CONTAINER ORCHESTRATION: COMMON ENTERPRISE USE CASES

Three Business Drivers for Open Source Container Management

APPRENDA WHITE PAPER

APPRENDA WHITE PAPER

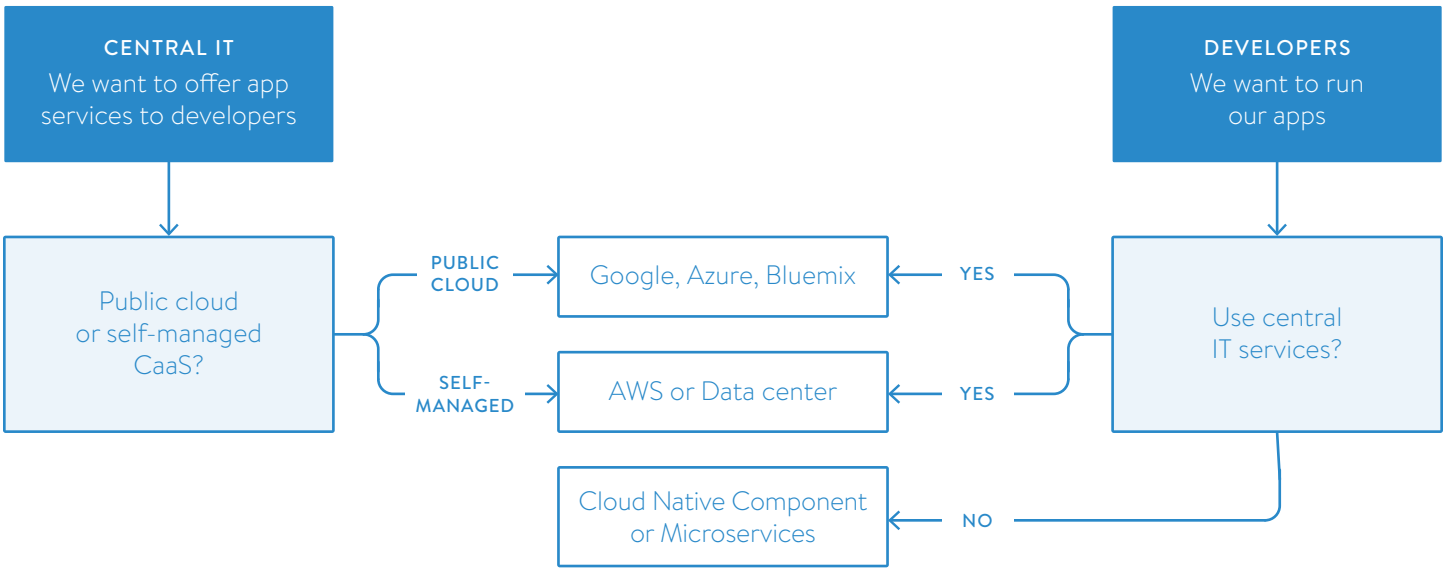
CONTAINER ORCHESTRATION: COMMON ENTERPRISE USE CASES

Three Business Drivers for Open Source Container Management

SUMMARY

As Docker, containerd, and rkt containers become a standard tool for developers, organizations need to find a container management solution that developers can use to build cloud native apps locally yet can scale to become a global cluster managed by operations.

This paper explores the business drivers behind popular open source container management solutions - Kubernetes, Swarm, Mesos and, sometimes, Cloud Foundry.



USE CASES BY POPULARITY IN MARKET

- *Cloud Native Application Component or Microservices Architecture*
- *Containers as a Service (CaaS)*
- *Cluster for Big Data, Analytics, COTS Frameworks*

EXECUTIVE TAKEAWAYS

Here is what you need to know about each use case:

Lead with Developers

Individual application containerization projects, wherein developers use container orchestration as underlying architectural component for apps, is by far the most common and successful use case - ie: "Cloud Native Component" above. Organizations should find and enable groups already interested or working on containerization projects.

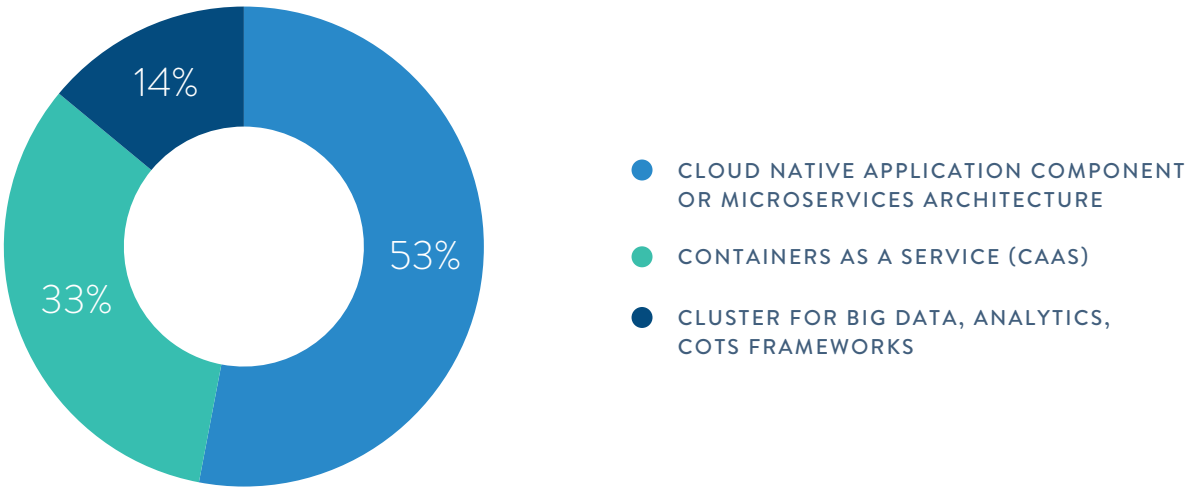
Operations Needs to Partner with Developers

Containers as a Service (CaaS) is the second most common use case but most of these projects are in early stages. Delivering a shared service to developers, whether building the service or using public cloud providers, is a large, transformative, project. Organizations with the highest success rates have system engineers, operations and developers working together closely.

Replace DIY PaaS with CaaS

Most organizations that were attempting to build a shared service using open source PaaS frameworks, or buying one with vendor software, have since graduated to more modern container orchestration solutions. The starting point for building a new application is never "Let me spin up 35 virtual machines," and neither should that be the starting point of CaaS. This will only leave software engineers puzzled and trying to find root causes among different components when problems arise.

Straw Poll on Production Use Case Market Size



KUBERNETES VS. DOCKER SWARM VS. MESOS VS. CLOUD FOUNDRY

Apprenda's S.W.O.T. (Strength, Weaknesses, Opportunity and Threat) analysis of container orchestration solutions can be found here.

The Apprenda research that led us to Kubernetes can be summarized with the three main differentiators:

OPEN SOURCE

Benefit from the speed of true community open source with over 1,000 collaborators that continue to commit hundreds of millions of dollars in development time

BATTLE TESTED

Secure peace of mind with a solution that became production ready from the collective experiences of industry leaders hosting mission critical workloads

MARKET LEADER

Join hundreds of CIOs and technologists that transformed application portfolios into microservices powered by the clear market leader in orchestration

For more information on why Apprenda believes Kubernetes is right choice for containerization, microservices and cloud native projects, see Why Kubernetes.

CLOUD NATIVE APPLICATION COMPONENTS OR MICROSERVICES ARCHITECTURE

- #1 use case for container orchestration projects
- Gives developers control of application architecture
- Majority are hosted on public cloud - AWS, Google Container Engine, Azure, etc.
- Often, there are multiple independent projects within a single organization

The primary container orchestration project seen among organizations comes from teams of developers working with operations to replatform an application. The developers are either building a modern distributed application using containers, which necessitates using orchestration solutions, or they are re-architecting an existing application into numerous microservices. Oftentimes, there are many such independent projects occurring within a single organization. Simply put: they are not centralized and using a shared container orchestration service (Container as a Service - CaaS).

Developers often start researching orchestration solutions when they're just starting with containers. There are many solutions, such as Minikube and Docker Swarm, that simplify running container orchestration on laptops. Organizations can also use managed container orchestration solutions in the public cloud.

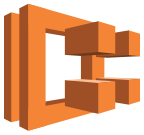
LIST OF POPULAR MANAGED CONTAINER ORCHESTRATION SERVICES:

GOOGLE

Google Container Engine (GKE)

BLUEMIX

Bluemix Container Services



AMAZON

Elastic Container Service (ECS)

MICROSOFT AZURE

Azure Container Service (ACS)

The success rate for this use case is higher than IT transformation projects, such as big CaaS solutions. For example, since software engineers are responsible for the architecture, they're required to dig into a new black box built by operations. This, unfortunately, transforms debugging from an identifiable source into something more closely resembling a murder mystery.

Some organizations with enough applications that utilize container orchestration may decide to centralize onto single cluster.

CONTAINERS AS A SERVICE (CAAS)

- #2 use case among container orchestration projects
- Operations teams wishing to manage or build a shared, self-service, container orchestration cluster for developers should start small to ensure success
- Delivering a self-service production cluster often requires regulatory, security controls, integrations, etc. Public cloud container services manages the cluster fabric but still often requires ancillary capabilities to deliver as production services to developers
- **Decisions:** To self service, or not to self service? Managed cluster or home grown? If building your own, where will you host it?

Most CTOs want greater agility and are looking to use Container as a Service (CaaS) to fulfill this promise. Before the advent of docker containers and container management, a few large organizations would build their own self-service application platforms from open-source PaaS frameworks. Most organizations that took this route (and new ones looking to do the same) are actively researching next-generation container orchestrations engines as the basis for these efforts.

Delivering a CaaS, and the effort that goes into it, should not be underestimated. It will take dedicated work to integrate existing systems and, possibly, full-time developers to close the technical gaps. Most large organizations will want to use a hybrid of managed container organizations clusters (as listed in the last section) and self-managed solutions. Note that private/public cloud does not really make sense here because the self-managed solution can also be on a public IaaS. Indeed, most container orchestration solutions are.

Not all container solutions need to be self service. There will be a number of organizations that want to keep control of production environments in the hands of IT. In these cases, the container orchestration and management solution becomes similar to classical VMware tools, but more efficient and not as coupled to infrastructure.

Operations will spend less money on virtualization, middleware, and operating systems, which are savings and efficiency not achieved through the use of virtual machines. Beyond the savings, IT will need to support container orchestration soon. That age is coming quickly, so if they are not able to transition to a self-service model soon, this is a great option.

Beyond the savings, IT will need to **support container orchestration** soon. That age is coming quickly...

CLUSTER FOR BIG DATA, ANALYTICS, COTS FRAMEWORKS

- Quickly maturing use case
- Can host more than stateless web applications in container orchestration solutions
- 70% that try to achieve this use case while simultaneously funding a CaaS project will fail to get the same desirable results

Hosting off-the-shelf applications and purpose-built container orchestration solutions is still in its early years, but it is a highly desirable future state. This use case will prove to have the longest path to success because the organization will need to have both a production cluster and a viable solution using that cluster. The maturity of this use case differs among the container orchestration options. Mesos and Kubernetes are the most mature. Mesos has the concept of "frameworks" and several popular Big Data tools such as Spark. Kubernetes has the concept of Statefull Sets, which supports broad numbers of analytics and COTS solutions.

Organizations that are trying to set up analytics and other solutions hosted on container orchestration should either finish the CaaS project first or concentrate only on delivering a single solution (e.g. Cassandra). Those that try to "boil the ocean" often regret the decision and rarely see the desired results.

DISCLOSURES

After Apprenda conducted extensive research on all container orchestration solutions, it chose to back Kubernetes and offer enterprise services, support, and ancillary features for the pure open source version of that orchestration solution. Beyond the preference for Kubernetes, Apprenda does not have strong preferences on use cases or whether Kubernetes is used as a public CaaS (like Google Container Engine - GKE or Azure Container Service) or rolled out by developer teams on AWS or elsewhere.

Apprenda Inc.
433 River Street
Troy NY 12180

WWW.APPRENDA.COM