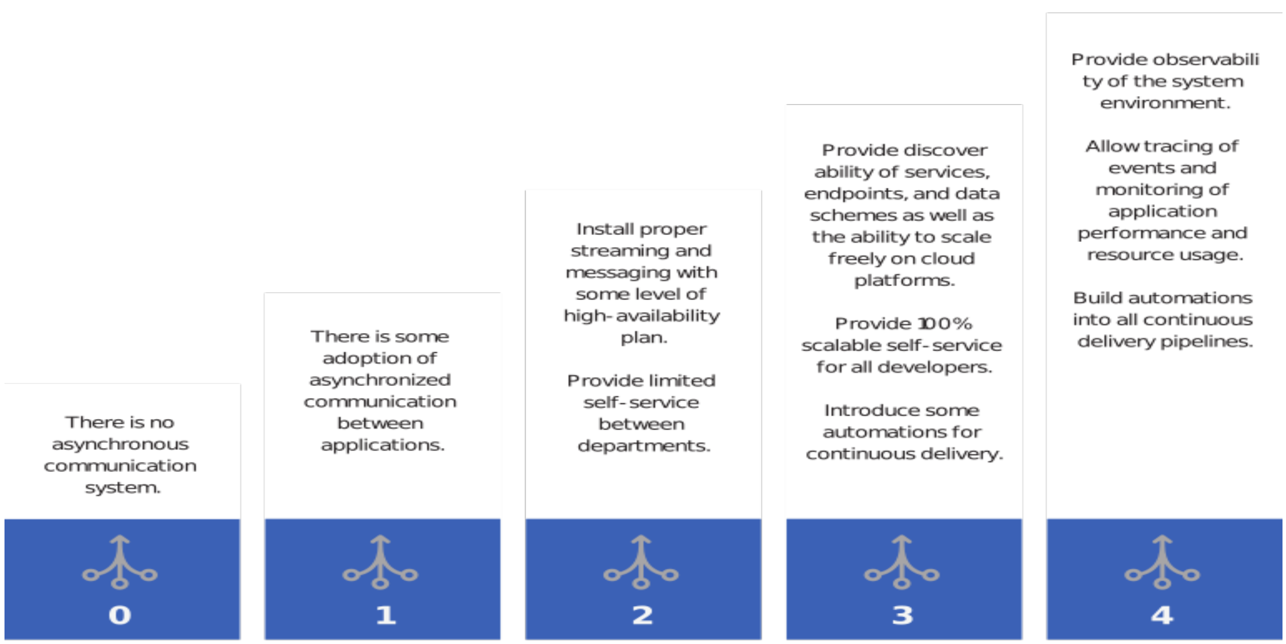
Level 0: Organizations with no applications and systems that use any type of asynchronous communication

Level 1: Organization begins to modularize services and adopt asynchronous calls between each application. Data operation teams provide infrastructure and common application services for application teams

Level 2: Messaging and Streaming solutions deployed between applications and services and high availability is planned as part of the configurations.

Level 3: EDA provides scalable self-service both on premises and in the cloud for resilient and quick delivery. Endpoints and services with data scheme description eliminate silos and achieve better communications

Level 4: Has well defined asynchronous communication between systems. EDA provides observability and events and message monitoring throughout the environment. Automation is built as part of the continuous delivery pipeline



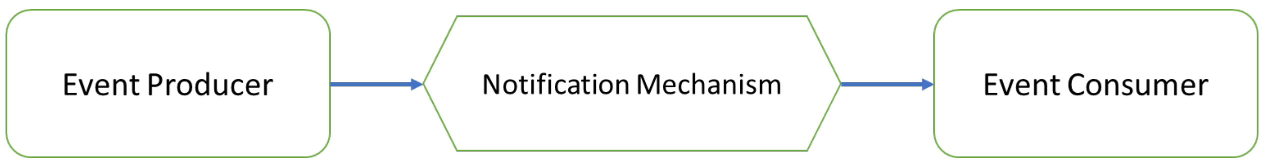
Picture 1: EDA Maturity level, Source: Red Hat

Source: https://www.redhat.com/en/resources/event-driven-architecture-hybrid-cloud-blueprint-detail

Components of Event Driven Architecture

The basic attributes of an event include source, header, metadata, key, payload, and time. An event is a significant and meaningful change of state of any data in any object. The data is often referred to as event payload. The object need not be aware of any other components of EDA

EDA minimally constitutes of 3 basic components:
Event Producer
Event Notification Mechanism



Picture 2: High Level EDA Architecture

Event Producer

Event Producer: Any event is generated from a source. The source might be a data store, application, business process, transmitter, sensor, or a tool. An ordinary event may be evaluated by an event preprocessor (router, filter) resulting in generation of a new notable event. As we can have variety of event generators, not all events will be generated in specific format to enable event processing. Hence the event needs to be transformed to the required formant (enterprise standard) prior to having deposited in event channel

Event Notification

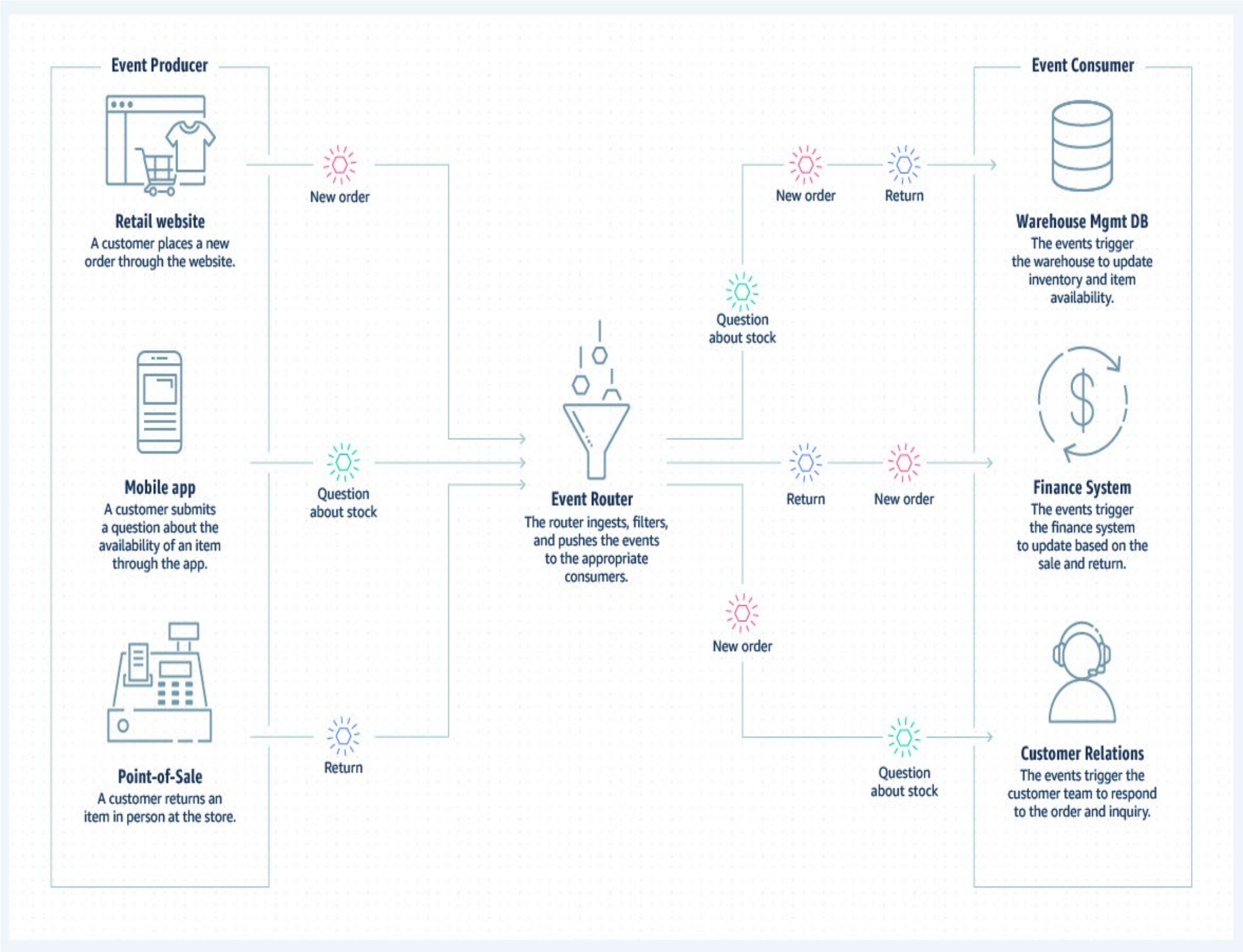
An event happens when there is change in state for any system hardware or software. The system sends a message to tell other parts of the system. The event notification is a result of the event. The event notification has two parts – Event Header and Event Body

Event Header fetches name for the event, timestamp, and type of event
Event Body provide additional details about the event
Event router routes information to different components of the system through defined channels

Event Consumer

Event initiates variety of activities such as commencing a service or business process. Event consumers include warehouses, automated agents, applications, active business processes, humans are dashboards. Event consumers may choose from wide variety of responses to every event or store the event for later processing

Here is an example of event drive architecture for an e-commerce site. The architecture enables the site to react to changes from various sources during peak demand, without over-provisioning resources or crashing the application



Picture 3: Overview EDA Architecture Example
Source: https://aws.amazon.com/what-is/eda/

Technologies used in EDA

There is no specific development framework for creating EDA based apps. Instead, developers integrate solutions from different technologies. Here are few technologies that are primarily used for developing EDA based app

<u>For Event Capturing:</u>

Dell Boomi, MuleSoft, Dataflow, Snaplogic, Apache Apex

<u>For Routing Events:</u>

Apache Kafka, ActiveMQ, RabbitMQ

<u>For Event Processing:</u>

Azure Stream Analytics, Amazon Kinesis, Apache Storm, Apache Flink

Advantages of the Event-Driven Architecture

Key Benefits of Event-Driven Architecture are:

Scales easily to provide highly-responsive reactions to events – EDA scales easily and provides highly-responsive reactions to all manner of applications and access to data in real time to enable fast and effective decision making

Enables Asynchronous functionality – All function in EDA is completely asynchronous enabling easier addition of new events and process easier. Roll back changes or moving to specific event are much easier. This makes the architecture versatile and easily replaceable

Big Data Scalability – EDA enables highly divergent, loosely couple's system to identify events happening in other systems that require a response. This enables automation that is highly responsive with substantial efficiency increases in delivering the right information that facilities faster decision making. EDA systems also facilitates rapid access to large volumes of data generated by IoT (Internet Of Things) devices and stored across various connected networks

Develop with agility – There is no longer needed to write custom code to poll, filter and route events. The event router will automatically push events to the consumer. The router removes the requirement of co-ordination between producer and consumer services

Cut costs – Event-driven architectures are push based, so everything happens on demand as the event presents itself in the router. This means less network bandwidth requirement, less CPU (Central Processing Unit) utilization thereby cutting cost

When to leverage EDA

Integration of heterogeneous systems – EDA can be leveraged to share information between different systems without coupling when system is running different stacks

Resource state monitoring and alerting – EDA can be leveraged to monitor and receive alerts on anomalies, changes and updated instead of constantly monitoring our resources

Cross account, cross region data replication - EDA can be used to coordinate systems between teams operating and deployed across different regions and accounts. By using Event router to transfer data between systems, services can be developed, deployed, and can scale independently

Fanout parallel processing – If there are lot of systems that need to operate in response to an event, EDA can be used to fanout the event without having to write custom code to push to each consumer. The router will push event to the system

Use Cases

EDA for Cloud Environment

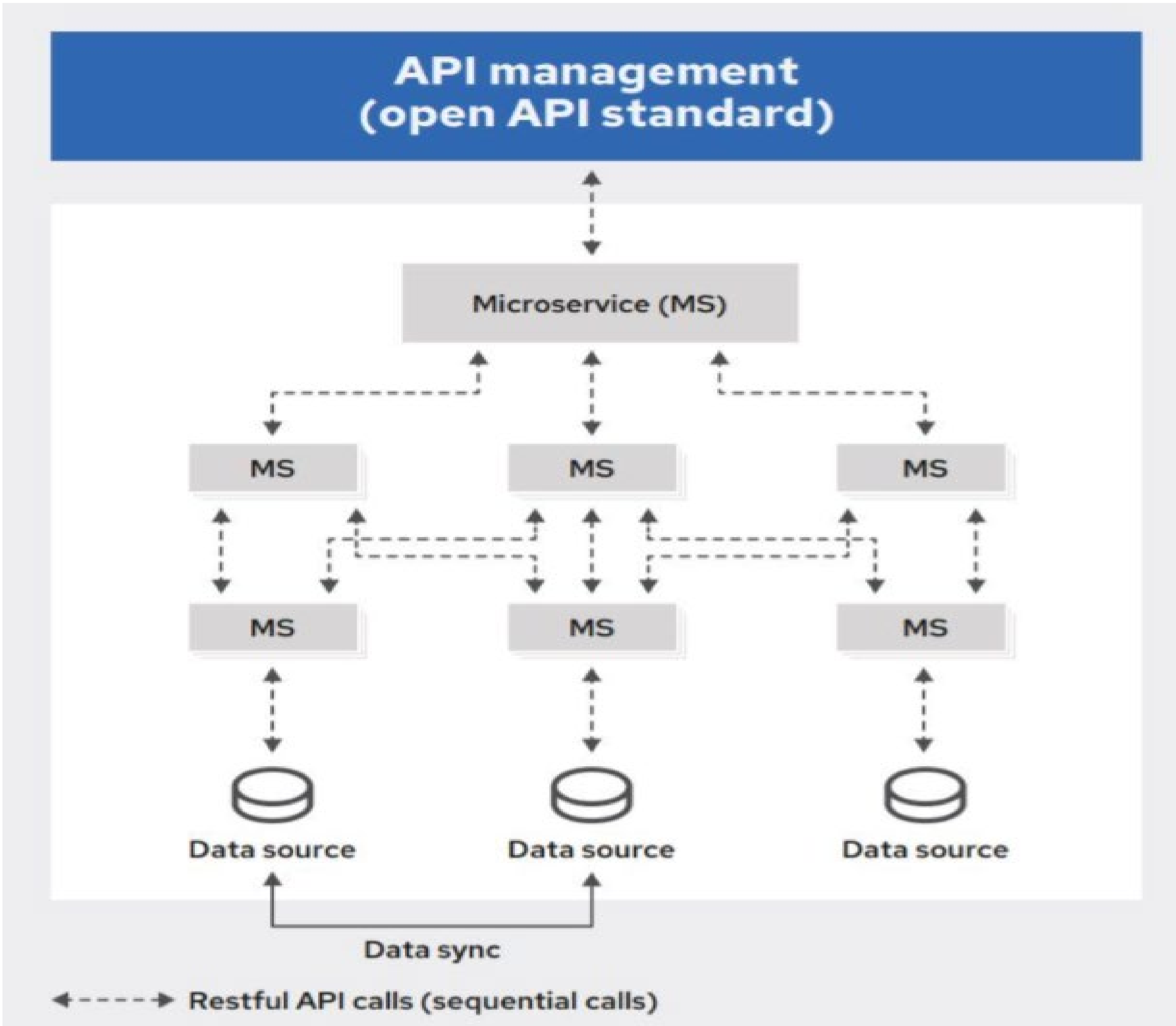
Organizations are rapidly adopting cloud computing to gain technical and business feature benefits. As part of this shift, application architectures are evolving to a model of distributed, modular, and portable components that can be easily deployed and run across cloud infrastructure, both on premises and on public or private clouds. The hybrid cloud approach minimizes infrastructure costs and increases the operational efficiency of applications. In this environment, strategic use of compute resources and effective coordination between application components are critical architecture design goals

EDA is ideal for hybrid cloud applications, enables Multicloud spanning across different geographies. It is better suited for real world use cases, where decoupling of protocols and runtime are required to achieve fine grained scaling. Cloud and Kubernetes container distributed systems are best suited for EDA architecture

Synchronous application programming interfaces (APIs) dominate the cloud native environment. Communication through HTTP/S is simple to implement and makes it easy to manage and trace a particular request. But there are limitations in areas like circuit breaking to isolate failures, multicasting requests and decoupling services. EDA address these limitations because they provide asynchronous communication approaches for effective fault tolerance, metrics in a highly distributed microservices architectures.

Synchronously communicating system has defined structural standard to set up communication between subsystems. The flow is mostly in sequential order, enabling easier call tracing.

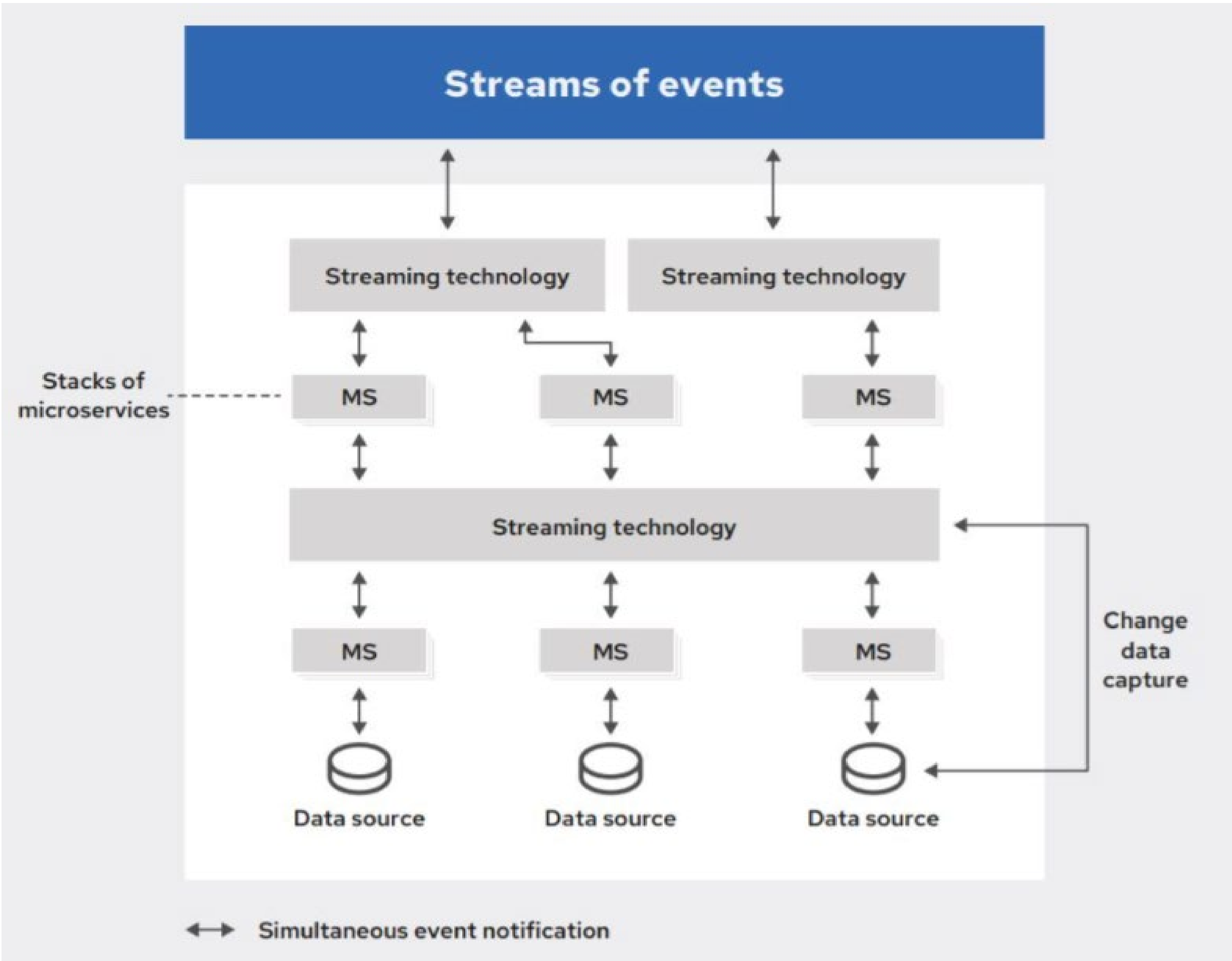
Picture 4 is a preview of logical architecture of communicating system



Picture 4: Overview of typical Synchronous System
Source: https://www.redhat.com/en/resources/event-driven-architecture-hybrid-cloud-blueprint-detail

In Asynchronous system, events are broadcasted to applications and each application responds to these events. Applications tend to be more modular with no dependency on any specific system. This architecture enables us to handle larger volumes of steaming events and leverage event sourcing to handle transactions.

Picture 5 is a preview of logical architecture of communicating system



Picture 5: Overview of typical Asynchronous system
Source: https://www.redhat.com/en/resources/event-driven-architecture-hybrid-cloud-blueprint-detail

EDA is based on asynchronous, non-blocking communication and can release resource usage instead of waiting for the response to get back. This is significantly relevant for cloud and container native development which demands scalability, flexibility, and agility from decoupled microservice environment. EDA enhances decoupling from the communication standpoint, as multiple subscribers can receive events simultaneously. This results the system with lower latency and higher throughput

Advantages of EDA in Cloud Environment

Organizations can optimize cloud resources consumption, drawing below benefits from EDA

Deployment across hybrid cloud: Components of EDAs can consume and generate events across cloud boundaries, taking advantage of individual cloud capabilities while remaining complaint with policy

Parallel Processing: On-linear calls allow faster data processing and can trigger a wide range of service and application processes waiting for events. The ability to scale can optimize resource usage
High observability and extensibility: Implementing systems in terms of events they produce makes it easier to observe the behavior of the system and extend it to meet future requirements without re-architecture

Cloud native nature: EDAs are designed to work perfectly in the distributed world because cloud native services are running separately on servers/nodes at separate locations in the cloud

Fault Tolerance: EDA emphasizes failure isolation and natural circuit breaking without the requirement of a separate implementation. The component decoupling stops failures from propagating to the rest of the system or application

EDA for Edge Environment

Computing needs to happen where it's most beneficial to the business. The operating model has moved from centralized to distributed as to where the employees operate from. As remote working becomes the norm, companies are eager to extract more insights when and where they happen, edge computing has gone from nice to have to necessity

In today's IT environment, what happens if there is no effortless way to integrate edge data with all available databases? How can we operate in multiple fragmented, inconsistent environments?

Some of these questions can be warranted when paired with event-driven architecture, enabling Edge to offer new opportunities.

When combined EDA and Edge have enormous opportunities to put consistent, scalable, and real time computing into parts of a company that was otherwise inaccessible. If an edge device goes offline, it can continue queueing up events to send as soon as it's back online, giving companies unprecedented confidence and flexibility unleashing news streams of data and insights.

There are 3 significant areas where EDA and Edge Computing overlap

Enterprise – Core Data Store is used to extend an application to edge computing resources

Operations – Typically involve industrial applications and the need to process data on site, which often comes in the form of on-machine IOT (Internet Of Things) assets that trigger events to begin predictive maintenance processes before an asset's performance degrades

Provider – A network, when combined with EDA lead to multi-access edge computing center adapting to consumers bandwidth hungry applications in remote locations

Advantages of EDA in Edge Environment

Building a robust event-driven architecture for edge environment led to seamless process and operations

Integration

As moving parts of EDA are event data transferring between producers and consumers, integration is often more straightforward than managing complex payloads in a hybrid environment.

Management and Maintenance at the edge

IT staff can't always stay on the edge. Sometimes they can't even get there without a harness. With EDA we can manage every part of the infrastructure including edge devices that need to operate independently and without connectivity at times. Kubernetes and Red Hat Quay simplify these challenges with automation, CI/CD pipelines, encryption for security conscious edge environments

Lock-in

EDA for Edge computing means using a variety of vendors to provide complete coverage from the first event to the end results in applications or data stores. A common solution is using solution or services with a foundation in open source. Pairing the open-source community's emphasis on interoperability with technical backing from a major cloud provider often equals a flexible architecture with swappable components that can change often.

EDA in the real world

EDA is an ideal architecture for enterprise applications that derive the most benefit from reliable and scalable communication, such as data integration, stream processing and website activity tracking.

Data Integration

EDA is suited for developing data integration applications responsible for capturing, aggregating, and processing data from multiple diverse sources and presenting it in a single unified view. Data integration applications provide significant support for decision makers with easily accessible, accurate insights. Data integration can be used to process streams of events for real time monitoring, alerting and analysis while concurrently sending same data to another system for retention and historical analysis.

Few examples of data integration built upon EDA are:

Retail dashboard – Delivers real time trends to decision makers while saving the data to enterprise data lakes for later analysis. EDA can enable application to process billions of events every day.

Application to track customer interactions – EDA enables the application to process data concurrently for usage by both internal systems and customer facing web and mobile applications. EDA fetches significant customer insights into the business while providing customers with timely information about their accounts and transactions.

Stream Processing

EDA enables developers to build applications designed to react to process or transform multiple streams of events in real time. A stream is a continuous flow of events sent from a producer to multiple producers. The advantages of EDA for stream processing applications include scalability to handle copious amounts of streaming data and the reliability to ensure open communication lines.

Few examples of stream processing with EDA are:

IoT Monitoring relying on Machine Learning microservice to process events in a timely manner to enable proactive repair and replacement of mission critical devices

Financial fraud detection - Discovery, identification, and remediation of intentional misrepresentation of financial data. Determining financial fraud requires near zero latency between the actual events. EDA enables shared, reliable, low latency event distribution so all stakeholders have access to timely and consistent data

Website activity tracking

EDA ca be helpful to developers while building applications that track user activity on websites. This type of application gives a company visibility to both website performance and customer activity. For the business this can provide insight into the customer's interest and potentially trigger automated suggestions for additional products and services based on the customers' requirements

Microservices and applications at cloud and edge have become more complex primarily because of synchronous communication between components. Due to inherent limitations, events and asynchronous communication are the ideal alternative
Fast, secure, and efficient EDA cloud solutions – that's what it takes to accelerate cloud journey.

Conclusion

Event Driven Architecture and its fitment in Edge and Cloud environment will give you an overview of basic operating principals and components of EDA and how organizations can benefit by EDA equipping sales teams to drive initial conversation on decoupled microservices at edge and cloud

Citations:

https://aws.amazon.com/event-driven-architecture/
https://www.redhat.com/en/resources/event-driven-architecture-hybrid-cloud-blueprint-detail