AGILE DATA PROCESSING PIPELINES USING MICROSERVICES AND MAPR
# Tables of Contents

## EXECUTIVE SUMMARY

3

## THE CASE FOR MICROSERVICES: HOW THEY WORK AND HOW THEY FOSTER AGILITY

4

## MICROSERVICES ON MAPR

6

### INTRODUCTION TO THE MAPR CONVERGED DATA PLATFORM

6

### MAPR-ES: A MESSAGING PLATFORM FOR MICROSERVICES

8

### MAPR-DB: THE HIGH-PERFORMANCE NOSQL DATABASE

8

### MAPR-XD: THE NEXT-GENERATION CLOUD-SCALE DATA STORE

9

### WHY IS MAPR THE BEST PLATFORM FOR DEVELOPING MICROSERVICES?

9

- **Converged Data Services**
  9
  - **Unified Storage for All Data Types**
    10
  - **Out-of-the-Box High Availability for All Data Services**
    10
  - **Data Protection From Hardware Failures and User Error**
    11
  - **Record-Setting High Performance**
    11
  - **Global Data Replication**
    11
  - **Horizontal Scalability**
    12
  - **Parallelized, In-Place Analytics**
    12
  - **Unified Security**
    12

## CONCLUSION

13
EXECUTIVE SUMMARY

We describe in this paper the mechanics of a data processing pipeline, implemented as a series of microservices and powered by a next-generation converged data platform. Microservices hold great promise for any organization that wants to deploy applications more quickly, run multiple versions of analytics in parallel, or upgrade systems with less downtime or risk. This new, more modular pattern of development has been gaining in popularity, and its impact has been described in the following way:

Over the last decade, there has been a strong movement toward a flexible style of building large systems that has lately been called microservices. This trend started earliest at innovative companies such as Google, and many aspects of microservices have since been reinvented at a variety of other companies. Now, among highly successful, fast-evolving companies that include Amazon, LinkedIn, and Netflix, the microservices approach has become more the rule than the exception, at least partly because companies who adopt this style of architecture move faster and compete better.

The microservices idea is simple: larger systems and data pipelines should be built by decomposing their functions into relatively simple, single-purpose services that communicate via lightweight and simple techniques. Such services can be built and maintained by small, very efficient teams.1

Although simple in concept, the reality is that deploying any piece of critical IT infrastructure within an enterprise can be a complex undertaking, and architecting a reliable and scalable microservices-driven data pipeline is no exception. Traditional approaches involve stitching together multiple subsystems, each of which has to be engineered separately for data resiliency, high availability, cross-data center replication, access control, and more. Greater complexity leads to higher costs, delayed implementations, challenging staffing and training requirements, and, ultimately, poses greater risks to mission success.

In contrast, MapR offers a greatly simplified picture: a single, horizontally scalable converged data platform that delivers world record performance across a breadth of data services, providing much of the functionality needed for a world-class data processing pipeline. By choosing the right data platform, customers can now implement microservices-based data processing pipelines with less complexity, less cost, faster performance, and greater interoperability than alternative architectures allow.

1 From Chapter 3 of the ebook Streaming Architecture: New Designs Using Apache Kafka and MapR Streams, by Ted Dunning and Ellen Friedman, which can be downloaded from the following URL: mapr.com/streaming-architecture-using-apache-kafka-mapr-streams.
THE CASE FOR MICROSERVICES: HOW THEY WORK AND HOW THEY FOSTER AGILITY

Microservices are simple, single-purpose applications or system components that work in unison via a lightweight communication mechanism. That communication mechanism is very frequently a publish/subscribe messaging system, which has become a core enabling technology behind microservice architectures and a key reason for their rising popularity. A central principle of publish/subscribe systems is decoupled communications, wherein producers don’t know who subscribes, and consumers don’t know who publishes; this system makes it easy to add new listeners or new publishers without disrupting existing processes.

One such messaging system is MapR Event Streams (MapR-ES). MapR-ES will be discussed in more detail below, but at a high level, MapR-ES allows any number of information producers (potentially millions of them) to publish information to a specified topic. MapR-ES will reliably persist those messages and make them accessible to any number of subscribers (again, potentially millions). MapR-ES can scale to very high throughput levels, easily delivering millions of messages per second using very modest hardware.

When you combine these messaging capabilities with the simple concept of microservices, organizations find that they can greatly enhance the agility with which they build, deploy, and maintain complex data pipelines. Pipelines are constructed by simply chaining together multiple microservices, each of which listens for the arrival of some data, performs its designated task, and optionally publishes its own messages back to a topic (in the same stream or to a different stream). The newly published data might be enhanced or transformed versions of the original data, or it might be new data altogether (an alert, for example, or an aggregated metric, or a message that finalized data is ready for broader consumption). The entire process is illustrated below in Figure 1. Note that in addition to pushing and pulling data to and from the data stream, each microservice oftentimes needs to interact with other system components (reading data from a file system, for example, or updating records in a database).

Figure 1. A data pipeline constructed as a series of microservices
It’s worth emphasizing that multiple microservices might be listening for the same piece of data, completely unaware of each other’s presence. The messaging platform ensures that all relevant messages are delivered to each subscriber for processing, which therefore occurs in parallel. As an example of how this process becomes useful in practice, development teams can deploy service upgrades more frequently and with less (sometimes zero) risk, because the vetted production version of the microservice does not need to be taken offline. Both versions of the service simply run in parallel, consuming new data as it arrives and producing multiple versions of output. Both output streams can be monitored over time; the older version can be decommissioned when it ceases to be useful.

Also note that when data is consumed, it does not have to disappear from the stream. Messages, once published, are immutable, and can be retained forever. New subscribers of information can replay the data stream, specifying a starting point as far back as the data retention policy enables. This continuity is a significant departure from legacy message queues, which stipulated that subscribers had to be online in order to receive data. Under this new paradigm, teams can develop a new analytic, deploy it, and then process every message that was ever received, in the proper sequence, at whatever pace the new analytic service can sustain. Assuming it can process data faster than the rate at which new data arrives, eventually it will process all available historical data, at which point new messages will be processed in real time.

### MICROSERVICE ARCHITECTURES ENABLE ENTERPRISES TO OPERATE WITH GREATER SPEED AND AGILITY

Teams are reorganized around a specific microservice, whereas before they were often divided between the tiers in an application. Cross-functional teams focusing on specific and well-defined system functions often see boosts in productivity and higher quality software.

Pipelines are created by chaining microservices together, loosely coupled through a publish/subscribe messaging system. The loose coupling reduces interdependencies between services, allowing them to be modified with more autonomy.

Newly received or newly processed data can be “pushed” to any number of interested parties, instantly.

New capabilities can be integrated into the pipeline by simply subscribing to the relevant data stream. No existing components have to be modified, and no process orchestration logic has to be rewired.

New versions of services can run in parallel with existing versions, greatly reducing the risk of system upgrades and greatly increasing the pace of innovation.

Data streams can be replayed, providing an easy mechanism to bring newly deployed services “up to date” with the existing state of data. Faulty services can be decommissioned, corrected, redeployed, and resynchronized.
MICROSERVICES ON MAPR

INTRODUCTION TO THE MAPR CONVERGED DATA PLATFORM

The MapR Converged Data Platform integrates global event streaming, real-time database capabilities, and scalable enterprise storage with a collection of data processing and analytical engines to power a new generation of data processing pipelines. The integration of these capabilities into a single platform is illustrated below in Figure 2.

MapR software is installed on a single cluster of standard servers and includes extremely powerful capabilities, spanning a cloud-scale data store (MapR-XD\(^2\)), No-SQL database (MapR-DB), and messaging system (MapR-ES). From an architectural perspective, these are all integral capabilities of one converged data platform and can be deployed on a single cluster; this capacity differentiates MapR from any other offering on the market. These capabilities are optimized to utilize system resources efficiently (data caching mechanisms, for example) and can scale horizontally by adding nodes (i.e., servers) to the cluster. With the addition of each new node, the system increases its storage capacity (disks\(^3\)), processing power (CPUs), and RAM. Clusters can scale to thousands of machines.

---

\(^{2}\) The ‘XD’ in MapR-XD denotes: eXtreme scale, every (X) data type, eXtreme speed, and every (X) infrastructure.

\(^{3}\) MapR does not utilize Storage Area Networks (SANs) or Network-Attached Storage (NAS). By utilizing the disk drives provided within each commodity server, MapR achieves horizontally scalable storage with better performance at lower cost than competing storage technologies. Additionally, MapR cluster nodes provide compute resources as well as storage capacity, with linear scalability.
From a functional perspective, these capabilities are administered under a single umbrella and share consistent features, such as data protection [e.g., snapshots and mirrors], security [e.g., access control expressions], and multi-tenancy [e.g., data placement constraints and size quotas]. MapR clusters can be set up in the cloud or on-premises. Data can be replicated and synchronized across multiple MapR clusters around the world, whether that data exists as files, database tables, or event streams.

Data services are accessed using industry-standard open APIs, providing familiar interfaces to system administrators and application developers while avoiding vendor lock-in. These APIs are provided in Table 1.

<table>
<thead>
<tr>
<th>SERVICE</th>
<th>API</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAPR-XD (CLOUD-SCALE DATA STORE)</td>
<td>NFS, POSIX, REST, HDFS</td>
</tr>
<tr>
<td>MAPR-DB (OPERATIONAL DATABASE)</td>
<td>HBase, OJAI</td>
</tr>
<tr>
<td>MAPR-ES (EVENT DATA STREAMS)</td>
<td>Kafka</td>
</tr>
<tr>
<td>ANALYTICAL &amp; ML ENGINES (SPARK, DRILL, HADOOP, AND MORE)</td>
<td>SQL*, Spark, Apache Hadoop</td>
</tr>
</tbody>
</table>

Table 1. Open APIs provide familiar interfaces and avoid vendor lock-in

Unlike competing technologies, MapR does not rely on the local Linux file system to manage data on disk. MapR has been built from the ground up to be the next-generation platform for big data and implements a patented system of distributed data containers. MapR uses this single underlying data architecture to support all core data services–files, tables, and streams. Many enterprise features (data distribution, replication, snapshots, mirrors, quotas, and more) are implemented at the container level, ensuring consistently high performance and reliability across the platform.
MAPR-ES: A MESSAGING PLATFORM FOR MICROSERVICES

MapR-ES is a global publish/subscribe event streaming system that connects data producers and consumers worldwide in real time, with unlimited scale. MapR-ES forms the foundation of a microservices-based data pipeline, ensuring that outputs of one processing stage are reliably delivered to the next. MapR scales by automatically spreading data and processing across all nodes in the cluster. Consumers will automatically load-balance across partitions, enabling applications to scale linearly with increasing data rates.

Developers access MapR-ES using the Kafka API; there is no dedicated API for MapR-ES. Unlike Apache Kafka, however, MapR-ES is part of a converged platform that can provide multiple data services (messaging, database, file system) as well as analytics on a single, horizontally scalable cluster. Data is automatically distributed across all available hardware, ensuring that administrators need not worry about individual servers reaching capacity, only the overall capacity of the cluster. MapR-ES also supports complex, global replication topologies and hybrid cloud deployments. Producers and consumers can automatically fail over from an on-premises cluster to a cloud cluster (or vice versa), helping organizations realize the full potential of the cloud as a mechanism for providing system resiliency.

MAPR-DB: THE HIGH PERFORMANCE NOSQL DATABASE

Each microservice is essentially a stand-alone mini-application, which often requires a back-end database to support its operations. MapR-DB is an enterprise-grade, high performance NoSQL database that excels at fast data insertion and even faster recall—exactly what is needed for data pipelines, where each stage in the process typically retrieves the most recently received data, operates on it in some way, and publishes the processed result back into the system for broader consumption or the next stage in the process. Based on Apache HBase™ but re-engineered to take advantage of the underlying MapR data architecture, MapR-DB scales to millions of tables and trillions of rows, delivering extremely high performance results and suffering none of the reliability issues and latency spikes with which HBase has been associated. (HBase is built on top of the read-only HDFS file system, which presents insurmountable challenges in production environments.) MapR-DB supports a key-value data model, like HBase, but also adds native JSON support, which is a huge benefit for the large community of developers that has embraced JSON for its flexibility and ease of use. Database operations can be implemented with a single line of code.

In contrast to traditional relational databases, MapR-DB is “schema-less,” which means that arbitrary attributes can be added as needed, allowing for sparse data sets and evolving schemas. Data types can be tagged with an essentially unlimited number of attributes, and unique or infrequently used attributes can be selectively applied to targeted subsets. This flexibility becomes extremely useful when “A/B testing” new system features: each version of a service can add its own unique attributes without affecting any other aspects of the system. Apache Drill, which also provides native JSON support, allows developers to query against these new attributes instantly, without any prior registration of the new schema elements.

An excellent demonstration of the scalability, performance, and reliability of MapR-DB is Aadhaar. Aadhaar is the largest biometric database in the world, used to authenticate over a billion people across the entire country of India for a wide variety of services. Biometric templates for all enrolled citizens are stored in MapR-DB, which is queried for all authentication requests. Some key system metrics are as follows:

- Over one billion citizens enrolled
- One million new enrollments added daily
- Millions of requests per day, with response times under 200ms
MAPR-XD: THE NEXT-GENERATION CLOUD-SCALE DATA STORE

The assumption behind messaging systems is that the messages themselves are not very large. Throughput suffers as the message size increases, so large files are not embedded in line with the message. The common practice is to store the larger data elsewhere (an external file system, for example), specifying the file location as a message attribute. This is typically true of NoSQL databases as well.

MapR-XD is the high performance, read/write, distributed file system built into the MapR Converged Data Platform. It is POSIX-compliant and provides transparent read/write file access via NFS, allowing system components and applications to mount the MapR cluster and seamlessly interact with it as if it were direct-attached storage. It includes many important features for production deployments, such as advanced data replication, access control, and transparent data compression at virtually unlimited scale.

MapR-XD can also be used to provide persistent file storage for Docker containers. Docker provides a useful deployment mechanism for microservices and is therefore frequently seen in microservice architectures. But Docker containers are ephemeral in nature; all data is lost when a container is shut down. Mounting MapR-XD to a Docker volume provides data persistence beyond the life of the container.

WHY IS MAPR THE BEST PLATFORM FOR DEVELOPING MICROSERVICES?

Converged Data Services

Revisiting the data pipeline illustration in Figure 1, we see that many of the key architectural components required for building microservices are included as core data services of the MapR Converged Data Platform (see Figure 4). Developers have immediate access to a suite of best-in-class technologies—a messaging system, a NoSQL database, and a file system—all of which scale to any size data or Service Level Agreement (SLA). The time-consuming tasks of evaluating, procuring, installing, integrating, administering, securing, and maintaining multiple point solutions are all avoided. This immediately lowers costs and reduces time-to-implement, even at the early prototype stage.

For each service, no additional engineering or administrative effort is required to scale the capability, make it highly available, make it resilient against data loss, or implement the many other features required of a mission critical system. Each data service inherits this critical feature set directly from the platform, which frees developers to do what they want to do: code new features that deliver value to the enterprise. The use of popular, open APIs decreases the learning curve, as developers will already be familiar with many of the interfaces. And native JSON support across the platform further increases productivity, allowing data to be accessed and queried quickly and easily, often with just a few lines of code.
Unified Storage for All Data Types

With MapR-XD, MapR-DB, and MapR-ES, MapR provides unified data storage within a global namespace that can accommodate all data types, including files, documents, analytical data sets, application tables, and event data streams. Whatever the data type, the MapR Converged Data Platform provides uniform mechanisms for distributing the data across the cluster, protecting against data loss, and replicating data across remote data centers. This uniformity translates to superior system reliability and greatly simplified system administration across the full range of services.

Data is automatically compressed as it comes into the system, and on-disk data structures are optimized for files, JSON data models, and streams. Customers can have full faith and confidence in MapR as a system of record for files, tables, and streams.

Out-of-the-Box High Availability for All Data Services

Data processing pipelines are almost always mission-critical to an organization and carry the associated requirements for high availability. MapR provides built-in high availability for file system, database, and messaging system services, requiring no additional hardware and no additional configuration or maintenance. Services are distributed across multiple nodes of the cluster, providing seamless failover in the event of unexpected hardware and process failures. The MapR architecture contains no single points of failure.

MapR is known for high reliability in production environments. MapR customers have been known to run for multiple years in heavily used production environments with zero downtime. Rolling upgrades are supported, such that core system software can be upgraded without loss of service. This reliability was one of many reasons Aadhaar chose MapR as the back-end database for authentication services, now considered to be critical infrastructure for the country of India. If MapR goes down, India goes down, and we won’t let that happen.
Data Protection from Hardware Failures and User Error

Data written to MapR is automatically replicated to protect against failures of disk drives, servers, and even entire racks of equipment. All copies are automatically dispersed across the cluster, algorithmically placed for optimal protection and automatic load balancing. Recovery from failures is automatic. This entire process is completely transparent and applies to all data types: files, tables, and streams. System administrators need only worry about overall system storage capacity, an easily monitored metric.

To protect against user and application errors, MapR provides true point-in-time snapshots. Snapshots can be initiated manually or scheduled to occur at regular time intervals—a simple administrative function that can be configured in minutes. When a snapshot is created, the existing data within MapR is captured instantly, with zero storage overhead. Snapshots reliably capture the existing state of files, tables, and streams.

Record-Setting High Performance

As a data management system, MapR is fast, efficient, and cost-effective. As demonstrated in publicly available benchmarks, MapR has set multiple world performance records in file, database, and streaming scenarios. Customer experience supports these claims: independently validated surveys show that 77% of MapR customers cite performance as a key factor in their selection of MapR, and MapR enjoys a 99% customer retention rate.

Global Data Replication

MapR can replicate and synchronize data across globally distributed clusters, both in the cloud and on-premises. Move only the data you need by specifying the appropriate files, tables, and/or event streams (which are organized by message topic). Data transfer is secure and extremely efficient: data changes are tracked at the block level (8k in size) and only new or modified blocks are transferred. MapR features converged, easily configurable replication technology, providing reliable worldwide transport of all data, including files, tables, and event streams. Replication across all data types can be configured in minutes, and once configured it requires very little administrative attention (“set and forget”).

Table replication can be active-active; all sites can be producing information, simultaneously sending and receiving updates to and from multiple remote locations. MapR-ES replicates not only the message data but also the consumer offset positions (i.e., which consumers have consumed which messages), such that client failover across multiple locations is seamless. For the file system, MapR provides mirrors, which automatically replicate all file operations (create, delete, and update) to one or more remote locations.

MapR supports complex, globally distributed topologies. A single source can replicate to up to 64 destinations and receive data from up to 64 sources. Replication chaining is supported, and MapR provides built-in loop detection so that data is never replicated to a remote cluster more than once.
Horizontal Scalability

MapR scales linearly, such that doubling the nodes in the cluster doubles the storage and processing capacity of the system. From a pipeline processing perspective, simply add nodes to do any of the following:

- Incorporate more inbound data feeds
- Store larger data files
- Increase the number or sophistication of pipeline processing stages
- Provide more analytics and reporting

Adding nodes to the system is a simple administrative process. Once registered with the cluster, MapR automatically redistributes data to the new nodes and redirects service requests accordingly.

Parallelized, In-Place Analytics

Once the data lands in MapR, it can be immediately processed, in place and in parallel, by any of the open-source processing frameworks packaged with the MapR distribution and deployed to the cluster. Common choices include Apache Drill for SQL-based analytics and Apache Spark for more complex programmatic analyses. MapR supports operational applications and analytical processing on the same cluster.

Data can be normalized, aggregated, and transformed as needed. Output data sets (in the form of analytical results, aggregated data, or normalized, enriched data sets) can be written back to MapR-XD, or they can be fed into MapR-DB to support real-time applications. Output data (alerts, aggregates, or individual data records, for example) can be published to MapR-ES as a series of messages, identified by topic, and MapR-ES will reliably deliver those messages to all system components that have registered as a subscriber to that information.

Apache Drill is an open source distributed SQL processing engine that can query petabytes of data. It talks natively to MapR-XD and MapR-DB, allowing you to combine data from both sources within a single query. Drill supports JSON natively, allowing data that arrives in JSON format to be queried instantly, without any schema registration required. Drill will discover the schema on the fly. New attributes are discovered automatically and are immediately available for query. Drill provides ODBC and JDBC interfaces, allowing access to MapR data (files, database tables, JSON documents, and streams) from a large number of third-party business intelligence (BI) and reporting tools, such as Tableau, MicroStrategy, and many others.

Unified Security

The MapR Converged Data Platform provides converged security: consistent access control mechanisms across files, tables, and streams. MapR can limit data access across all data services to specific groups and users as specified, using MapR Access Control Expressions (ACEs). System auditing is also comprehensive. All data access can be logged and queried, regardless of data type. The audit logs themselves are written directly to MapR-XD in JSON format, which can be queried natively using Apache Drill.

MapR is a multi-tenant environment. Individual tenants can be restricted to specific work areas (called MapR volumes) to which administrative policies can be applied that control access, restrict disk space consumption, and protect data, according to pre-defined schedules. Data and processing can be restricted to specific nodes of the cluster, further ensuring that the actions of one tenant do not adversely impact other tenants hosted within the same MapR environment.
CONCLUSION

Data pipelines are the lifeblood of an organization and critical to its success. The ability of an organization to accomplish its mission depends, to a very great extent, on its ability to ingest relevant data, move that data through multiple stages of processing and enrichment, deliver high-quality information to analysts as soon as it becomes available, and continually cycle new insights and developments back across the enterprise to all interested subscribers.

The traditional approaches to constructing these pipelines, while successful in their day and foundational to world-class enterprise capabilities, now pose a great threat to mission success by hampering an organization’s ability to innovate rapidly. System enhancements have become too difficult to deliver within operational environments, largely because complex system interdependencies increase the risk of failure.

In contrast, industry has adopted a new approach that allows companies to deliver incremental capabilities on a continuing basis, without subjecting their customers to delays or downtime that would put their brand loyalty at risk. This new approach relies on a microservice architecture, powered by an underlying messaging system that reliably moves data from publishers to consumers. This approach enables smaller, focused teams to deliver innovation at the fastest speed they can achieve. The impact to downstream systems is minimized or completely eliminated by the parallel operation of multiple versions of a service.

MapR has built a world-class data services tier that is highly reliable, scales to global deployments, and delivers some of the fastest performance characteristics achievable. At any scale, MapR efficiently and reliably manages the full range of essential data types and exposes a wide suite of services and capabilities for building modern, next-generation data pipelines. MapR delivers these capabilities on a single converged platform, allowing system-level enterprise requirements (data distribution, replication, security, and more) to be provided in a uniform fashion, leading to superior system performance and stability at a lower cost while greatly reducing administrative overhead.

To learn more about how MapR can help you implement more powerful and flexible data processing pipelines using microservices, please visit the MapR website (mapr.com/appblueprint/architectures/) or contact your local MapR sales and technical representatives, as identified on the cover page of this paper. Free training is also available at training.mapr.com.