ARCHITECTING FOR FAST DATA APPLICATIONS

Building the infrastructure and data services to deliver real-time customer interactions at scale.

MESOSPHERE

INTRODUCTION

In today's always-connected economy, businesses need to provide real-time services to customers that utilize vast amounts of data. Examples abound—real-time decision-making in finance and insurance, to enabling the connected home, to powering autonomous cars. While innovators such as Twitter, Uber and Netflix were at the forefront of creating personalized, real-time services for their customers, companies of all shapes and sizes in industries including telecom, financial services, healthcare, retail, and many more now need to respond or face risk of disruption.

To serve customers at scale and process and store the huge amount of data they produce and consume, successful businesses are changing how they build applications. Modern enterprise applications are shifting from monolithic architectures to cloud native architectures: distributed systems of microservices, containers, and data services. Modern applications built on cloud native platform services are always-on, scalable, and efficient, while taking advantage of huge volumes of real-time data.

However, building and maintaining the infrastructure and platform services (for example, container orchestration, databases and analytics engines) for these modern distributed applications is complex and time-consuming. For immediate access, many companies leverage cloud-based technologies, but risk lock-in as they build their applications on a specific cloud provider's APIs.

This eBook details the vital shift from big data to fast data, describes the changing requirements for applications utilizing real-time data, and presents a reference architecture for fast data infrastructure.

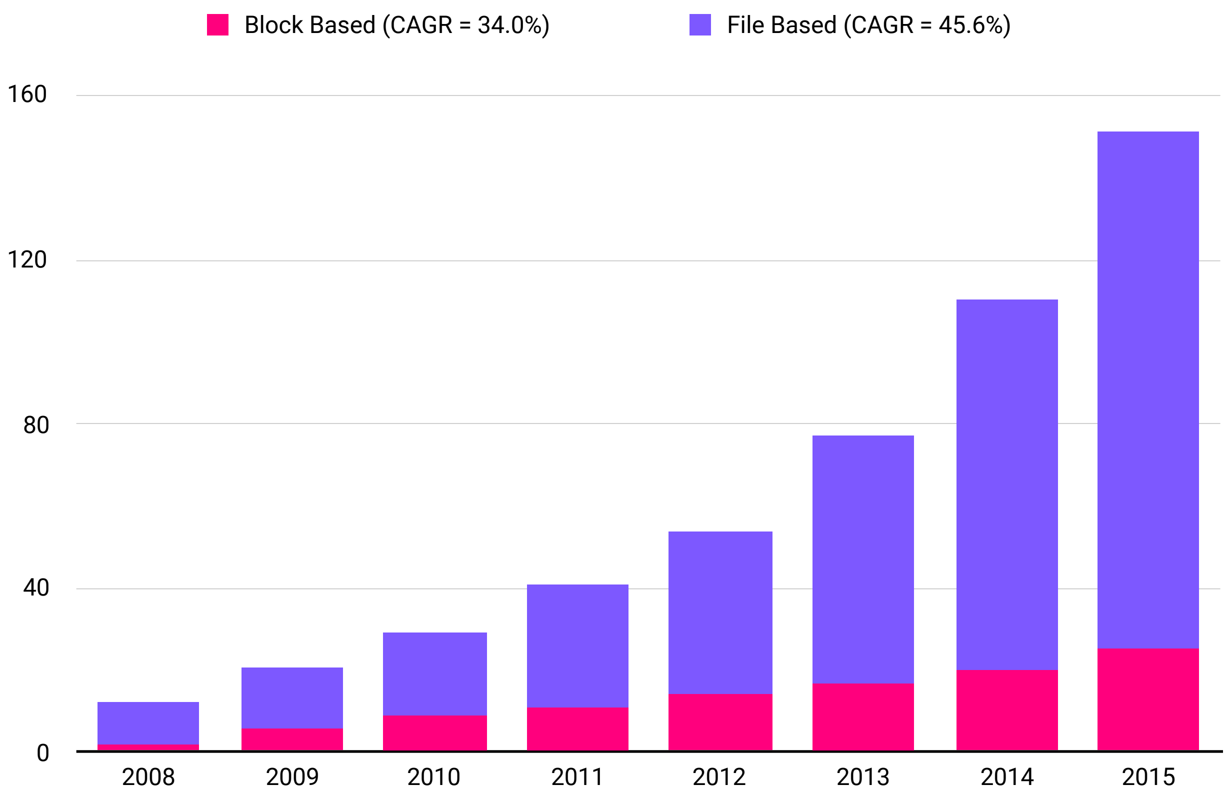
“FAST DATA”: THE NEW “BIG DATA”

Data is growing at a rate faster than ever before. Every day, 2.5 quintillion bytes of data are created[1] - equivalent to more than 8 iPads per person.[2] The average American household has 13 connected devices, and enterprise data is growing by 40% annually. While the volume of data is massive, the benefits of this data will be lost if the information is not processed and acted on quickly enough.

One of the key drivers of the sheer increase in the volume of data is the growth of unstructured data, which now makes up approximately 80% of enterprise data. Structured data is information, usually text files, displayed in titled columns and rows which can easily be analyzed. Historically, structured data was the norm because of limited processing capability, inadequate memory and high costs of storage. In contrast, unstructured data has no identifiable internal structure; examples include emails, video, audio and social media. Unstructured data has skyrocketed due to the increased availability of storage and the number of complex data sources.

[1] Vouchercloud Big Data infographic

[2] Based on 32 GB iPad



Worldwide File-Based Versus Block-Based Storage Capacity Shipments, 2008-2015

Source: IDC Worldwide File-Based Storage 2011-2015 Forecast, December 2011

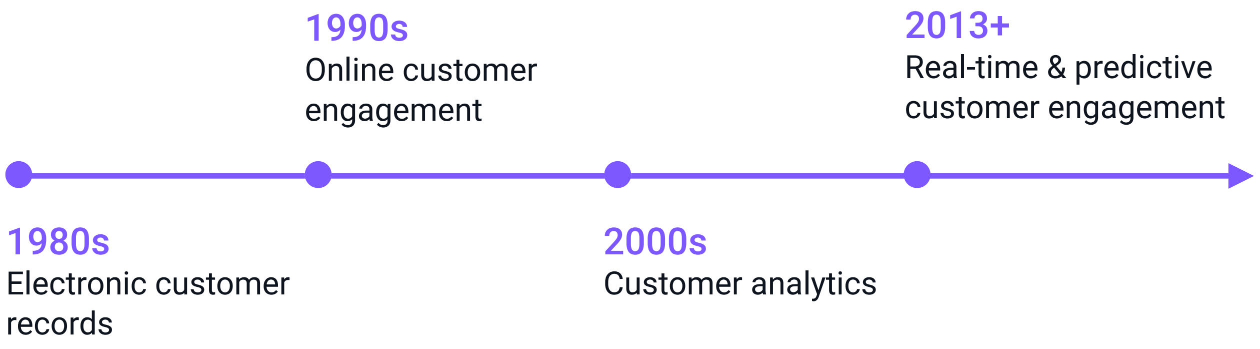
The term “big data” was popularized in the early- to mid-2000s, when many companies started to focus on obtaining business insights from the vast amounts of data being generated. Hadoop was created in 2006 to handle the explosion of data from the web.

While most large enterprises have put forth efforts to build data warehouses, the challenge is in seeing real business impact—organizations leave the vast amount of unstructured data unused. Despite substantial hype and reported successes for early adopters, over half of the respondents to a Gartner survey reported no plans to invest in Hadoop as of 2015.[3] The key big data adoption inhibitors include:

[3] Survey Analysis: Hadoop Adoption Drivers and Challenges, Gartner, May 2015

1. Skills gaps (57% of respondents): Large, distributed systems are complex, and most companies do not want to staff an entire team on a Hadoop distribution.

2. Unclear how to get value from Hadoop (49% of respondents): Most companies have heard they need Hadoop, but cannot always think of applications for it.


1980s
Electronic customer records
1990s
Online customer engagement
2000s
Customer analytics
2013+
Real-time & predictive customer engagement

Industry Transitions

Over the past two to three years, companies have started transitioning from *big* data, where analytics are processed after-the-fact in batch mode, to *fast* data, where data analysis is done in real-time to provide immediate insights. For example, in the past, retail stores such as Macy's analyzed historical purchases by store to determine which products to add to stores in the next year. In comparison, Amazon drives personalized recommendations based on hundreds of individual characteristics about you, including what products you viewed in the last five minutes.

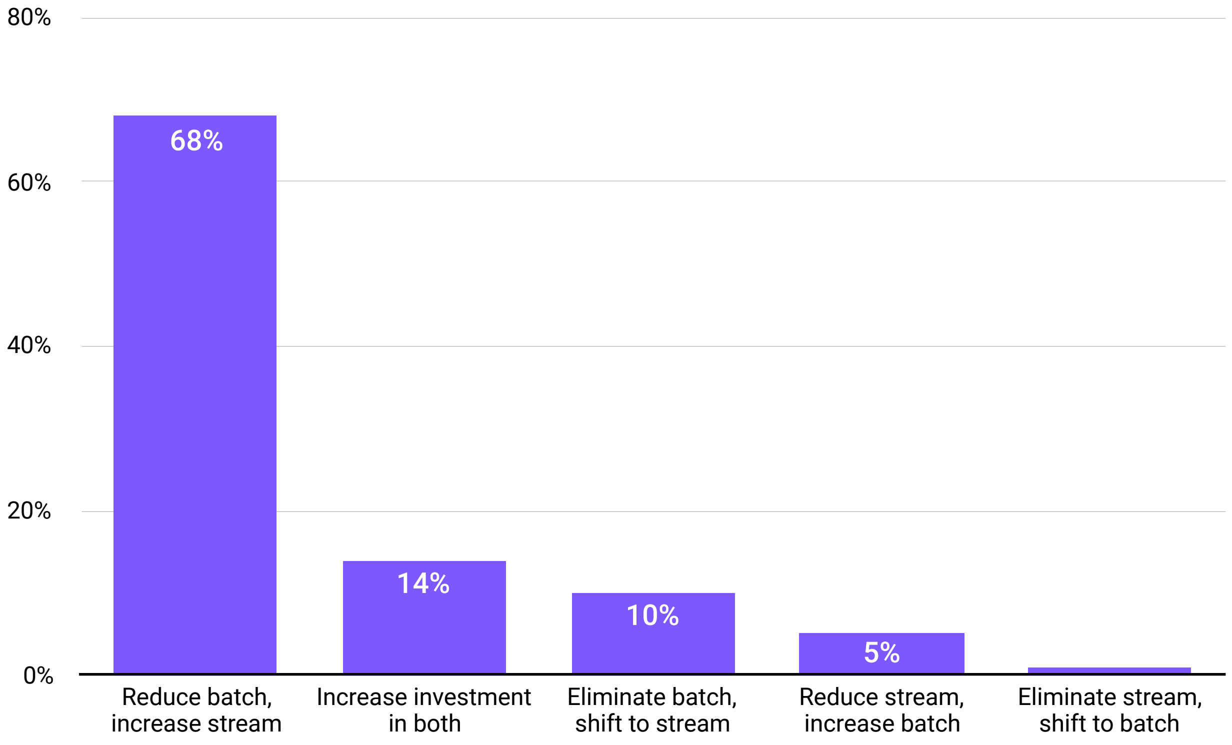
Big data is collected from many sources in real-time, but is processed after collection in batches to provide information about the past. The benefits of data are lost if real-time streaming data is dumped into a database because of the inability to act on data as it is collected.

Modern applications need to respond to events happening now, to provide insights in real time. To do this they use fast data, which is processed as it is collected to provide real-time insights. Whereas big data provided insights into user segmentation and seasonal trending using descriptive (what happened) and predictive analytics (what will likely happen), fast data allows for real-time recommendations and alerting using prescriptive analytics (what should you do about it).

VERTICAL	BIG DATA	FAST DATA
Automotive	Automakers analyze large sets of crash and car-based sensor data to improve safety features	Connected cars provide real-time traffic information and alerts for predictive maintenance.
Healthcare	Doctors provide care suggestions based on historical analysis of large datasets	Doctors provide insightful care recommendations based on predictive models and in-the-moment patient data.
Retail	Stores determine which products to stock based on analysis of previous quarter's purchase data.	Online retailers provide personalized recommendations based on hundreds of individual characteristics, including products you viewed in last five minutes.
Financial Services	Credit card companies create models for credit risk based on demographic data.	Credit card companies alert customers of potential fraud in real-time.
Manufacturing	Manufacturing plants improve efficiency based on throughput analysis.	Manufacturing plants detect product quality issues before they even occur.

Big Data Vs. Fast Data Examples

Businesses are realizing they can leverage multiple streams of real-time data to make in-the-moment decisions. But more importantly, fast data powers business critical applications, allowing companies to create new business opportunities and serve their customers in new ways. Over 92% of companies plan to increase their investment in streaming data in the next year[4], and those who don't face risk of disruption.



How Will Usage of Batch and Streaming Shift in Your Company in the Next One Year?

Source: 2016 State of Fast Data and Streaming Applications, OpsClarity

[4] OpsClarity Fast Data Survey, 2016

FAST DATA APPLICATIONS IN ACTION

GE is an example of an organization that is already using fast data both to improve existing revenue streams and create new ones. GE is harnessing the data generated by its equipment to improve performance and customer experience through preventive maintenance. Additional benefits include reduced unplanned downtime, increased productivity, lowered fuel costs, and reduced emissions. The platform will also be able to offer new services such as remote monitoring and customer behavior analysis that will represent new revenue streams.[5]

UBER

Uber's ride sharing service depends on fast data—the ability to take a request from anywhere in the world, map that request to available drivers, calculate the route cost, and link all that information back to the customer. This requirement may seem simple, but it is actually a complex problem to solve—responding within just a few seconds is necessary in order to differentiate Uber from the wider market. Uber is also using their fast data platform to generate new revenue streams, including food delivery.

[5] Big & Fast Data, CapGemini, 2015

Capital One

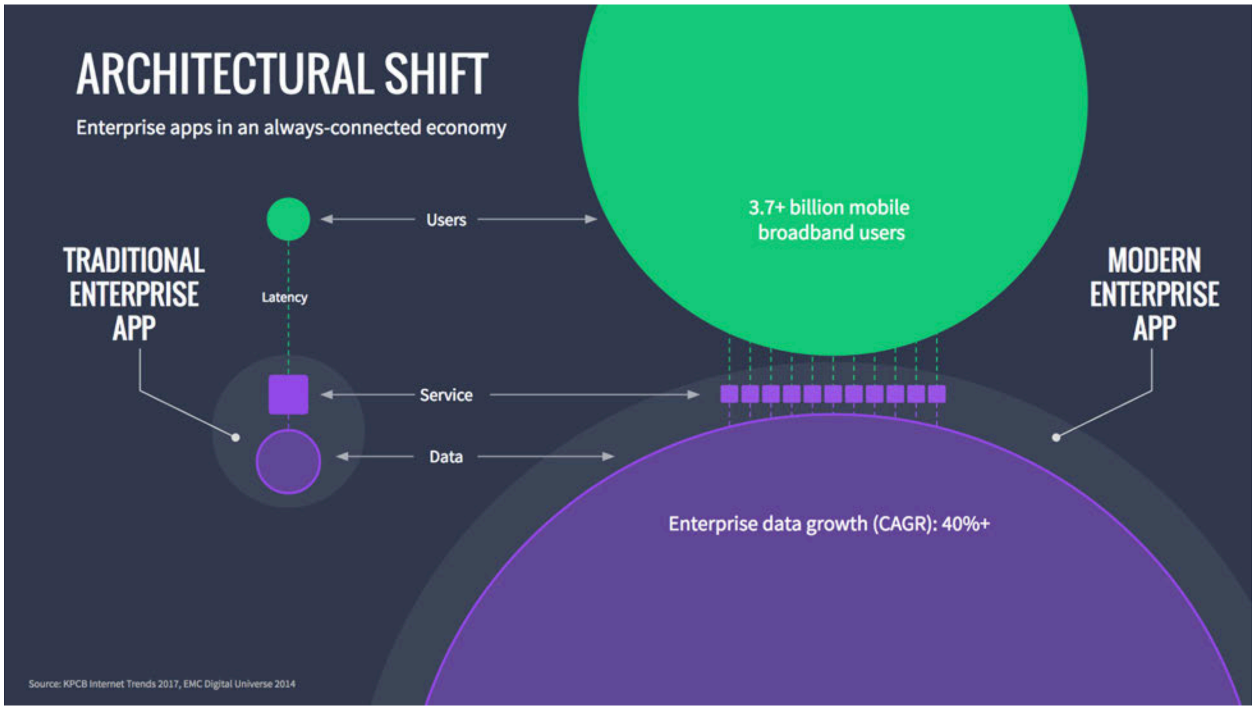
At Capital One, analytics are not just used for pricing and fraud detection, but also for predictive sales, driving customer retention, and reducing the cost of customer acquisition. Machine learning algorithms play a critical role at Capital One. “Every time a Capital One card gets swiped, we capture that data and are running modeling on it,” Capital One data scientist Brendan Herger says. The results of the fast data analytics have made their way into new offerings, such as the Mobile Deals app that sends coupon offers to customers based on their spending habits. It has also enabled predictive capabilities in the call center, which CapGemini says can determine the topic of a customer’s call within 100 milliseconds with 70 percent accuracy.[6]

[6] How Credit Card Companies Are Evolving with Big Data, Datanami, May 2016

A REFERENCE ARCHITECTURE FOR FAST DATA APPLICATIONS

Real-time data-rich applications are inherently complicated. Data is constantly in motion, moving through distributed systems including message queues, stream processors, and distributed databases.

To handle massive amounts of data in real-time, successful businesses are changing how they build applications. This shift primarily entails moving from monolithic architectures to distributed systems composed of microservices deployed in containers, and platform services such as message queues, distributed databases, and analytics engines.


ARCHITECTURAL SHIFT
Enterprise apps in an always-connected economy
TRADITIONAL ENTERPRISE APP
Users
Latency
Service
Data
3.7+ billion mobile broadband users
MODERN ENTERPRISE APP
Enterprise data growth (CAGR): 40%+
Source: KPCB Internet Trends 2017, EMC Digital Universe 2014

Architectural Shift From Monolithic Architectures to Distributed Systems

The key reasons enterprises are moving to a distributed computing approach include:

1. The large volume of data created today cannot be processed on any single computer. The data pipeline needs to scale as data volumes increase in size and complexity.
2. Having a potential single point of failure is unacceptable in an era when decisions are made in real time, loss of access to business data is a disaster, and end users don't tolerate downtime.

To successfully build and operate fast data applications on distributed architectures, there are 6 critical requirements for the underlying infrastructure.

1. High availability with no single point of failure

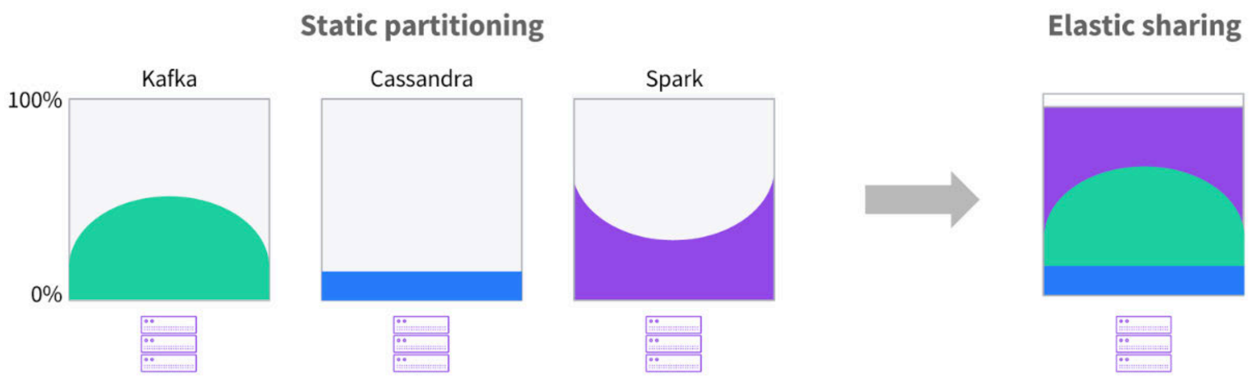
Always-on streaming data pipelines require a new architecture to retain high availability while simultaneously scaling to meet the demands of users. This is in contrast to batch jobs that are run offline—if a three-hour batch job is unsuccessful, you can rerun it. Streaming applications need to run consistently with no downtime, with guarantees that every piece of data is processed and analyzed and that no information gets lost.

Today, applications no longer fit on a single server, but instead run across a number of servers in a datacenter. To ensure each application (e.g. Apache Cassandra™) has the resources it needs, a common approach is to create separate clusters for each application. But what happens when a machine dies in one of these static partitions? Either there is extra capacity available (in which case the machines have been over-provisioned, wasting money), or another machine will need to be provisioned quickly (wasting effort).

The answer lies in datacenter-wide resource scheduling. Machines are the wrong level of abstraction for building and running distributed applications. Aggregating machines and deploying distributed applications datacenter-wide allows the system to be resilient against failure of any one component, including servers, hard drives, and network partitions. If one node crashes, the workloads on that node can be immediately rescheduled to another node, without downtime.

2. Elastic scaling

Fast data workloads can vary considerably over a month, week, day, or even hour. In addition, the volume of data continues to multiply. Based on these two factors, fast data infrastructure must be able to dynamically and automatically scale horizontally (i.e. changing the number of service instances), and vertically (i.e. allocating more or less resources to services), up or down. And so data doesn't get lost, scaling must occur with no downtime.



Elastic Resource Sharing Example

A shared pool of resources across data services facilitates elastic scalability, as workloads can burst into available capacity occupied in the cluster.

3. Storage management

Fast data applications must be able to read and write data from storage in real time. There are many kinds of storage, such as local file systems, volumes, object stores, block devices, and shared, network-attached, or distributed filesystems, to name a few. Each of these storage systems have different characteristics and each data service may require or support a different storage type.

In some cases, the data service by nature is distributed. Most NoSQL databases subscribe to this model and are optimized for a scale out architecture. In these cases, each instance has its own dedicated storage and the application itself has semantics for synchronization of data. For this use case, local, dedicated, persistent storage optimized for performance and resource isolation is key. Local persistent storage is "local" to the node within the cluster and is usually the storage resident within the machine. These disks can be partitioned for specific services and will typically provide the best in terms of performance and data isolation. The downside to local persistent storage is that it binds the service or container to a specific node.

In other cases, services that can take advantage of a shared backend storage system are better suited for external storage which may be network attached and optimized for sharing between instances. External storage in that case may be implemented in some form of storage fabric, distributed or shared filesystem, object store, or other "storage service".

4. Infrastructure and application-level monitoring & metrics

Collecting metrics is a requirement for any application platform, but it is even more vital for data pipelines and distributed systems because of the interdependent nature of each pipeline component and the many processes distributed across a cluster. Metrics allow operators to analyze issues across the data pipeline, including latency, throughput, and data loss. In addition, metrics allow organizations to gain visibility on infrastructure and application resource utilization, so that they can right size the application and the underlying infrastructure, ensuring optimum resource utilization.

Traditional monitoring tools do not address the specific capabilities and requirements of web scale, fast data environments. With no service-level metrics, operators cannot troubleshoot or monitor performance and/or capacity. Traditional monitoring tools can be adapted, but they require additional operational overhead for distributed applications. If monitoring tools are custom implemented, they require significant upfront and ongoing engineering effort.

To build a robust data pipeline, collect as many metrics as feasible, with sufficient granularity. And to analyze metrics, aggregate data in a central location. But beyond per-process logging and metrics monitoring, building microservices at scale also requires distributed tracing to reconstruct the elaborate journeys that transactions take as they propagate across a distributed system. Distributed tracing has historically been a significant challenge because of a lack of tracing instrumentation, but new tools such as OpenTracing[7] make it much easier.

5. Security and access control

Without platform-level security, businesses are exposed to tremendous risks of downtime and malicious attacks. Independent teams can accidentally alter or destroy services owned by other teams or impact production services.

Traditionally, teams build and maintain separate clusters for their applications including dev, staging, and production. As monolithic applications are rebuilt as microservices and data services, the size and complexity of these clusters continue to grow, siloed by teams and the technology being used. Without multitenancy, running modern applications becomes exponentially complex, because different teams

[7] http://opentracing.io

may be using different versions of data services, each configured with hardware expecting peak demand. The result is extremely high operations, infrastructure and cloud costs driven by administration overhead, low utilization, and multiple snowflake implementations (with unique clusters useable for only one purpose).

To create a multi-tenant environment while providing appropriate platform and application-level security, it is necessary to:

1. Integrate with enterprise security infrastructure (directory services and single sign-on).
2. Define fine-grained authentication and authorization policies to isolate user access to specific services based on a user's role, group membership, or responsibilities.

6. Ability to build and run applications on any infrastructure

Fast data pipelines should be architected to flexibly run on any on-premise or cloud infrastructure. For performance and scalability of fast data workloads, you need to have the choice to deploy infrastructure that meets the specific needs of your application. For example, the most sensitive data can be kept on-premises for privacy and compliance reasons, while the public cloud can be leveraged for dev and test environments. The cloud can also be used for burst capacity. Wikibon estimates that worldwide big data revenue in the public cloud was $1.1B in 2015 and will grow to $21.8B by 2026—or from 5% of all big data revenue in 2015 to 24% of all big data spending by 2026.[8]

For such hybrid scenarios companies often find themselves stuck with two separate data pipeline solutions and no unified view of the data flows. While the choice of infrastructure is vital, a key requirement is a similar

[8] Big Data in the Public Cloud Forecast, Wikibon, 2016

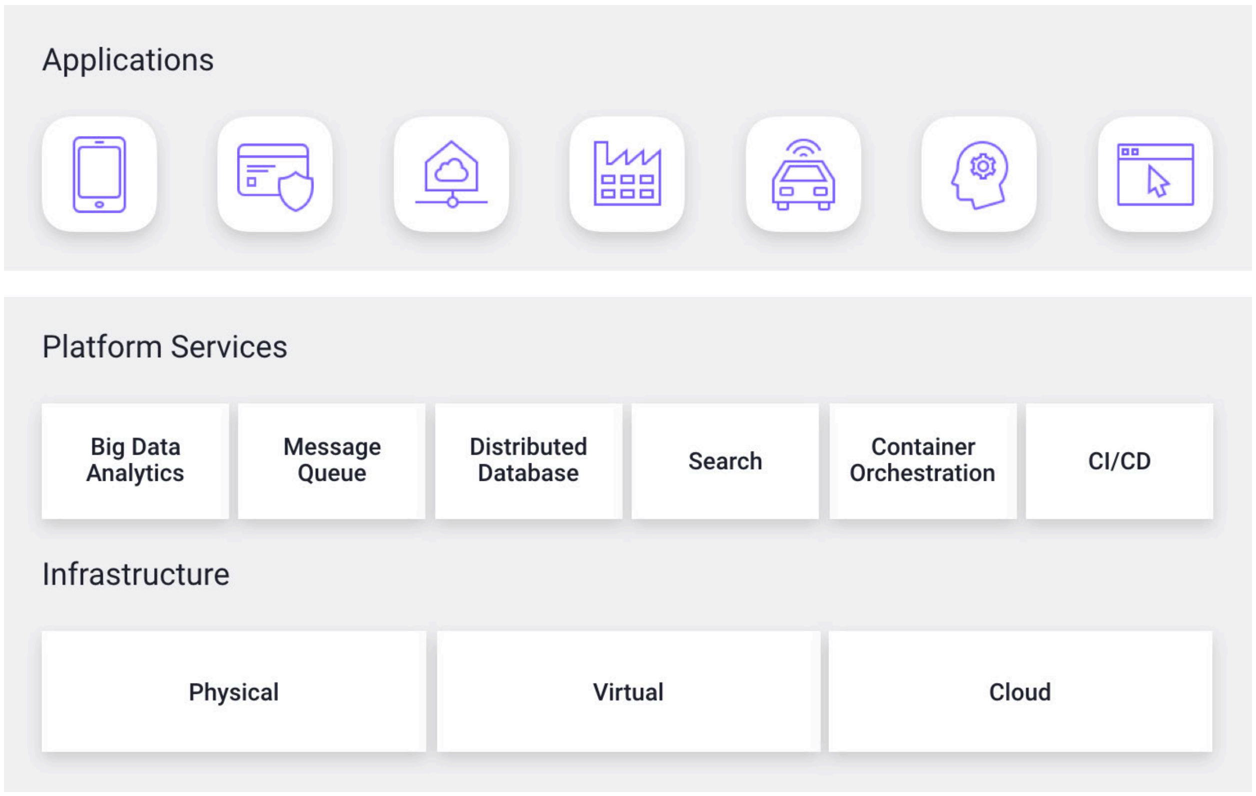
operating environment and/or single pane of glass, so that workloads can easily be developed in one cloud and deployed to production in another.

FAST DATA APPLICATIONS REQUIRE NEW PLATFORM SERVICES

We've covered the requirements for the underlying infrastructure for fast data applications. What about the fast data services deployed on this infrastructure, such as analytics engines and distributed databases?

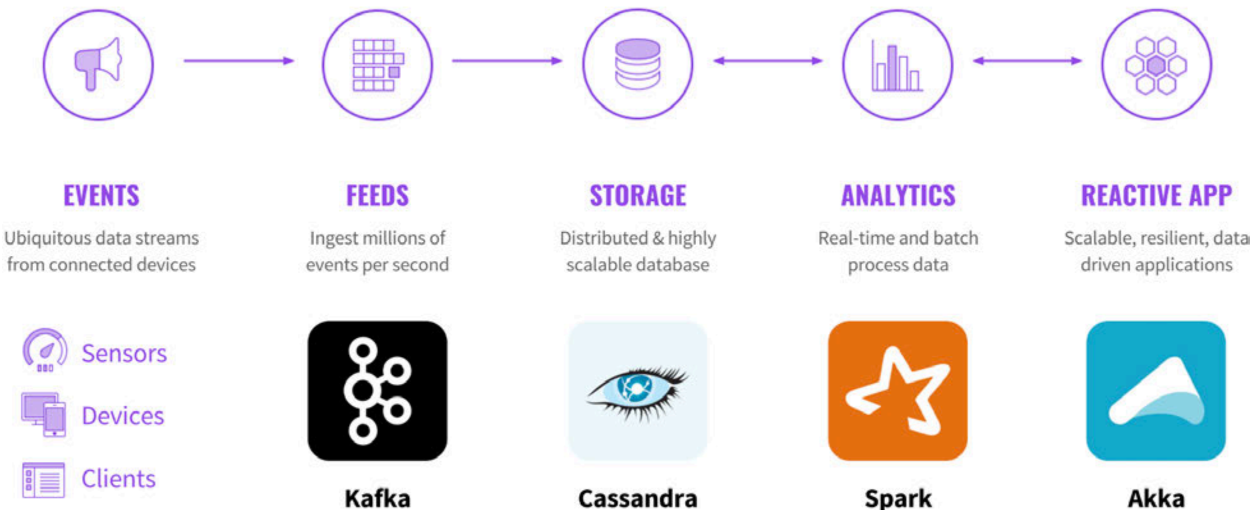
Another key shift in the architectural components of fast data systems is from the use of proprietary closed source systems to data pipelines stitched together from a variety of open source tools. In a recent survey, over 90% of respondents leveraged open source distributions for fast data applications, and almost 50% used open source exclusively.[9] Open source tools are a force multiplier for developers getting started, and also can be used to avoid lock-in to proprietary solutions.

[9] OpsClarity Fast Data Survey, 2016



Platform Services

Today, most people think of Hadoop or NoSQL databases when they think of big data. Recently, several open source technologies have emerged to address the challenges of processing high-volume, real-time data, most prominently including Apache Kafka™ for data ingestion, Apache Spark™ for data analysis, Apache Cassandra for distributed storage, and Akka for building fast data applications.


EVENTS
Ubiquitous data streams from connected devices
Sensors
Devices
Clients
FEEDS
Ingest millions of events per second
Kafka
STORAGE
Distributed & highly scalable database
Cassandra
ANALYTICS
Real-time and batch process data
Spark
REACTIVE APP
Scalable, resilient, data driven applications
Akka

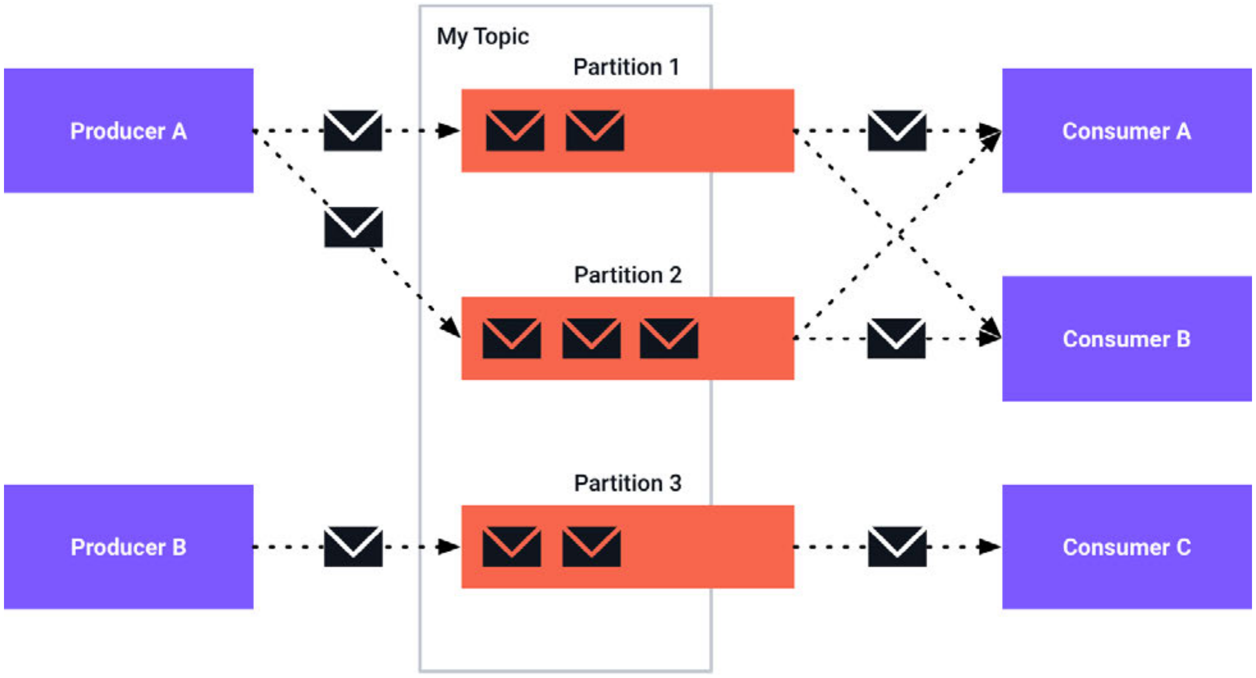
Fast Data Pipeline with Kafka, Cassandra, Spark, and Akka

1. Delivering real-time data

When constant streams of data arrive in millions of events per second from connected sensors and applications—from cars, wearables, buildings to everything else, the data needs to be ingested in real-time, with no data loss.

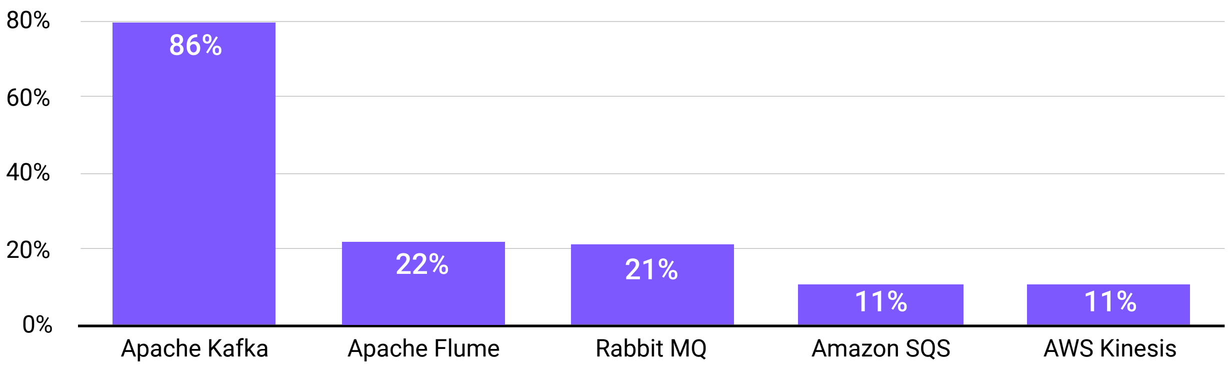
Apache Kafka, originally developed by the engineering team at LinkedIn, is a high-throughput distributed message queue system for ingesting streaming data. Kafka is known for its unlimited scalability, distributed deployments, multitenancy, and strong persistence.

Kafka allows companies to publish and subscribe to streams of records (i.e. as a message queue), store streams of records in a fault-tolerant way, and process streams of records as they occur. Kafka makes a great buffer between downstream tools like Spark and upstream sources of data, in particular data sources that cannot be queried again if data is lost downstream.



Apache Kafka Architecture

While Kafka is the most popular message broker, other popular tools include Apache Flume™ and RabbitMQ. Apache Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of data. RabbitMQ, backed by Pivotal, is a popular open source message broker that gives applications a common platform to send and receive messages. RabbitMQ is preferred for use-cases requiring support for Advanced Message Queuing Protocol (AMQP).



Most Popular Ingestion Queues

Source: OpsClarity Fast Data Survey, 2016

2. Storing distributed data

Traditional relational databases (RDBMSs) were the primary data stores for business applications for 20 years, and new databases such as MySQL were introduced with the first phase of the web. However, the scaling and availability needs of modern applications require a new highly durable, available, and scalable database to store streaming and processed data.

Apache Cassandra is a large-scale NoSQL database designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. Cassandra supports clusters spanning multiple datacenters, providing lower latency and keeping data safe in the case of regional outages.

Some of the largest production deployments include Apple's, with over 75,000 nodes storing over 10 PB of data, Netflix (2,500 nodes, 420 TB, over 1 trillion requests per day), Chinese search engine Easou (270 nodes, 300 TB, over 800 million requests per day), and eBay (over 100 nodes, 250 TB).[10]

Other popular distributed NoSQL databases include MongoDB, Redis, and Couchbase:

- MongoDB is an open-source, document database designed for ease of development and scaling.
- Redis is an in-memory data structure store, used as a database, cache and message broker, for high performance operational, analytics or hybrid use
- Couchbase is a NoSQL database that makes it simple to build adaptable, responsive always-available applications that scale to meet unpredictable spikes in demand and enable mobile and IoT applications to work offline.

[10] http://cassandra.apache.org/

3. Processing fast data

Incoming data from sensors and applications needs to be processed using batch, streaming, machine learning, or graph compilation to gain new insights. Hadoop is a well-known tool for batch analysis, but Hadoop MapReduce has limitations for rapid analysis of smaller datasets.

Apache Spark is an open source processing engine built around speed, ease of use, and sophisticated analytics. Originally developed at UC Berkeley in 2009, Spark allows companies to run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.[11] It is easy to use, with the ability to write applications quickly in Java, Scala, Python, and R. Spark supports SQL, streaming, machine learning, and graph computation.

Apache Spark is seeing exponential growth in adoption and awareness. Today, Spark remains the most active open source project in big data, with over 1,000 contributors. In a recent Spark survey by Taneja group, nearly one half of all respondents were already using Spark, and of those, 64 percent say that it's proving invaluable and that they intend to increase usage of Spark within the next 12 months.[12]

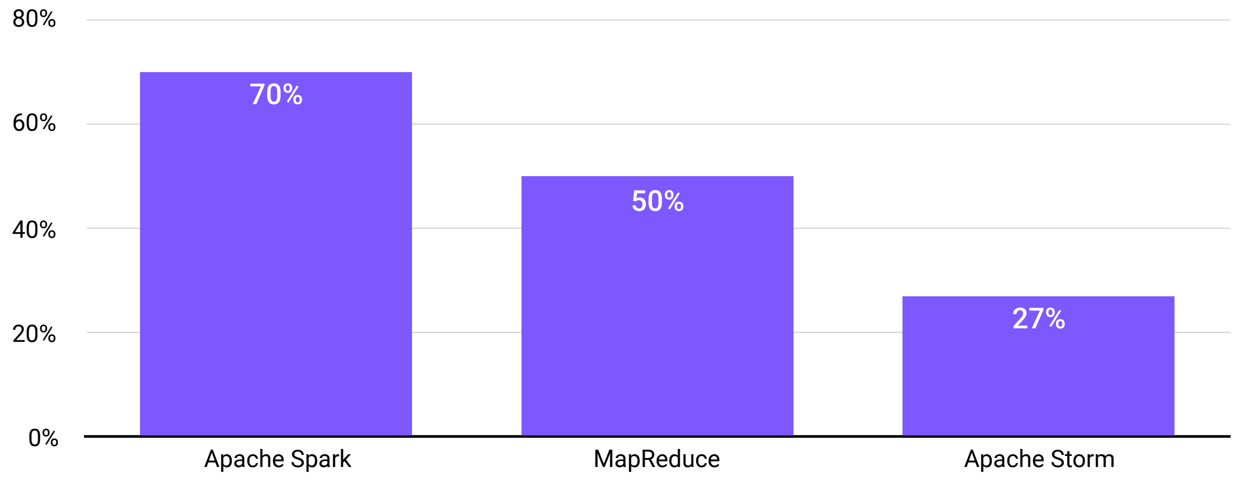
Other popular analytics tools include MapReduce, Apache Storm™ and Apache Flink™:

- MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster. Spark runs in memory, while MapReduce is batch-based and is restricted to writing to and from disk. MapReduce is part of Hadoop and thus requires HDFS as its primary storage layer. MapReduce is included in Hadoop distributions, and powers approximately half of processing environments.

[11] http://spark.apache.org/

[12] Apache Spark Market Survey, Taneja Group, November 2016

- Apache Storm, an open source distributed realtime computation system, is another popular stream processing platform. Storm makes it easy to reliably process unbounded streams of data, doing for realtime processing what Hadoop did for batch processing.
- Apache Flink is an open source distributed data stream processor; Flink provides efficient, fast, consistent, and robust handling of massive streams of events, as well as batch processing as a special case of stream processing.



Most Popular Data Processing Technologies

Source: OpsClarity Fast Data Survey, 2016

4. Acting on data

Once real-time data is analyzed, insights need to be presented to a human or trigger actions in connected devices or applications. Akka is a popular toolkit and runtime to simplify development of data centric applications. Akka was designed to enable developers to easily build reactive applications using a high level of abstraction, and the technology makes building highly concurrent, distributed, and resilient message-driven applications on the JVM a much simpler process.

KEY CHALLENGES IMPLEMENTING FAST DATA SERVICES

While the use of open source tools such as Kafka and Spark has grown immensely, the distributed nature of these fast data technologies can make them difficult to deploy and operate. For example, Braintree, a mobile and web payment systems company, built their own data pipeline using open source software and lots of custom code. Braintree devoted a team of four full-time engineers for six months to get their data infrastructure off the ground, and even after the initial launch, a two-person team is required to maintain and extend the project.[13]

But just why is building and maintaining data infrastructure so taxing?

1. Deploying each data service is time consuming

First, deploying each data service is time consuming. Installing a production-grade platform service such as Kafka or Cassandra requires specialized knowledge for operators; even for an expert, deployment is time consuming and often requires significant engineering effort. For example, when the engineering team at AirBnB first attempted to deploy Kafka, it took multiple months, and ultimately ran into many issues. Additionally, if fast data services are not architected correctly, the result is snowflake implementations that are dependent on key personnel to maintain.

In the past five to ten years, an explosion of datastores and analytics engines have emerged, many open source. Beyond the significant cost of

[13] Stitchdata blog, “Why you shouldn’t build your own data pipeline”

resources to deploy each new data service, a primary downside of deployment challenges is that developers and data engineers cannot easily experiment with these new technologies.

Explosion of Datastores and Analytics Engines

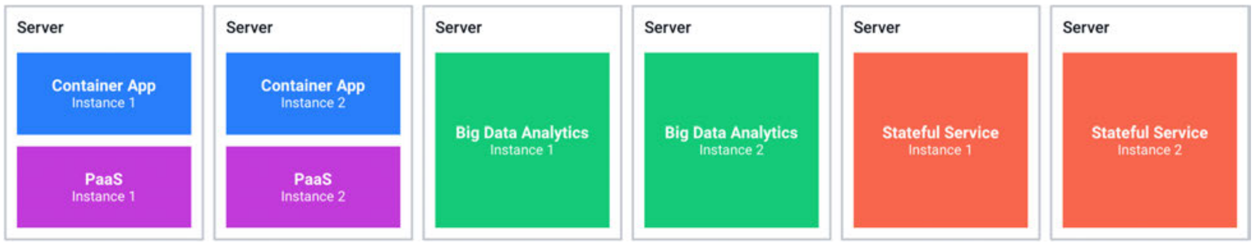
2. Operating data services is manual and error-prone

The second major challenge with implementing fast data technologies is the ongoing operations of these technologies; common tasks such as upgrading software, deploying updates, rolling back in failure scenarios, monitoring health, and managing storage resources are often manual and error-prone. This time-consuming ongoing maintenance requires additional headcount and can be a drag on new innovation.

Applications teams and operators need the latest capabilities in data services in order to fix critical bugs, add valuable features, and reduce operational overhead. Software upgrades and updates are time consuming because operators and developers manually upgrade the services or are required to build configuration management and orchestration software to automate the upgrade. In addition, updates and rollback need to be thoroughly validated on a testing environment before they are ready for production.

3. Infrastructure silos with low utilization

The third key challenge with operating fast data infrastructure is maintaining enough infrastructure to handle data-processing peaks in a cost-effective way. Average datacenter utilization is approximately 6-12%[14] and remained static from 2006-2012[15], driven by companies' desire to maintain high service quality through peak load periods.



Traditional Approach: Silos with Low Utilization

The increased utilization benefit of virtual machines does not apply to distributed systems. Operators typically create separate clusters for Spark, Kafka, Cassandra, and so on. This is because these services are distributed systems with their own scheduling logic to grow and shrink their footprint, so operators run them in different clusters to avoid resource conflicts. For example, the Spark data processing service may want all the capacity it can get to finish a job, the Kafka message queue may need resourcing based on the volume of data passing through, while the Cassandra distributed database may need steady resourcing to persist data. The result is extremely low utilization, and waste of infrastructure resources.

[14] "The Sorry State of Server Utilization and the Impending Post Hypervisor Era", Gigaom, November 2013

[15] NRDC Data Center Efficiency Assessment, August 2014

Rather than running workloads in silos based on the application, datacenter-wide resource sharing drives increased utilization as all workloads can access a shared pool of resources.

PUBLIC CLOUD - THE SOLUTION?

Many companies are shifting fast data workloads to the cloud or hybrid environments. The two key reasons for moving the cloud are:

1. To get immediate access to platform services such as analytics tools and databases. Cloud providers such as Amazon Web Services and Microsoft Azure package data services and make them easy to install and operate.

2. To enable scalability and elasticity, an imperative for bursting fast data workloads.

In a recent survey, less than a third of companies hosted their data pipeline on-premises, while more than two thirds hosted in the cloud or hybrid environments.[16]

While public cloud provides clear advantages for fast data workloads, the major downside is the risk of lock-in. Applications that are developed using public cloud platforms are tied to a specific cloud provider's APIs, and moving workloads after the fact is near impossible without rewriting them.

One recent story highlights the risk of cloud lock-in, that of Snap (of Snapchat). In the S1 Registration Statement issued by Snap[17] in February 2017, it came to light that Snap had handcuffed itself to Google Cloud. Of the annual loss of over $500 million, 80% was attributed to contractually obligated spend with Google. In the same filing, Snap states that they wrote their application to use some Google services "which do not have an alternative in the market." Google now has them in handcuffs, and there is

[16] OpsClarity Fast Data Survey, 2016

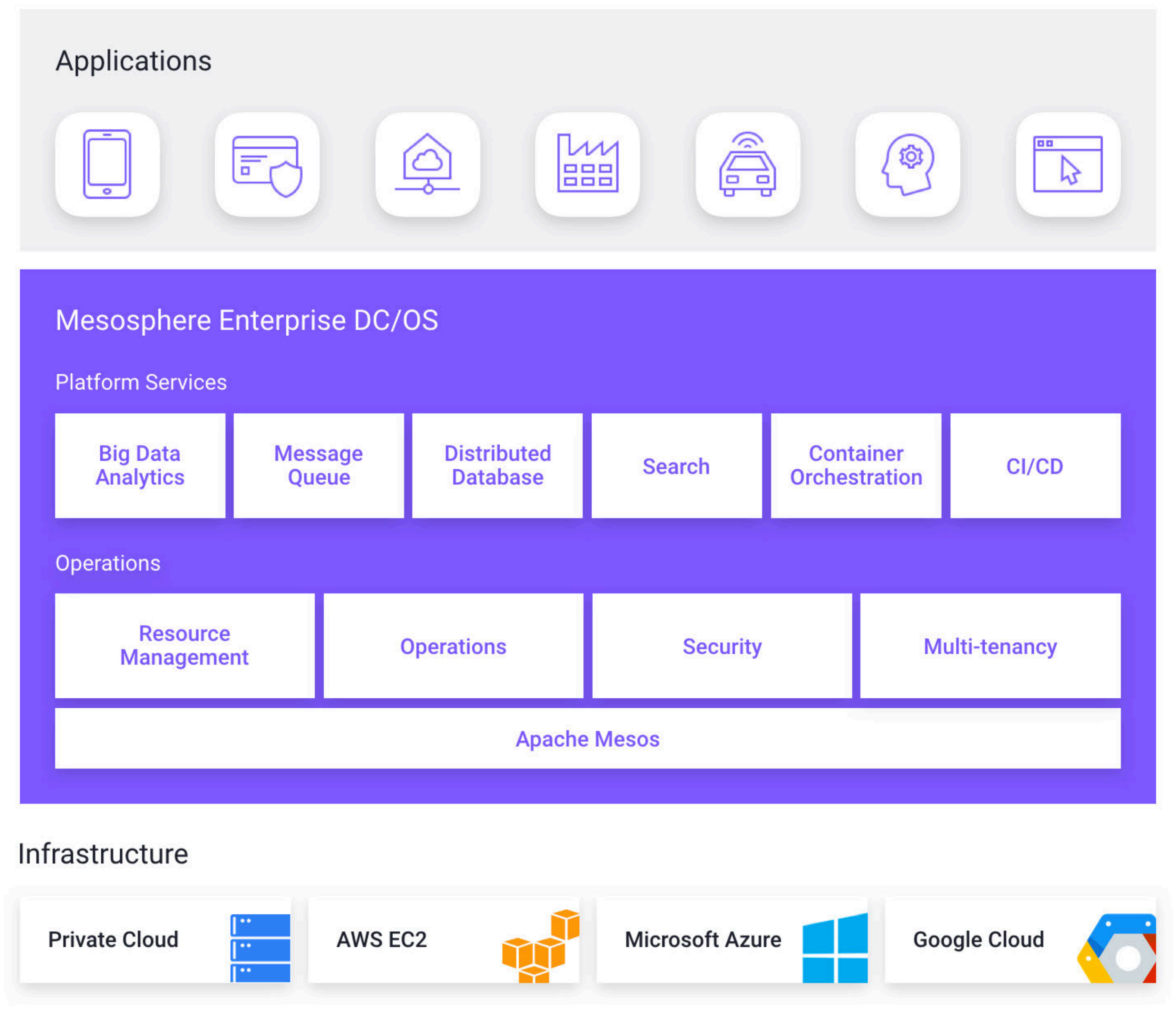
[17] Snap S1 Registration, February 2017

little Snap can do to change that without having to invest a tremendous amount of money to free themselves.

As companies continue to rush to the cloud to quickly build new services, the costly risk of lock-in continues to mount. But what is the alternative?

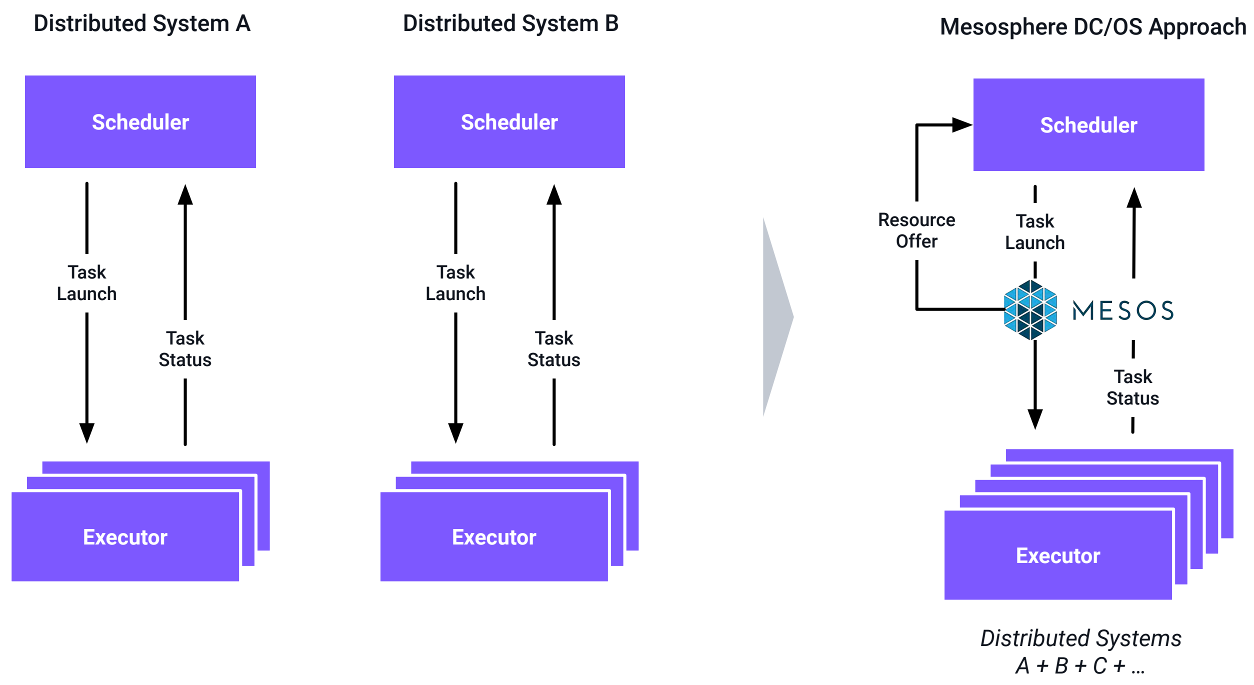
MESOSPHERE DC/OS: SIMPLIFYING THE DEVELOPMENT AND OPERATIONS OF FAST DATA APPLICATIONS

Mesosphere delivers a platform for building and running highly scalable data pipelines, in any cloud or datacenter. Mesosphere DC/OS accelerates deployment and simplifies operations for a broad set of data services including databases, message queues, analytics engines, and more. Mesosphere enables experimentation with new data services and provides a future-proof platform that is highly available and scales to meet the demands of users.



Mesosphere DC/OS: Cloud Native Platform Services on Any Infrastructure

The core of DC/OS is the Apache Mesos™ distributed systems kernel. Its power comes from the two-level scheduling that enables distributed systems to be pooled and share datacenter resources. Mesos provides the core primitives for distributed systems, such as resource allocation, isolation, and quota management. DC/OS provides a highly-available infrastructure for fast data workloads—workloads are automatically restarted when a server fails. Pooling resources across a datacenter or cloud also enables elastic scaling, where workloads can scale up or down based on demand.

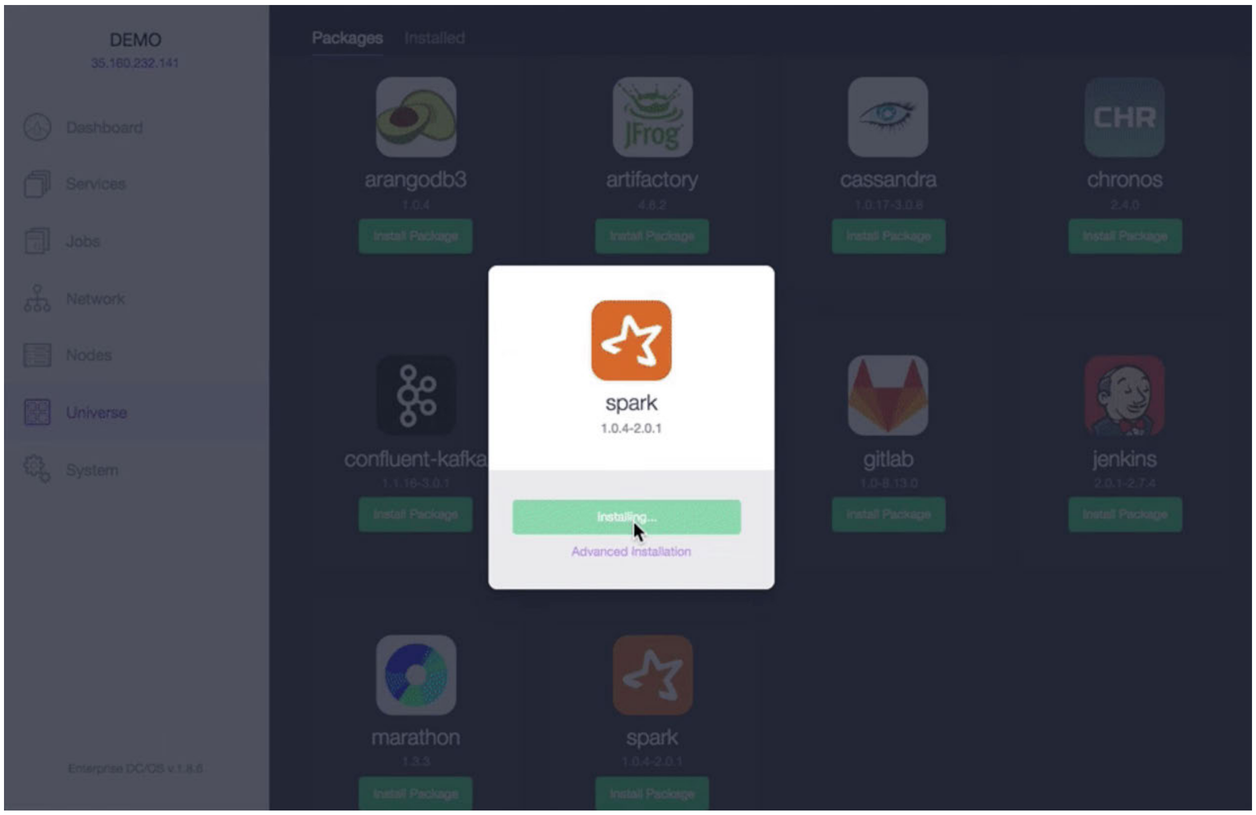


Apache Mesos Two-Level Scheduling

Two-level scheduling in DC/OS provides key differentiators versus simply running services in containers on Kubernetes or Docker Swarm, and simplifies the three key challenges with operating data services—deploying services, operating them, and overcoming silos with low utilization.

1. On-demand provisioning

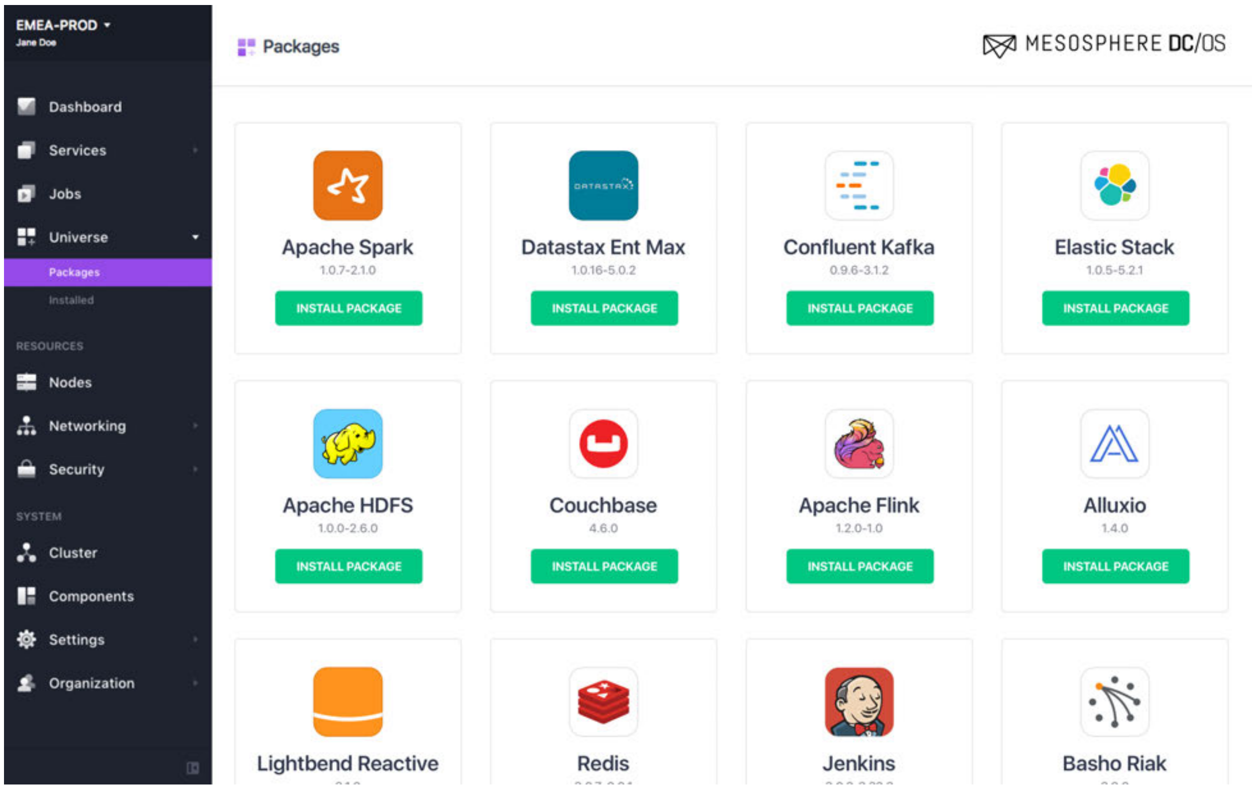
Mesosphere DC/OS enables single-command install of data services such as Spark, Cassandra, Kafka and Elasticsearch, among many others. Where deployment of these services used to be incredibly time-consuming and error prone, data services can be up and running across an entire cluster in a matter of minutes with Mesosphere DC/OS.



Data Service Installation with Mesosphere DC/OS

The DC/OS Universe is an open source ecosystem where anyone can publish services to be installed on DC/OS. The DC/OS Universe includes both open source and partner-supported data services, including products from DataStax, Confluent, Elastic, and Alluxio, among many others. There is a growing community of users and partners creating new DC/OS packages.

In addition, an open source SDK provides a high-level interface for building new stateful services on DC/OS. Developers can write a stateful service complete with persistent volumes, fault tolerance, and configuration management in about 100 lines of code. This SDK is the product of Mesosphere's experience writing production stateful services for DC/OS such as Kafka, Cassandra and HDFS.



Mesosphere DC/OS Universe with Over 100 Services

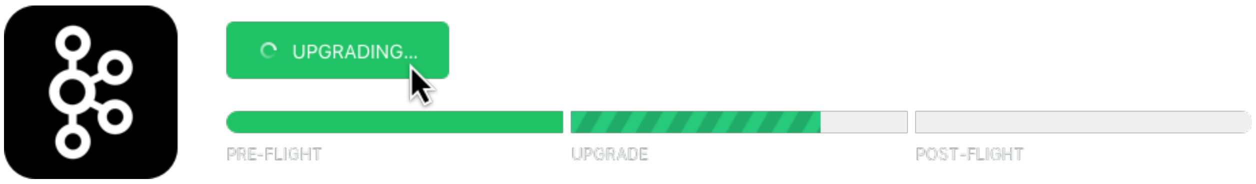
Faster deployment of data services allows companies to avoid finding and paying for specialized talent, enables faster time to market for new fast data applications, and allows data scientists to experiment with the broad swath of new data services, and build on a composable architecture.

Mesosphere DC/OS also dramatically simplifies resizing instances of a data service, as well as adding more instances. With Mesosphere DC/OS, operators can easily scale up and scale out data services, with no downtime.

2. Simplified operations

Mesosphere DC/OS dramatically reduces the time and effort involved with operating data services through simple runtime software upgrades and updates, application-level monitoring and metrics, and managed persistent storage volumes.

A key challenge with operating data services is upgrading and updating config settings in the data services themselves—the risk is downtime and wasted operator time. Mesosphere DC/OS includes built-in runtime software upgrade capabilities, with rollback in case of failure. Application setting updates can also be performed during runtime using the same mechanism. Runtime upgrades and updates with the ability to rollback minimizes maintenance windows and the risk due to unsuccessful config updates, and saves on operator and engineering time.



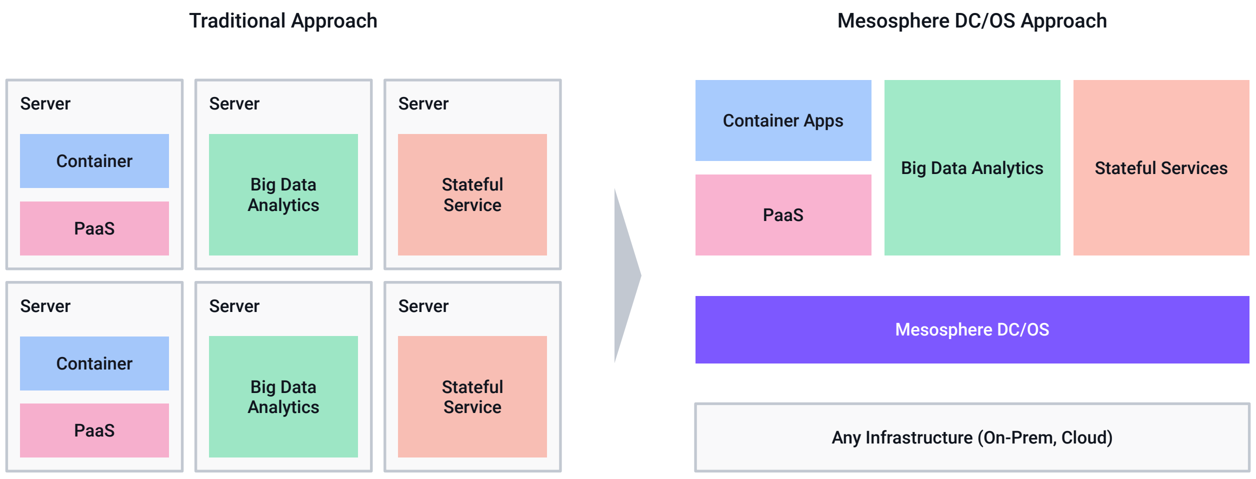
Mesosphere DC/OS Data Service Upgrades

Mesosphere DC/OS provides out-of-the-box application-level monitoring and troubleshooting, so operators can easily troubleshoot or monitor performance and capacity. DC/OS services send metrics to the customer's provided statsd metrics service, and commercial monitoring tools integrate to provide aggregate metrics and logs per node, app and container.

Mesosphere DC/OS also simplifies operations by managing storage volumes, like CPU, memory and network resources, so operators do not need to manually keep track of resources. As a result, administration time and overhead for provisioning required resources for data services at scale are significantly reduced.

3. Elastic data infrastructure

Mesosphere DC/OS enables multiple data services, containerized applications and traditional applications to all run on the same infrastructure, dramatically increasing utilization. Some Mesos and DC/OS users have approached average utilization rates of over 90%, and reduced hardware and cloud costs by over 60%. Operators can easily bring new services such as Kafka online, or add more instances of the same service to a cluster with already available resources, increasing efficiency. In addition, app teams can use the specific version of software they prefer, without the need to create separate silos.



Elastic Data Infrastructure with Mesosphere DC/OS

To increase utilization further, multiple users and teams can share a DC/OS cluster. Built-in security features such as fine grained access control lists, secrets management, and integration with directory services (LDAP) and single sign-on solutions (SAML, OpenID connect) enable companies to isolate access to specific services based on a user's role, group membership, or responsibilities.

CASE STUDIES: FAST DATA DONE WELL

verizon✓

Verizon Adopts New Strategic Technologies to Serve Millions of Subscribers in Real-Time

Verizon Communications is the #1 wireless phone service in the US. The company's core mobile business, Verizon Wireless, serves about 113 million retail connections.

Verizon needed to easily deploy and manage thousands of Docker containers and needed to be able to quickly adopt new strategic technologies such as Apache Spark. Verizon's research identified Apache Mesos as the best option given Verizon's scale. Verizon chose Mesosphere DC/OS because it let the company more easily adopt Mesos and the necessary components for proper microservices architectures, and helped Verizon adopt new strategic technologies such as Apache Spark for data-processing and analytics.

Verizon uses Mesosphere DC/OS across thousands of nodes across multiple datacenters, and is continuing to expand deployment. Verizon uses Spark, Kafka and Cassandra, among other data services. Thanks to strong collaboration between Verizon and Mesosphere, Mesosphere DC/OS is now the computing backbone of next-generation products—on-demand TV on Fios, smartphone streaming via Go90, and smarter devices via the Internet of Things—serving tens of millions of consumers.

Mesosphere DC/OS enabled:

- Shared data infrastructure: The ability to run Spark, Hadoop, Kafka, Cassandra and more on a single Mesosphere DC/OS cluster lets Verizon build data-driven applications without moving data across clusters or managing multiple environments.
- Faster time to market: No dedicated hardware means applications come online in a fraction of the time compared with historical approaches. Verizon deployed a Mesosphere DC/OS datacenter environment in the time it used to take to stand up application infrastructure.
- Acting fast on new trends: From streaming video services like Go90 to drone video analysis, Mesosphere DC/OS lets Verizon act quickly on the types of applications its consumer and enterprise customers demand.

“Mesosphere DC/OS gives Verizon far-reaching benefits to quickly launch new products and services while reducing the IT requirements in our data centers”

- Kumar Vishwanathan, VP & Chief Technologist, Verizon

Esri Builds Real-Time Mapping Service With Kafka, Spark, and More

Esri is a world leader in geographic information system (GIS) technology. Esri's clients wanted to deploy new IoT and data-driven applications that were pushing the boundaries of Esri's current on-premises Real-Time GIS software solution. The performance and scalability demands of these new applications required Esri to adopt a new technology platform to help customers take their applications to the next level.

Esri has developed a new managed service—built on Mesosphere DC/OS—that lets users achieve real-time, predictive mapping. It combines Esri's ArcGIS platform with real-time and big data analytic capabilities to process and analyze up to millions of events per second from sources such as:

- Sensors on moving objects such as vehicles, vessels and people
- Stationary sensors on electric, water and gas utility networks
- Feeds from social media, weather and environmental sensors

Esri utilizes Apache Kafka, Apache Spark (and Spark Streaming), Elasticsearch, and the Lightbend Reactive Platform (which includes Akka) to bring their service to market.

Mesosphere DC/OS enabled:

- Increased performance and scalability: Esri's previous on-premises software was able to process thousands of events per second. On the Mesosphere DC/OS platform, Esri's new managed service can process millions of events per second, easily meeting the expanding needs of its clients.

- New class of customer: Esri is now getting requests from a new class of customer with more sophisticated and large-scale applications. Before DC/OS, Esri would shy away from those opportunities because it was outside the scale of what its technology could handle.
- Faster time to market: The time to get a customer's environment up and running has been drastically reduced. Depending on the deployment, it could take anywhere from hours to days to get the environment established and sometimes required sending staff to the customer site if the customer requested. With the new cloud-based platform, Esri can deploy a new cluster in minutes.
- A foundation for innovation: The Mesosphere DC/OS platform provides an innovative delivery model and is also a foundation to build higher-level business applications. Having the investment in place enables Esri to move from real-time GIS to predictive GIS, where they can make predictions and recommendations rather than only providing alerts about what has already happened.

"With the Mesosphere DC/OS platform, we can serve a new set of customers with entirely new capabilities in terms of the performance and intelligence of their map and analytic applications. And with our cloud-based platform, we can get these solutions up and running in minutes. This gives Esri and our clients a level of innovation and business agility we've never had before."

- Adam Mollenkopf, Real-Time & Big Data GIS Capability Lead, Esri

Wellframe Expands its Healthcare Management Platform

Wellframe's intelligent care-management platform allows health plans and care-delivery organizations to better manage large populations of patients with complex medical conditions such as diabetes, heart disease and transplants. To meet the diverse and ever changing needs of these patients, Wellframe provides personalized and adaptive care programs that dynamically adjust based on a combination of data from the healthcare system and the Wellframe platform.

Wellframe faced a twofold challenge as its healthcare management platform matured: increased complexity and limited resources. As Wellframe's business and product offering became more sophisticated, the company needed to add new services. Each service leveraged new and different technologies (including Apache Spark, Apache Kafka and Apache Cassandra) and required additional hardware and engineering resources to manage. With its small team and limited resources, Wellframe soon realized it needed a different approach.

Wellframe had experience managing infrastructure with Apache Mesos, and it chose DC/OS as the long-term foundation for its personalized healthcare offering because DC/OS offered a more complete and much simpler platform experience. Wellframe also was interested in the growing number of services that DC/OS supports—including Spark, Cassandra, Kafka and more being added all the time. The company liked the unified user experience of being able to manage all of these systems through a single interface.

Mesosphere DC/OS allows Wellframe to spend more time building technology that directly affects the core business of the company, and less time on the operational management that comes with running a large production deployment. DC/OS allows Wellframe to add services and technologies with very few people to manage the infrastructure.

“DC/OS has allowed us to take on, manage and open up wide areas of business that we couldn’t address before. We are able to expand our business to new geographies and deliver services that address the complexity of serving patients managing chronic health conditions.”

- **Gopal Ramachandran, CTO, Wellframe**

ABOUT MESOSPHERE

Mesosphere is leading the enterprise transformation toward distributed computing and hybrid cloud. We combine the rich capability you get from public cloud providers with the freedom and control of choosing your own infrastructure. Mesosphere DC/OS is the premier platform for building, deploying, and elastically scaling modern applications and big data services. DC/OS makes running containers, data services, and microservices easy across your own hardware and cloud instances. Mesosphere was founded in 2013 by the architects of hyperscale infrastructures at Airbnb and Twitter and the co-creator of Apache Mesos. Mesosphere is headquartered in San Francisco with additional offices in New York and Hamburg, Germany. Mesosphere's investors include Andreessen Horowitz, Hewlett Packard Enterprise, Khosla Ventures, Kleiner Perkins Caufield & Byers, and Microsoft.

MESOSPHERE