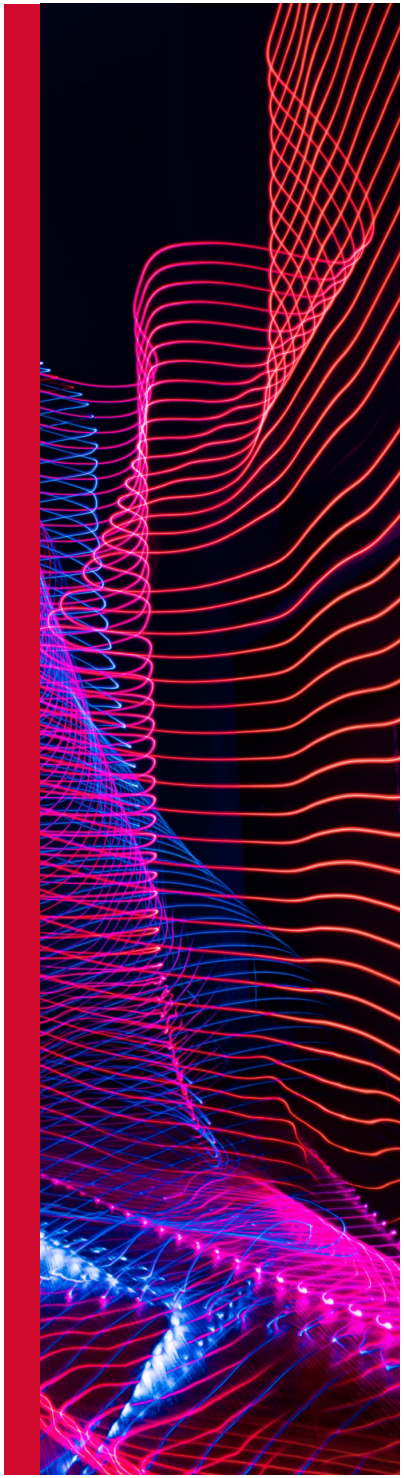
responsiv
simple · effective · distinctive

POINT OF VIEW

Containerisation is the New Normal



Introduction

Containerisation is the IT industry's response to the need for workloads to move across public and private clouds, and between vendors.

The motivation is to achieve **hyper-convergence** of workloads, **prevent lock-in**, and to facilitate **right-placement** of work to achieve cost, performance, security, or capacity goals.

Encapsulation of assemblies of products and configuration provides a highly standardised way to transport, store, and manage functional units. The container concept is not new to IT, the principles of decoupling, isolation, separation of concerns, and encapsulation all point to the need for a way to hide implementation details; containers do all of this.

Build Once Deploy Many

The container concept allows functionality to be built, including installation of products, and deployed to different platforms and places. This is perfect when building for cloud, and Responsiv use similar concepts to support DevOps and remote customer development.

The container does not contain a full operating system and has no hardware abstraction layer, which means it does have dependencies on the underlying (host) operating system and hardware architecture.

A Short History of Containers

Containers of the steel type were invented in 1956 and were almost immediately successful. Within a few years the new normal delivered a scale of packing loads that is unimaginable compared to the previous techniques.

> Today's global marketplace has been made possible by containerization. The first shipping container was invented and patented in 1956 by Malcolm McLean. Between 1965 and 1970 the capital locked up per tonne of inventory between Hamburg to Sydney fell by half. (Economist)

Companies that did not embrace the new container-technology were marginalised and, in many cases, put out of business.

Traditional dock workers were no longer needed, new container-handling skills were in demand, and major investments into standard container-moving cranes and equipment at ports and in new ships was needed

> The cost was considerable.

Justifying a complete change to the way goods were packaged, transferred, secured, and transported must have been incredibly difficult. Imagine walking into a board room and announcing that you were going to change the way every company, port, ship, lorry, and train moved goods; that your way was going to be substantially adopted within 10 years.

Containerisation in the IT Industry

When steel containers were introduced, they had obvious benefits, including:

- Cargo loading cost went from $ 5.86 per tonne to $ 0.16 per tonne
- Cargo loading efficiency went from 1.3 tonnes per hour to 10,000 tonnes per hour
- Goods are private, tracked, and the handling equipment is highly standardised
- Containers are customs checked and sealed on point of EXIT
- Border crossing as frictionless and secure. Paperwork minimised, and goods protected

All these benefits are relevant to the IT industry, and especially when we consider them in the context of Hybrid Cloud.

Hybrid Cloud

In a Hybrid Cloud, multiple clouds work together, coordinated by a cloud broker that federates data, applications, user identity, security and other details. A hybrid cloud can be delivered by a federated cloud provider and has the capability to combine its own resources with those of other providers. The provider of the hybrid cloud must manage the cloud resources based on consumer requirements.

> As with the shipping industry, adoption is disruptive and expensive.

The concept of a hybrid cloud is simple. The challenge is to build a unified platform across multiple locations and vendors, and at the same time deliver hyper-convergence, and robust operations.

Containerisation

One answer is containerisation. The cloud-broker becomes a distributed container system, and the hyper-convergence delivered by allowing many containers to share resources, but be moved and balanced to deliver the required performance.

- **Containerisation** is a lightweight alternative to full machine virtualisation that encapsulates an application in a container (cloud-foundry, Docker, Rocket, and Open Container Initiative) with its own operating environment. In a way that can be run on a containerisation system.
- **Containerisation systems** such as Docker, Rocket, Warden, and Windows Containers have emerged as an alternative way to install and run applications on servers. A container system is used to define and package a runtime environment for a process into a **container image.**
- **Distributed Containerisation systems** such as Docker Swarm and Kubernetes provide a way to build a containerisation solution that spans clouds and private servers. Products, including OpenShift and IBM Cloud Private are built on top of Kubernetes.

Like physical containers, IT containers facilitate fast loading, unloading, and movement of content between hosts, and across clouds.

- Workload loading cost and time are reduced
- Software is private, tracked, and the handling equipment is highly standardised

Import to company infrastructure is frictionless and secure. Paperwork minimised.

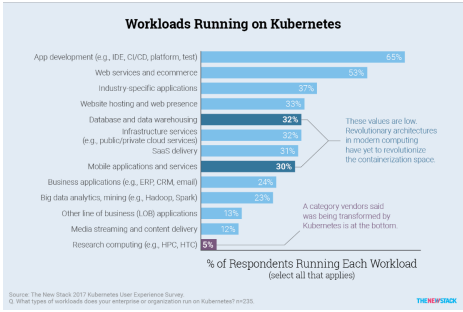


Figure 1; Kubernetes Adoption

Containerisation Is Not For Everyone

Steel containers are rarely used by local fishermen to transport their catch. They have their place, and so do their IT counterparts.

To provide for dynamic relocation and capacity scaling of workload it is necessary for containers to prohibit changes to their state. This includes database data and file system changes, it also prevents access to the local filesystem – otherwise the container cannot be moved across servers.

	Container	Non-Container
Micro Services (stateless)	Yes	
Website (stateless)	Yes	
Web Application	Yes	Yes
Stateless Mediation	Yes	
Compute Engine	Yes	
Database		Yes
Persistent Storage		Yes
File Share		Yes
In Memory Connectivity		Yes
		Yes
HA-CMP, Veritas		Yes

The encapsulated software "thinks" it has a complete machine. In fact, the container does not have an OS kernel it is dependent on compatibility with the host OS (Durban, Berkley, Linux) and hardware (Intel, Power) architecture. Each container system is similar but different; tools are available to convert containers between systems.

IT Containers share resources with other containers on the same host, however they cannot share memory segments and must communicate as-if they were on different physical hosts.

- Once a container is destroyed, the files in the container no longer exist

- Containers do not have access to files or NFS mounts on the host machine. They can be mounted as remote file systems to avoid fixing them to a particular host.
- Containers can map (private) filesystems and OS interfaces to the host operating system

The result is a system that is excellent if your need is right location of workloads to manage latency, data movement, access to functions, and cost of compute, or encapsulated, secured and managed functionality that can be assured at point of creation.

Where containers are not so clever, is when data is involved; for example, when a messaging system with persistence, a file-share, or a database is involved. Databases that are not expected to change can be considered stateless if their state is in the container template or external to the container (in a script).

Skills

Steel containers appear simple, can be kicked, and considered relatively low technology, yet their adoption required new skills in handling, operating new equipment, and tracking. The IT equivalent is no different.

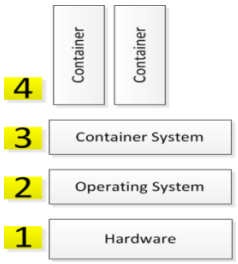
To operate a container system requires new skills and insights that are in addition to the need to understand operating systems and code. The result is a sophisticated administration environment that delivers simplicity to the consumer.

Companies have sprung up to deliver Open Source or Vendor neutral offerings to the market. These include cloud-foundry, Docker, Rocket, and OCI (Open Container Initiative).

Demand from developers, specifically to build DevOps tool chains and test environments, has played a big part in showing the value of containers. They offer unprecedented consistency and portability for testing and shipping modern software applications. Traditionally, you'd spin up a virtual machine to test and deploy applications; these days, containers offer a more lightweight, easy-to-manage option for delivering ready-to-run applications, irrespective of environment.

The Container Eco-System

The containerisation reference model can be considered as a series of layers. The top layer is the container, which operates in a container-system, that relies on an operating system, that runs on a hardware (host) platform. The single host model provides the ability to run several "independent" containers [4] on a container system [3] (Rocket, Docker), deployed onto a Unix operating system [2] running on an intel host [1].



Hardware [1]

Compiled code cannot run on a hardware architecture (Intel, Power) for which it was not built. If the container includes any compiled code, it must be specially designed to operate in more than one architecture.

Operating system [2]

The container depends on the host operating system for kernel activity and filesystem functions. It does provide a level of isolation, however moving a container between Durban, Linux, Berkley and other flavours of Unix can be problematic. The target OS must be considered when building the container (template)

- Windows, Microsoft Hyper-V, RedHat, Centos, Ubuntu, et-al
- Containers on power platforms are not binary compatible with the Intel platform

Note that containers running on Windows are contained inside a Linux kernel virtual machine.

Container System [3]

The container system for a single host is responsible for feeding and watering one of more containers. Each container runs as a process, which is often a sub-process or forked from a controller. In some cases (Docker) there are several layers of these processes.

The next step toward a hybrid cloud is the ability to extend the single host to multiple hosts, and multiple locations (data centres). This is achieved using Docker Swarm, Kubernetes or another **distributed container system.**

Container [4]

A container is a template that can be run inside a container system. There are several flavours of container, including Docker, and the Open Container Initiative (OCI) containers.

Moving to Containers

The appeal of hybrid cloud solutions is that they allow organisations to attain desired business outcomes by combining services and capabilities in a way that promotes agility, is budget friendly, and secure.

Enterprises are moving many of their production applications to containers. The container systems facilitate creation and deployment of containers in production, but lack data management and protection features. To fully embrace the container journey requires persistence, scale, security, availability, and high-performance capabilities.

> **The appeal of hybrid cloud solutions is that they allow organisations to attain business outcomes by combining services and capabilities in a way that promotes agility, is budget friendly, and secure.**

There are three aspects of moving to containers and the hybrid cloud model that may be less obvious: the need for integration, and the organisational impact of such a move. Finally, the migration duration and disruption to business must be properly considered.

Organisational Impact

It is important to recognise that hybrid cloud (container) solutions are not traditional IT projects, and their impact can be felt throughout an organization. Successful organisations have implemented coaching programs to facilitate the change in technology and processes for their stakeholders.

Container And Enterprise Integration

Integration is critical in a successful container implementation. The model will drive significant sharing of infrastructure, and requires thought about how data will be managed and accessed. For the majority of containers, the storage of data must be remote; for others it will be through remotely mounted devices.

Financial allocation of costs and responsibilities must be properly considered. Will the model be based on capacity, compute consumed, or transaction throughput?

How will the enterprise risk be considered? The container model drives hyper-convergence and with-it systemic risk. A single failure may disrupt a significant breadth of functionality that is otherwise not considered to be connected or related.

Such an implementation aggregates capabilities and solutions from cloud service providers, and those hosted on-site in order to leverage the best available combination.

Service Oriented Architecture (SOA) Representational State Transfer (REST), Application Programming Interfaces (APIs), and cloud management and orchestration frameworks have opened up new options for integrating cloud services.

Migration

The migration of a data centre is difficult and full of risk and surprises. When moving to the cloud, or changing to containers in a local DC, it is important to consider how data will be controlled when functions that were once part of a single monolith are now completely separated. How will users access the functions and data, and how will the transition be planned, and controlled.

The State Of Technology

The containerisation landscape is in flux and an exciting place for techno-folks, and unsettling place for businesses and investors.

> The question is: "which technology will be the winner?".

In 2011 cloud-foundry started to develop and introduce the idea of distributed containerisation. By 2013 they were joined by Docker, and the world backed docker containers as the de-facto standard for container technology.

Then along came Rocket (RKT), and the Open Container Initiative in 2015.

The current trend is for clouds to support Docker as an existing standard while moving toward the **Open Container Initiative (OCI)** as the strategic target.

The Docker container and Docker Swarm technology is far from dead, however OCI and Kubernetes has overtaken it in terms of volume and vendor support.

The technology landscape for containerisation, virtualisation, and development of applications to run on these environments – is simply huge.

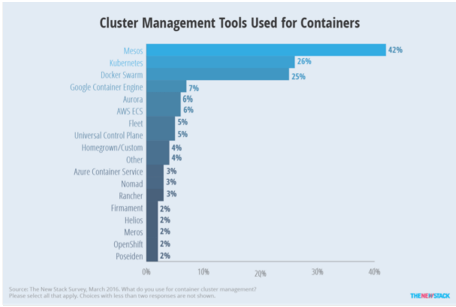


Figure 2; Products that deliver Container management across clouds

The illustration above is a sample of the different products available to deliver hosted hardware (clouds), solve provisioning, runtime, container orchestration and management, and provide environments for development testing and release of applications.

Point Of View

It is likely that not planning for a journey toward using containers in the enterprise will lead to problems. The industry is moving in that direction, and new versions of products, new features, and available skills will drift toward that future.

Buying something that offers benefits that you do not need, or that do not return the investment is never good business.

Containers offer high value when applied to appropriate problems. They deliver new options for deployment, make possible new models for architectures, and solve specific problems that have been expensive to address.

There are periphery concerns and costs that should not be overlooked, for example the container system, development and operational processes (DevOps), and skills. As we have mentioned, even the steel containers required significant changes to the surrounding infrastructure, including in some cases widening railway bridges.

They are not the answer to all problems. It is a mistake to believe that containers are the answer to all problems, or that they will be appropriate to all situations. We need to understand, for each situation, what the contribution will be from using containers, as for any other component they need to pay their way.

- Standardisation of hyper-converged runtime platforms
- Company skills
- Specific problem types where state is not the primary concern (websites, development, compute applications)

If you are considering a move to containers for any reason, then we will be happy to come and discuss some of our research and experience. As always, we need to look at things to ensure we are doing the right things in the right place.

Conclusion And Next Steps

If you decide to give it a go, then we will start with a discovery phase. This involves reviewing your systems and their suitability for containerisation or conversion to micro-services. As part of this activity, we will identify two or three that can be delivered quickly and that will demonstrate value.

Following discovery, we will review with you how you want to proceed, whether this technology will deliver value, and if so, plan the transition.