Whitepaper

Kubernetes: the cornerstone of the modern cloud infrastructure

ITSVIT

Kubernetes: the cornerstone of the modern cloud infrastructure

Kubernetes is a huge leap forward from disparate isles of platform-specific services and tools to cloud-agnostic immutable infrastructure as code. It is an influential cloud technology, around which new tools and practices are growing. Kubernetes turned out to be so-o-o convenient for container management, it became the de-facto standard for deploying and managing easily scalable applications to clouds, hosting providers, local nodes. Just as Linux has became the most commonly used operating system for server-side system management, Kubernetes may become the most used tool for developing and managing the cloud-native or on-prem infrastructures.

To say even more, every time somebody adds anything new to an already impressive list of convenient tools for Kubernetes — be it Helm, Brigade, or any other instrument — these DevOps tools are accepted, used and improved by the ever-growing community of DevOps engineers. All such additions are improving the usefulness and performance of Kubernetes ecosystem, moving the software development industry forward in seven-league steps.

In this white paper we describe 5 reasons why Kubernetes has become the cornerstone of the modern cloud architecture, its prominent use cases and future prospects.

1. Kubernetes is the de-facto standard of container management.
2. Kubernetes has won the container orchestration race.
3. Kubernetes is adopted as a service by cloud giants like AWS and Azure.
4. Kubernetes helps building an infrastructure that avoids Cloud Service Provider (CSP) vendor lock-in.
5. Kubernetes enables ease of distributed systems management with Helm.

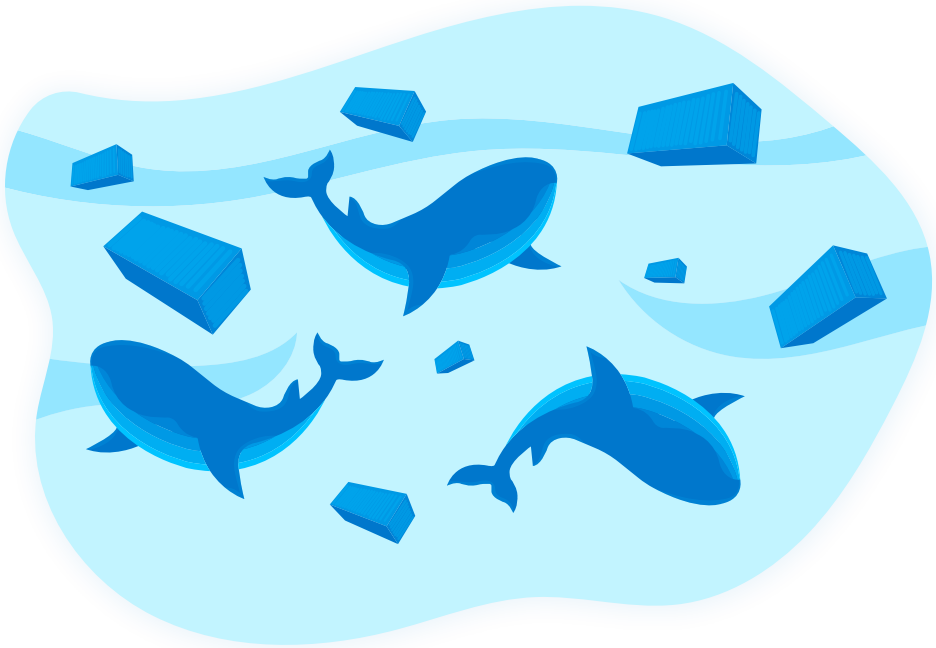
Kubernetes as a new standard of container management

Standardizing has both positive and negative sides. When anything is built according to a certain standard, the owner can expect two important things: it will fit other pieces built according to the same standard and there will be some measurable audience willing to use this product. This is quite profitable for the product vendor, as it assures the customers will be 100% able to use their offers on supported devices, like iOS apps on iPhones or Windows 10 on all PC's.

From the other point of view, proprietary standardization inevitably leads to fragmentation of the market. Sorry, boy, this shiny iPhone app runs on iPad, not on Kindle. Snapchat-style augmented reality emojis? Nope, not in Facebook Messenger. This is the other side of the coin with measurable audiences — these audiences are LIMITED.

Developers despise limitations. Developers go open-source to avoid vendor lock-in, go cloud-agnostic to be able to move their apps between the clouds on demand. Developers address the issues of proprietary platforms and make sure their open-source code is free from such flaws. This is the way Linux became the #1 server OS and nginx became the #1 web server out there, as we described in our article on the state of DevOps adoption in 2017.

Another wave of open-sourced standards came with Docker, which allowed to encapsulate any app along with the runtime environment for it and run it on any machine with a Docker installed. Any such container is now a node that can be easily composed, launched or shut down, but how does one manage such containers, when there are hundreds of them?



This is why Kubernetes is so great — it simplifies the container management immensely, allowing the DevOps team to know at all times what is happening with their containerized applications within the cluster and the cluster itself. It also simplifies the issues fixing, when they arise.

Container orchestration with Kubernetes

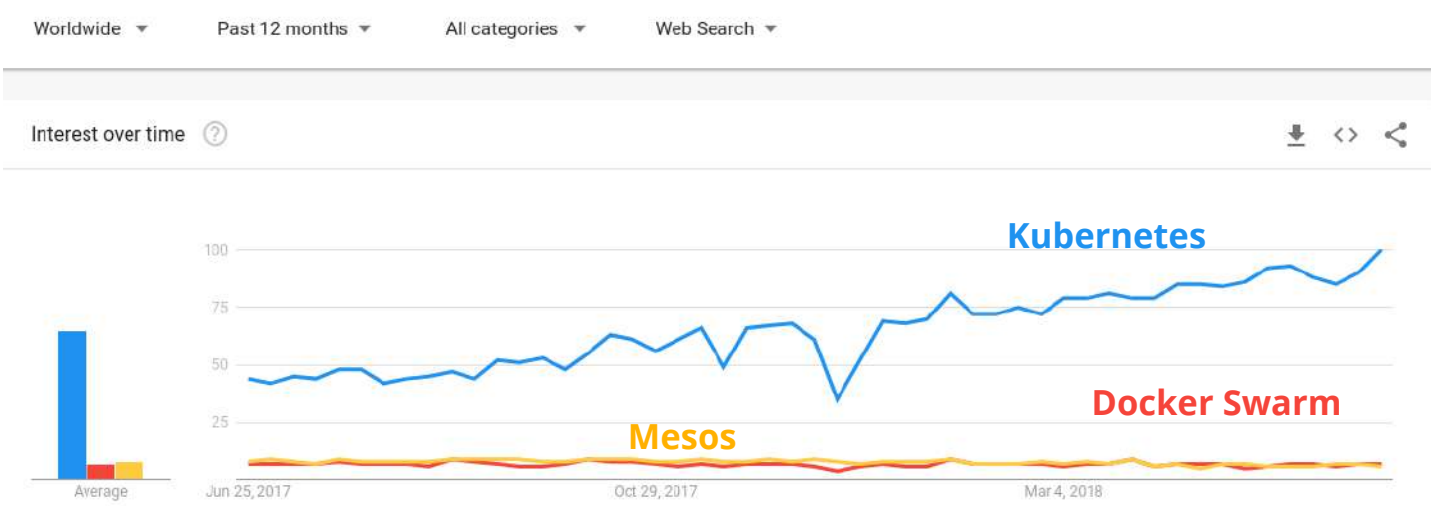
Container is as essential to a distributed software architecture, as an object is to an OOP. As soon as DevOps specialists worldwide understood Docker is the solution that is there to stay, they began looking for the ways to streamline and standardize the container orchestration processes.

Container Orchestration Processes

Scheduling	Resource Management	Service Management
• Placement	• Memory	
• Replication/ Scaling	• CPU	• Labels
• Resurrection	• GPU	• Groups/ Namespaces
• Rescheduling	• Volumes	• Dependencies
• Rolling Deployment	• Ports	• Load Balancing
• Upgrades	• IPs	• Readiness Checking
• Downgrades		
• Collocation		

There were various attempts to reach this goal, like Docker Swarm, Mesos and Kubernetes. Cloud service providers (CSP) like AWS or Azure provided their platform-specific tools for container management, and many cloud architects are still refusing to use Kubernetes, preferring to write their own BASH scripts or capabilities of their favorite tools like Puppet or Chef.

The sad truth for them is that the open-source tool is always the best, as there is a passionate community generating ideas that help Kubernetes evolve in the best directions possible. To say even more, the industry-leading giants like AWS or GCP, Azure or DigitalOcean implemented the native Kubernetes support to their platforms and have already built multiple services leveraging Kubernetes features and capabilities. Thus said, in the end of 2017 it already became obvious that Kubernetes was the winner of the container orchestration tool race, and it remains this way halfway through 2018.



Kubernetes as a service by AWS and Azure, GCP and DO

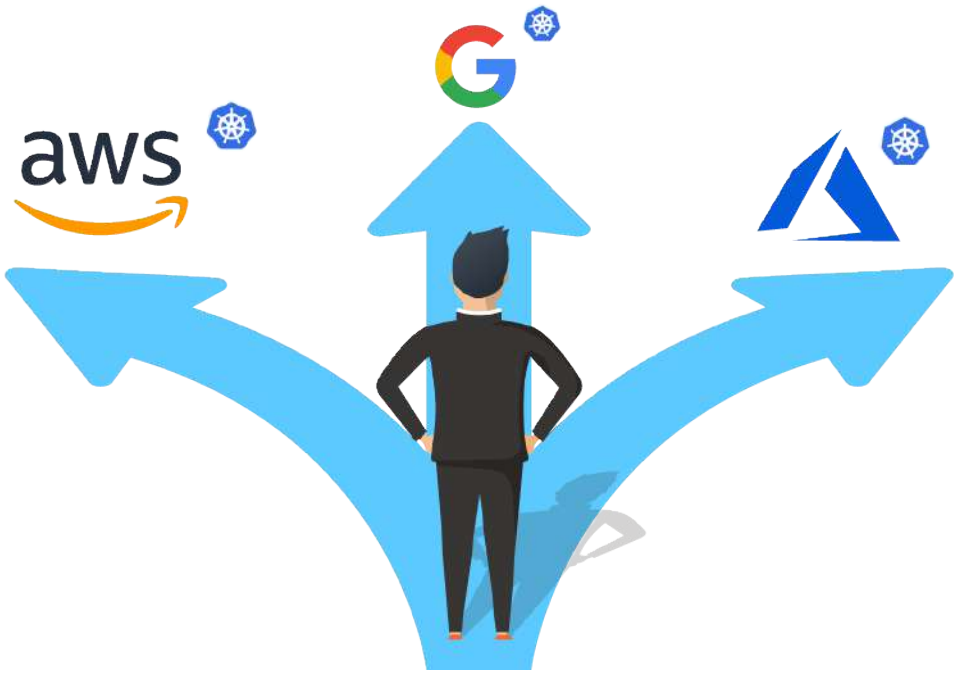
You might recall the time when Microsoft offered container provisioning using Docker, Kubernetes or Mesos/Marathon through their Azure Container Service (ACS). However, it did not gain much popularity due to the powerful rival nearby — the Google Container Engine, which offered managed Kubernetes support with automatic updates and configurable user interface.

Microsoft has re-launched ACS under new acronym AKS — Automated Kubernetes Service, now providing adjustable user interface, automatic Kubernetes updates and other features making it a worthy match for GCE. In addition, Microsoft launched another service named Azure Container Instances (ACI), which allows to manage the containers in a simple and fast way, without the need to manage the underlying VM infrastructure. Thus said, Azure users can combine AKS and ACI to get the best of two worlds — freedom to launch and manage containers while granularly paying only for the resources used.

AWS also offers Kubernetes through their Amazon EKS feature introduced late November 2017. This was a great progress as compared to the process of deploying Kubernetes cluster to EC2 instances using 4 possible ways, with Amazon supporting none of them. Amazon has also launched Amazon Fargate, the analog of Azure ACI service, allowing anyone to spin up containers without handling the underlying infrastructure, using the payment scheme of the serverless computing.

This might be just the future of cloud computing right there — running the apps without having to worry what happens under the hood. It also brings simplicity to containerized infrastructures, as AWS tech support team handles all the infrastructure-related activities and management. Is it good? Is it bad? This is to find out yet, once EKS management with Fargate gains more traction.

As Kubernetes was initially a Google product, Google Kubernetes Engine is the most advanced service of them all, and other cloud service providers do their best to match this offer.

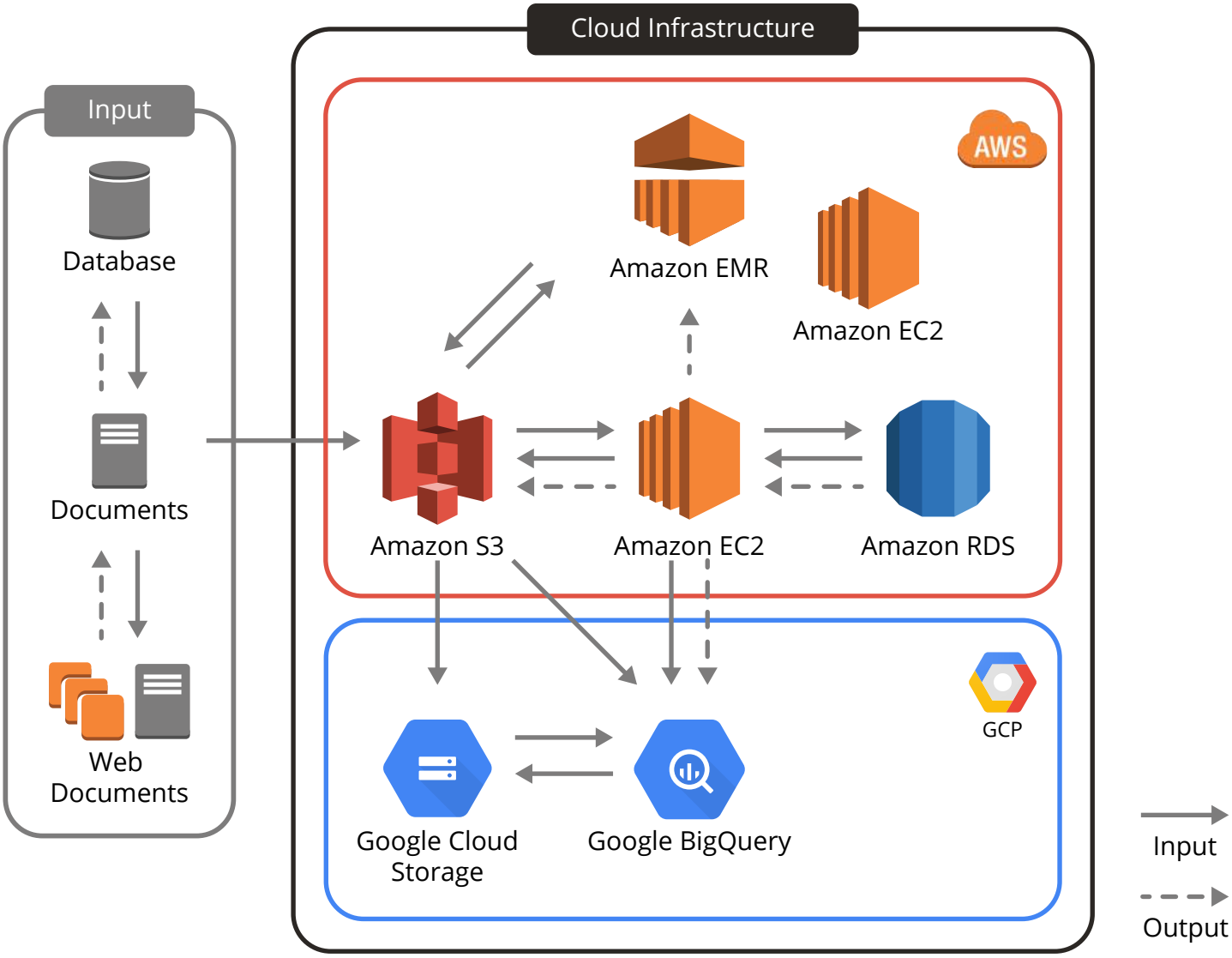


DigitalOcean is currently in the process of beta-testing their Kubernetes-as-a-Service, and the release date is September the 1st, 2018. Over the last 2 years this CSP has introduced or improved a wide variety of cloud primitives like load balancers, cloud firewalls, new types of Spaces and DigitalOcean droplets, new billing schemes, etc. The provider is now in the process of combining all these primitives to provide developer-oriented holistic DigitalOcean Kubernetes service. Will it be a worthy competitor for the Big Three? We'll see in the autumn!

Kubernetes helps to build ubiquitous infrastructure layer to avoid CSP vendor lock-in

AWS is obviously the leader in providing cloud services, as it is mature, functional, feature-rich and cheap. However, once the business becomes too involved with Amazon and starts relying heavily on Amazon-specific offers like DynamoDB, Amazon Kinesis or Amazon Elastic Container Service, moving away becomes nearly impossible. This is where using Kubernetes helps a ton, as Kubernetes allows building a cloud-agnostic infrastructure layer that provides the unified API for provisioning the required services from different cloud providers.

You have the application code running on Kubernetes cluster, which interacts with various cloud-specific services and provides a single Kubernetes API for you. This even allows building the infrastructures that use services from multiple cloud providers as a part of an integral infrastructure, thus ensuring the multi-cloud flexibility for any company.



CSPs are not easily replaceable, obviously. Their services and offers vary greatly, and with time this differentiation will grow only stronger. Using Kubernetes to create a ubiquitous infrastructure layer allows enterprises to benefit from having streamlined access to varying functions and features from various cloud service providers.

Kubernetes enables simple management of distributed systems with Helm

Developers adore scalable and powerful multi-node tools with clear APIs, like AWS RedShift or Google BigQuery. These tools provide extreme efficiency and scalability, yet are a pain to configure. Any cloud architect that had to configure a Kafka cluster or deploy Hadoop can confirm that. Kubernetes comes to the rescue along with the Helm tool.

> **"Helm is a Kubernetes configuration manager that makes it easy to define, manage, update and destroy distributed applications using simple Helm charts."**

Just to make it absolutely clear — Helm enables 1-click installation of complex distributed system like Hadoop or Kafka. Isn't it amazing?

Well, of course, there were step-by-step guides on installing Kafka on AWS, GCP or Azure. However, there are 3 separate guides and they are no simpler than this 10-step guide on Kafka installation on DigitalOcean. Replacing all this mess with 1-click installation is incredibly useful. Helm is inevitably going to become the standard for distributed systems management, just as Kubernetes has become the standard for container management.

Kubernetes: the situation so far and the prospects of 2018

Thus said, as of the first half of 2018, the situation with Kubernetes usage stands as follows:

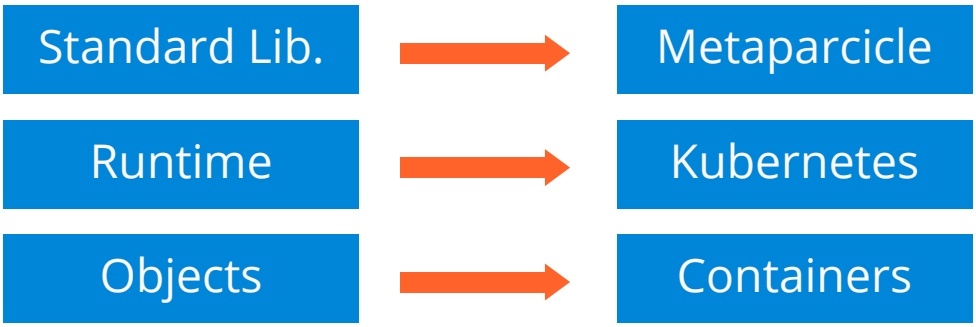
- **Continuity.** Huge communities of open source developers can evolve their products having a common ground and general direction.
- **Uniformity.** Both legacy enterprises and fast-growing startups buy into Kubernetes as a great tool for scaling their cloud infrastructures.
- **Variability.** The Big Three CSPs (AWS, GCP, Azure) provide relatively cheap and feature rich Kubernetes-as-a-Service offers.
- **Predictability.** The vendors of monitoring, security, logging and compliance products have a solid understanding of the systems their products should integrate and interact with, so they can provide even better services to their customers.
- **Flexibility.** Pay-per-use model of billing for using containers in the cloud allows combining the best of two worlds — flexibility of containers and affordability of serverless computing.
- **Competitiveness.** Google Cloud Platform is currently battling AWS and Azure in the terms of serverless container services — they provide Google Kubernetes Engine with their versions of EKS and Fargate, AKS and ACI. AWS and Azure are popular and providing solid Kubernetes-oriented services. This allows them to reap the benefits and further advance their positions.

- **Escalation.** A higher level of problems is soon to appear. Docker made our applications easily portable, but the need for a container management tool became obvious. Kubernetes provides simple container management features, but managing applications spanning thousands of containers across various clouds remained an issue, with Kubernetes Federation still being in alpha. Helm solved this issue and immensely streamlined the management of distributed systems. The new horizon of problems is sure to present itself in the future... and we are sure a solution for these will emerge, opening a new plateau of possibilities!

What will these possibilities include? Using Helm to **start a cross-platform app store** for Kubernetes and simplify the Cassandra or Apache Spark installation to a 1-click package? Managing the software delivery pipelines across multiple environments? Time will show.

Using Kubernetes to advance the cloud evolution

It turns out one of the most important Kubernetes+Helm applications lies within enabling streamlined deployment, configuration and management of complex distributed systems like Kafka, Cassandra, Hadoop or Spark. However, Kubernetes+Helm can do much more than that — open sourcing the development of these products, for example!



Microsoft

kubernetes

Developing cloud-native distributed software is the most complex level of software delivery known today. Only a handful of expert developers worldwide can operate on that level of difficulty, so the advance of those systems is pretty much limited by human capabilities of a group of specialists. The same problem arose in the past and the solution is known — open sourcing the development to leverage the skills of a vast community of developers. However, the skill barrier remains. The solution to it is the Metaparticle project, presented by Brendan Burns on the Kube Con/CloudNativeCon in 2017.

> **"Metaparticle is a standardized cloud-native development library for open sourcing the development of distributed systems."**

Metaparticle is essentially a set of utilities using familiar programming languages, so every developer can simplify the issue to meet their level and contribute to solving the task. This way the developers of any level of skill can participate in developing the most complex software systems known today, in order to collectively shape and speed up the cloud evolution. Metaparticle uses Kubernetes primitives to enable simple synchronization and provides language-independent modules for leader election and module locking in form of easy-to-use abstractions for collaboration.

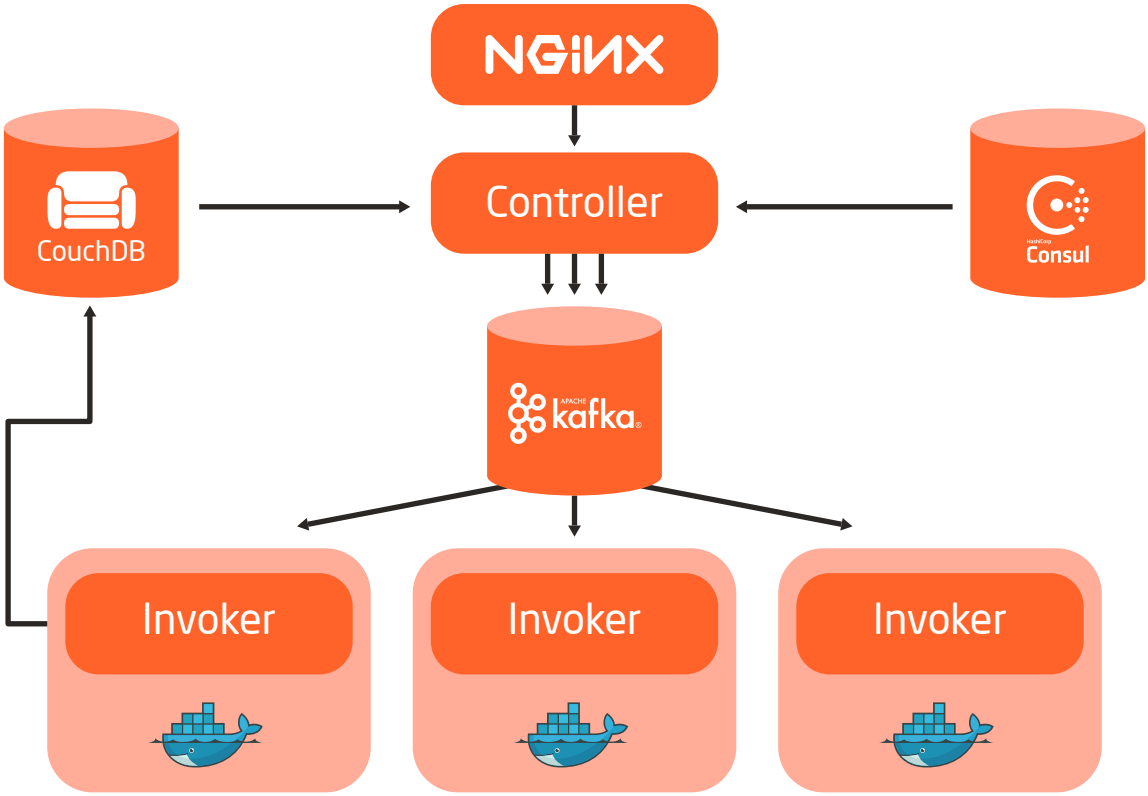
The accumulated experience of distributed systems development shows the necessity of flawless locking of the nodes and establishing the new master, so if the master node crashes the new master is elected and the system work is not interrupted. Kubernetes makes this kind of system orchestration much easier, thus providing a huge boost of efficiency in developing distributed systems.

Kubernetes is the basis of serverless computing

The leading cloud providers like AWS, Azure, GCP and IBM offer serverless computing options — AWS Lambda, Azure Functions, Google Cloud Functions and IBM Functions.

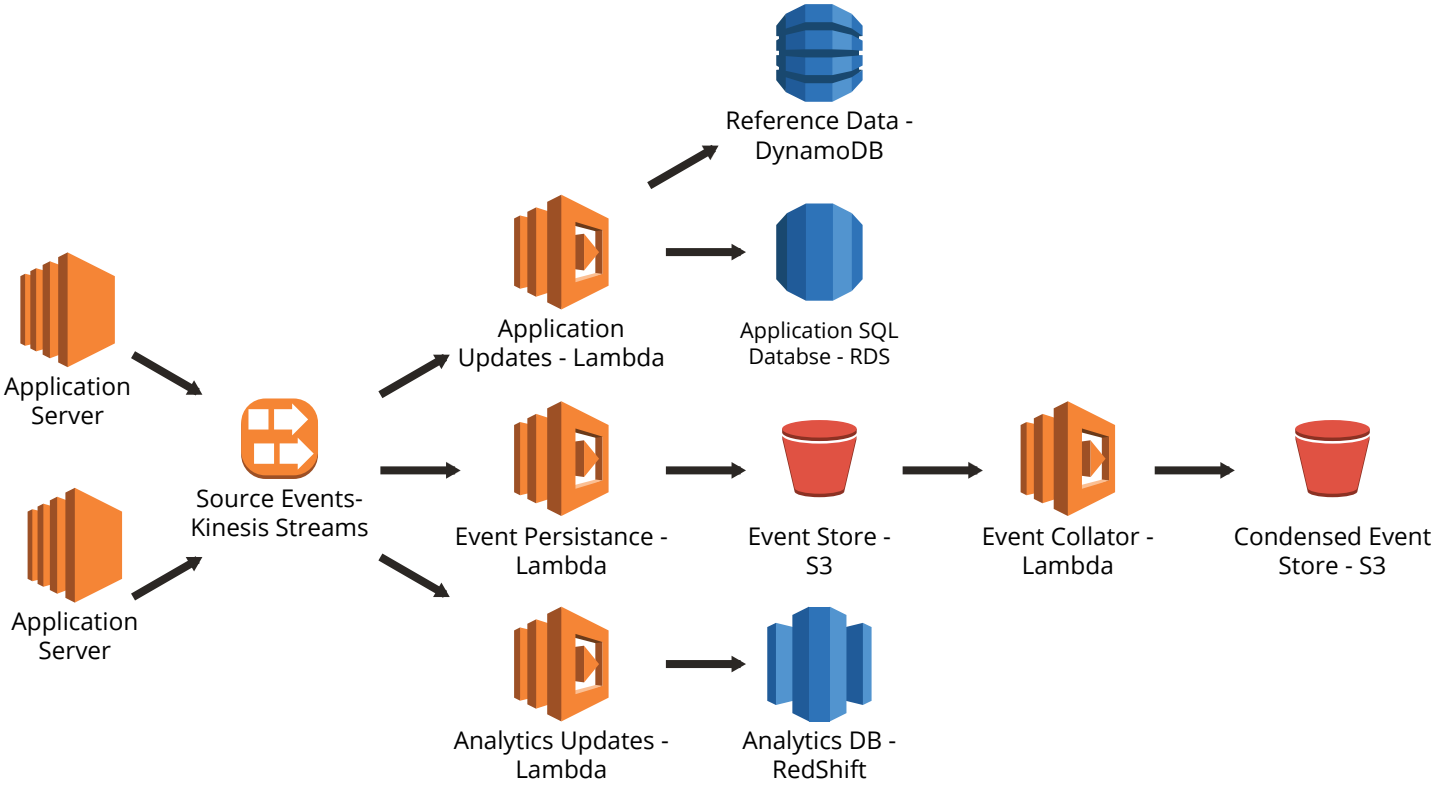
> **“Functions-as-a-Service (FaaS) is the service of providing serverless computing functionality, where your executable code is stored in the service and is invoked directly over the HTTP or triggered by certain system events.”**

You do not pay for idle servers, you do not reserve the instances — once you need the computing capacities, the code is loaded into Docker invokers that spin up the Kubernetes cluster that performs the computations.



Two main use cases for such computing are as follows:

- Easily-scalable computing that grows and operates on demand
- “Glue code” of event-driven workloads with varying frequency, like sourcing the events to a wide array of database consumers.



The best thing is, these solutions are not too complicated. Any decent developer can design and deliver a self-hosted serverless tool for their needs. Just deploy a Kubernetes cluster, configure the Docker invokers, write the functions code and store it, write the web hooks and voila — a self-hosted FaaS tool is ready. Configure a Helm package and sell the software for a one-time $60 payment from the Kubernetes App Store we discussed earlier.

However, such experienced developers are not easy to come by. This is why the tasks like Kubernetes deployment or configuring complex production environments using Kubernetes clusters are best entrusted to reputable and reliable Managed Service Providers, like IT Svit.

Kubernetes in production: real-world use cases by IT Svit

IT Svit is one of the top 15 MSPs worldwide and one of the leaders of the IT outsourcing market in Ukraine, according to international business rating agency Clutch, based in Washington DC.

We house a team of excellent DevOps specialists who have completed more than a hundred projects involving digital transformation, transition to DevOps practices, designing and deploying top-notch cloud infrastructure, configuring smart logging and monitoring solutions, as well as increasing the performance and reliability of the existing production environments using Kubernetes clusters. Below are some real-world use cases of how we use Kubernetes in production.

Using Kubernetes to automate the app deployment

We helped our customer to dockerize all the app and infrastructure components and move them to AWS. Using Kubernetes and Terraform manifests the task of launching a new app instance was reduced to 1 command in -kube-aws CLI tool. This **shortened the app deployment time** from days to minutes.

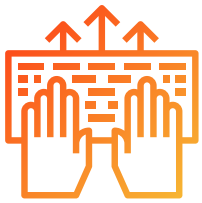
Using Kubernetes to provide the common development environment

Another IT Svit customer faced a common problem of distributed development team, where each developer used their own development and testing environment. The code worked well on their computers, but unexpected bugs were appearing far too often on the release stage. By using Kubernetes to dockerize all the application components and spin up the common testing environment on AWS for all the members of the development team, this issue was solved. This **shortened the development time by 30%** on average and helped ensure a stable release schedule.

Using Kubernetes to automate the infrastructure management

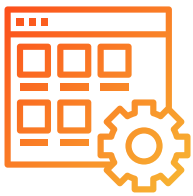
A customer of ours used Rancher to manage their AWS infrastructure and faced multiple irritating issues. By using Kubernetes we helped them automate a multitude of routine tasks, and created reliable CI/CD pipelines using Jenkins, which works best with Kubernetes. As a result, the customer got a **reliable and flexible infrastructure** without the room for human errors, as the operations were done according to Kubernetes manifests, not manually.

App deployment automation



Shortened the app deployment time from days to minutes

Common development environment



Shortened the development time by 30%

Infrastructure management automation

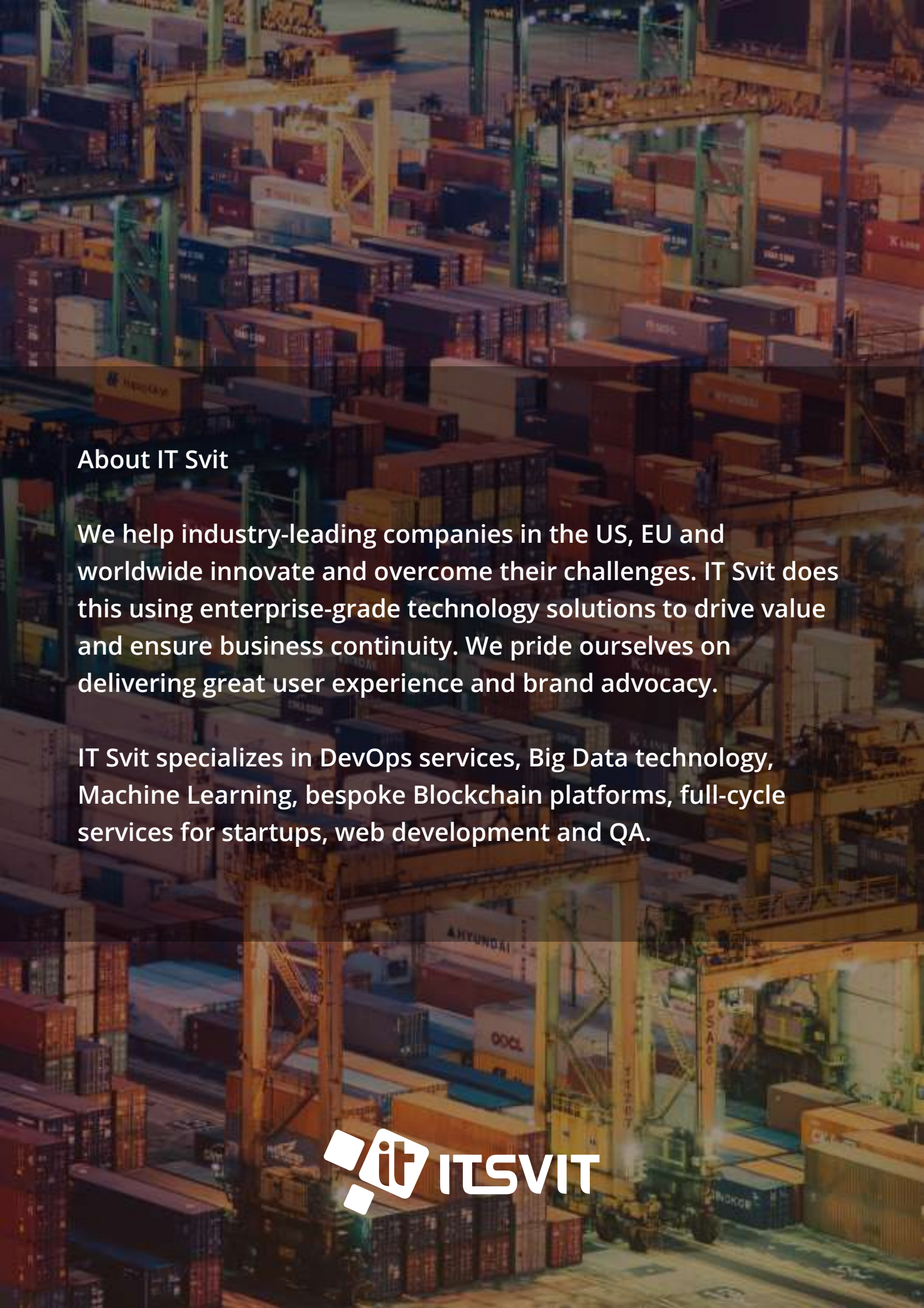
Relaible and flexible infrastructure without the room for error

Conclusions: Kubernetes is indeed the cornerstone of modern cloud infrastructure

Through the course of this whitepaper we showed why Kubernetes has become so important among DevOps tools. By now you know the following:

- top 5 reasons for Kubernetes to become so immensely popular;
- the state of affairs with using Kubernetes as of the first half of 2018;
- the prospects of Kubernetes application for further cloud evolvement;
- the ability to use Kubernetes in distributed systems development;
- the real-world use cases of IT Svit projects where IT Svit used Kubernetes to help our customers increase the efficiency and reliability of their IT infrastructure;

Should you also like to benefit from partnership with IT Svit — you are welcome to contact us and we will help increase the performance and profitability of your business!



About IT Svit

We help industry-leading companies in the US, EU and worldwide innovate and overcome their challenges. IT Svit does this using enterprise-grade technology solutions to drive value and ensure business continuity. We pride ourselves on delivering great user experience and brand advocacy.

IT Svit specializes in DevOps services, Big Data technology, Machine Learning, bespoke Blockchain platforms, full-cycle services for startups, web development and QA.

ITSVIT