A National View of Containers and Kubernetes in Research

Shared challenges and opportunities

Steven Manos[1], Kieran Spear[2], Paula A. Martinez[3] , Steve Quenette[4] , Mark Gray[5], Andrew White[6], ARCOS Working Group[7]

[1]University of Melbourne eResearch Centre, Melbourne, Australia
[2,6]Australian Research Data Commons, Melbourne, Australia
[3]National Imaging Facility (NIF), University of Queensland, Brisbane, Australia
[4]Monash University Monash eResearch Centre, Melbourne, Australia
[5]Pawsey Supercomputing Centre, Perth, Australia
[7]See Appendix C for list of members and affiliations.

1. Context and methodology

This report has been developed through the *ARDC Community of Practice* - Australian Research Container Orchestration Service (ARCOS), an initiative[1] which was launched at the national Kubernetes workshop at the Pawsey Supercomputing Centre on 11 March 2020.

This report summarises the findings and challenges expressed through engagements and a survey conducted with various stakeholders, including: Australian Research Data Commons (ARDC), AARNet, Australian BioCommons, Australian Characterisation Commons at Scale (ACCS), National Imaging Facility (NIF), Monash University, CSIRO, Pawsey, TERN and the EcoCommons, amongst others.

2. Containers 101

Containers provide a lightweight, isolated execution environment for software and applications. Container images package up precompiled software with all of its dependencies into a single bundle that is isolated from any environment that runs it. This provides a way to share libraries, code and configuration amongst researchers - potentially avoiding the complexities of operating systems, libraries, code compilation, and so on.

There is diversity in application(s) that are containerised, and these are realised in two ways:

1. **Containerised toolset** - Where a whole toolset or pipeline is included in a single container.
2. **Containerised application** - Where a single application or function is built into a container (sometimes called a 'microservice').

The pros and cons of each of these approaches present themselves in different ways. For example, microservices can be linked together to build a pipeline, and new versions of a tool incorporated as needed. On the other hand, a toolset could present a complex pipeline of applications in a single ready to use container. Container best practices suggests that each container contains a single process or application.

2.1. Docker and Singularity

For building and running containers, different container image formats and container runtimes (the container execution environment) are available. The two key types of container formats used are Docker and Singularity, with Singularity increasingly used in research computing and HPC environments[2]. Docker uses a standard runtime called containerd, while Singularity has its own runtime.

Singularity's security posture targets multi-tenanted environments such as HPC. A Docker daemon runs with admin privilege and a running container has the ability to make changes to the host

[1] Platforms EOI: Establishing a core services for kubernetes and kubernetes federations, 19 September 2019. https://staging.ardc.edu.au/wp-content/uploads/2019/09/Establishing-a-core-services-for-kubernetes-and-kubernetes-federations.pdf
[2] https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0177459

machine via file system mounts. This means that access to a Docker daemon effectively grants admin privilege to the user interacting with it. In contrast, following some privileged setup steps, Singularity runs the container process with the same privilege as the user.

Docker's security posture targets web-scale cloud native computing. In this scenario the focus is on a rapid DevOps (development to production deployment) cycle for an increasingly larger set of smaller units of code, and the unlimited scaling out of instances of these units of work. Data access is expected to be shared across an application of many containers, and hence the orchestration instance (i.e. Kubernetes cluster instance) is the bounds of information assurance.

A third posture is emerging, of data processing workloads (and hence containers) being pushed out from clouds, to the edge devices. This occurs for both privacy and scalability reasons. Whereas the former two postures have well established research communities and production tools, and hence in need of operational maturation, the edge posture is very much at the exploratory phase.

3. Containers and Kubernetes in research

In order to manage containers efficiently, systems and tools are needed that can configure, run and manage them throughout their lifecycle. Container orchestration considers the deployment, management, maintenance, scaling, scheduling, and monitoring of containers.

Kubernetes[3] (also referred to as 'K8s') is one such orchestrator which provides a standard declarative model for describing container deployments. Today, it is the defacto standard for container management, and supported across various private and community cloud environments, HPC centres and commercial clouds such as Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform (GCP).

Even with this promise and standards, there are complexities. Differences in how the compute, storage and networking abstractions are implemented by the underlying platform means that not all Kubernetes deployments are functionally equivalent nor compatible. For example, no two instances of Kubernetes across the Research Cloud, AWS, Azure, GCP or HPC centres are guaranteed to be the same from a research application's lens. Yet research groups increasingly need their toolset to readily port to where their colleagues have resources[4]. As a result, the deployment and ongoing management of Kubernetes clusters for national research projects is challenging.

3.1. Drivers

Stakeholders in this initiative were requested to outline their current and expected future use cases around their use of containers. The results are summarised in Section 4 and Appendix A.

Based on the use cases, the following were identified as the key drivers of container uptake in research infrastructure:

[3] https://stackoverflow.blog/2020/05/29/why-kubernetes-getting-so-popular/

[4] Quenette. (2019, December 5). Global experiences digital research infrastructure federations. Zenodo. http://doi.org/10.5281/zenodo.3563246

Driver 1: Minimise infrastructure complexity through Isolated software environments	Software inside a container has only its own binaries and its dependencies - and multiple applications can be run on the same system without regard to system-wide conflicting dependencies
Driver 2: Portability and easier code distribution	Code and data can be wrapped up in a single distributable package and be deployed elsewhere
Driver 3: Containers support reproducibility and reusability	Code, library and data coherence is guaranteed through packaging the materials into a container.
Driver 4: Simplification of underlying compute infrastructure	Infrastructure just needs to run containers, without being tightly coupled to a particular application running inside them.

Table 1. What are the Drivers to use containers in a research environment?

3.2. The Kubernetes advantage

Globally over the last five years, Kubernetes[5] has emerged as a standard in commercial and research software infrastructure at scale. Our finding is that using Kubernetes provides a common infrastructure stack and framework that allows for separation of concerns between:

- **Infrastructure team** (deploy/manage Kubernetes underlying infrastructure)
- **Application developers** (research software engineers (RSEs) who write apps and/or containerise applications)
- **Application users** (researchers who run containers on container infrastructure)

We refer to this as the 'Kubernetes community framework'.

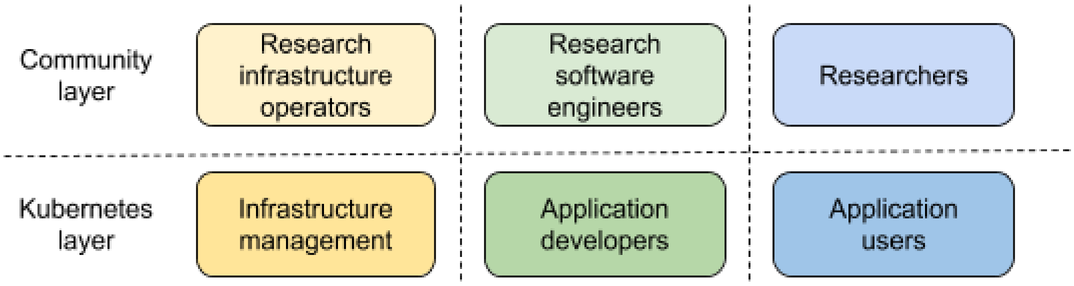

Figure 1. Kubernetes community framework

The result of this separation is that little application-level knowledge is required of an infrastructure team, little infrastructure knowledge is required of app developers, and no knowledge of the complexities of libraries and software compilation are required of the users.

[5] Kubernetes open-source system for automating deployment, scaling, and management of containerized applications - https://kubernetes.io/

The implications of this separation is profound as each concern relates to a distinct community, each one can be scaled independently, and each one can have its own practices and strategies applied independently of others. For example, containerised tools can be shared globally, whereas expertise on infrastructure management can be shared in a local context.

4. Kubernetes - use and challenges

4.1. Survey summary

As part of this initiative, community members were surveyed to explore the areas of Services, Staffing, Community, Containerisation, and Container Orchestration. The survey questions can be found in Appendix B. The key findings of the survey shown in the following table.

Table 2. Key survey findings and its category descriptions

Key survey findings	
Services	
Services provided	Most common service was using containers to host user applications (either multi-tenant or dedicated instances). Other services provided include running user containers (using something other than Kubernetes), Kubernetes clusters as a service, supporting users to create their own clusters, and providing container images.
How do users request services	Web interface, command line interface, helpdesk tickets.
Current status with containers	Most organisations are using containers in some form and about half of those use Kubernetes to orchestrate them.
Next goals	Putting container workloads in production, testing out Kubernetes, putting Kubernetes into production, growing existing Kubernetes clusters.
Staffing	
Current staff time	Most projects or organisations surveyed have assigned less than 1 FTE into containers and K8s initiatives. There are a few cases of greater effort (>3 FTE), which also correspond to larger deployments and more advanced usage of K8s.
Skill level - Configuration & deployment	Kubernetes can be used to deploy software infrastructure for improved scalability and sophisticated networking. Most organisations using Kubernetes don’t leverage these capabilities. This may be due to limited staff capability and human resource in these organisations.

Skill level - Operations	There is a wide spread of effort, capability and maturity.
Aspirations without current constraints	There is a shared need for 'trusted' container repositories, possibly at the discipline level. Trusted has two distinct interpretations here - (1) trusted by operators & institutions to be secure and legitimate, and (2) trusted by users to run as expected.
Community	
Upstream science/technology communities	Sciences: Characterisation, Medical Imaging, Transport, Ecology, Neurosciences, Geophysics, Microscopy, Spectroscopy, Bioinformatics, Radio astronomy, Fluid Dynamics, Data science. Technologies: Singularity, Python, R, Docker, Prometheus, Traefik, Go
Other partners	ANSTO, Auscope, BioCommons, CERN, Characterisation Virtual Lab (CVL), EcoCommons, Griffith University, MASSIVE, Microscopy Australia, Monash University, MyTardis, NCI, National Imaging Facility (NIF), CSIRO, Pawsey, TERN, UQ, USyd, UWA, ownCloud.
Other recommended resources	Kubernetes guides and deployment resources. Repositories/catalogues of containers.
Containerisation	
Build	Building containers for specific environments (ie: HPC) presents challenges. Standard practice is needed, especially to ensure security in container use.
Images location	Most organisations are using Docker Hub or a local registry for storing images. With recent changes to Docker Hub's free offering putting limits on storage and downloads, finding a home for these images may be more difficult.
Discovery mechanisms	Local (bespoke) builds, Public repositories
Pain points for migration	Limitations in compute environments and integration with storage backends
Current Workload locations	Public research clouds (Nectar, Pawsey Nimbus, others) Commercial Cloud (AWS, Other) Public HPC
Future Workload locations	Increasing. Some organisations have adopted containerisation across the board, others are interested in exploring expanded use of containers.
Public cloud motivations	Extra capacity, stability, features, performance

Container Orchestration	
Motivations/Use cases	Two key use cases became evident through the consultation: 1. I operate a service/platform/infrastructure, and I want to deploy Kubernetes upon which to run that platform (the user engages with the end platform/application) 2. I am a user with a container or set of containers, and I want to run/deploy those containers on a managed Kubernetes platform (the user engaged with the Kubernetes cluster APIs directly)
Cluster deployment tools	There is a large diversity of tools to run K8s, with no single 'best practice' set of tools evident. Various tools are used to deploy Kubernetes, including Kubespray, Magnum (OpenStack), Rancher and Kops. Some survey responses gave multiple answers for choice of deployment tool as they are still in the exploration stage of Kubernetes usage.
Container technology	Many organisations are also using Singularity to provide containers to users.
Useful features	Whereas the value of containers is clear - in particular the benefits of portability and isolation of software - to some respondents the value and benefits of operating Kubernetes is not as clear.
Alternative technologies	There are few widely used alternatives to Kubernetes for container orchestration. The most common alternative may be Docker Swarm, intended for use with Docker based container deployments. Commercial cloud services offer service-based container orchestration that may be explicitly based on Kubernetes and provide improved ease of use, or be based on internal technology stacks.
Scale	Few users are really running stuff at scale, in production.
Effort, capability and maturity	Many users are already running containers, though most without orchestration. Some HPC workloads.
Deployment diversity	There is a large diversity of tools to run K8s, with no single 'best practice' set of tools evident. Various tools are used to deploy Kubernetes, including Kubespray, Magnum (OpenStack), Rancher and Kops. Some survey responses gave multiple answers for choice of deployment tool as they are still in the exploration stage of Kubernetes usage.

4.2. Community challenges

Engagement with the community raised a variety of challenges summarised below.

Challenge 1: **Expertise, experience and practices are isolated and disconnected.**	Most deployments of K8s are self managed - i.e. the K8s platform and tools are deployed and managed by administrators in that organisation. The only exception are some cases where the ARDC Cloud Kubernetes platform (Magnum) is utilised. The outcomes of this are twofold - (1) That they have excellent practices but there is no way to make it available to anyone else, and (2) They utilise sub-optimal practices, are struggling, etc. - and have no way to access expertise elsewhere.
Challenge 2: **Diverse deployments are leading to incompatible islands**	There are differences in K8s platform configurations and this limits the transferability of containers from one deployment to another.
Challenge 3: **Kubernetes is complex**	K8s is inherently complex, difficult to adopt and use and current implementations are not overcoming this difficulty for users.
Challenge 4: **Kubernetes is feature rich and extremely capable**	K8s is feature rich, and there is currently not a lot of exploitation of the advantages offered by K8s. Many people are building and running containers, but deep intimate understanding and use of K8s is rare.
Challenge 5: **FAIR containers are rare**	The metadata and provenance of containers is important but has not been addressed. The view is that the FAIR (Findable, Accessible, Interoperable, Reusable) principles applied to data need to be addressed for containers.

Table 3. Summary of Key Community Challenges

4.3. Australia's Opportunity

The adoption of Kubernetes is still embryonic and is being investigated and developed by many research infrastructure groups in Australia. Without a coordinated effort there is a risk of the current initiative losing momentum as a consequence of the challenges listed above. This is likely to have a number of impacts including:

- Some groups are likely to abandon its use due to the difficulty of implementation and providing support,
- The variety of implementations are likely to continue to proliferate which will further exacerbate the issues of non-standardisation, a lack of trust between implementations and increase support efforts to a level which are not sustainable by the groups,
- Efficiencies of scale and resulting cost savings are not likely to be realised.

There is an opportunity for Australia in that "coordinated Kubernetes" is already pervasive within the Australian research community either as a pilot and/or production. Kubernetes almost uniquely caters to the ARDC Platforms and Software strategy to "Enable transformative, interdisciplinary, collaborative, reproducible research through the use of advanced software and platforms."[6]

[6] ARDC Software and Platform Strategy, 25 July 2019, https://ardc.edu.au/wp-content/uploads/2019/07/Software-and-Platforms-Strategy-Summary.pdf

5. Recommendations

In order to realise the benefits of Kubernetes and for it to be utilised to its full potential, additional resourcing, impetus and coordination is required at a national level. Four recommendations are detailed here, aligned with the Kubernetes community framework (Figure 1).

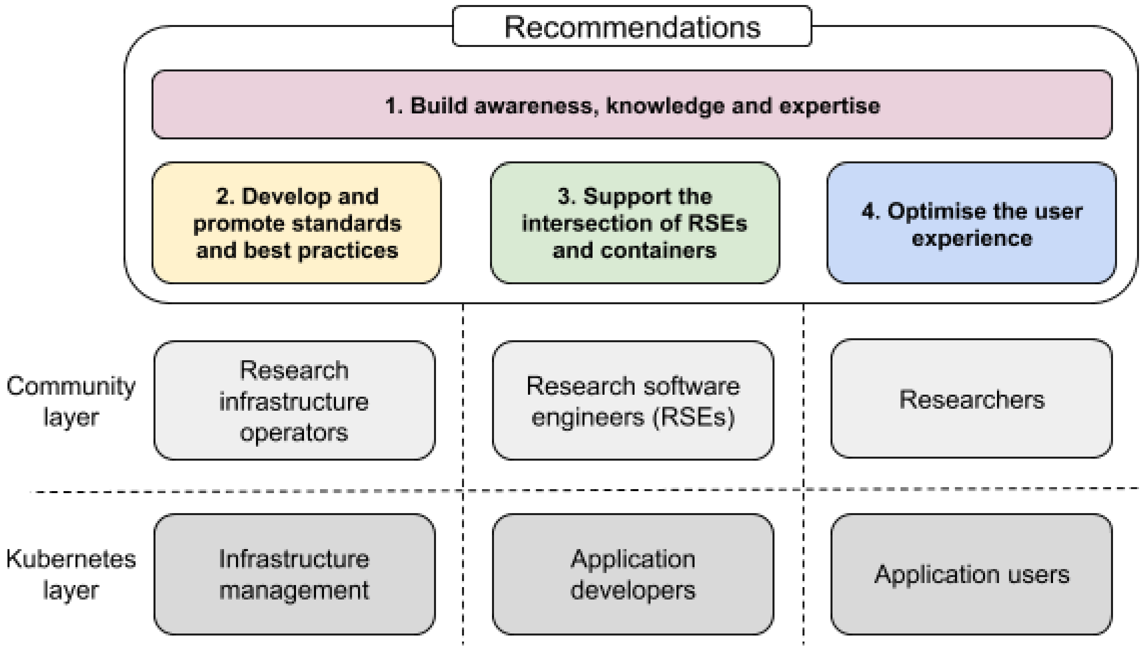

Figure 2. Recommendations aligned with the Kubernetes community framework.

The recommendations should be interpreted in the context of the two key use cases that emerged from the consultations to date, namely:

Use case 1: *I operate a service/platform/infrastructure, and I want to deploy Kubernetes upon which to run that platform (the user engages with the end platform/application)*

Use case 2: *I am a user with a container or set of containers, and I want to run/deploy those containers on a managed Kubernetes platform (the user engaged with the Kubernetes cluster APIs directly)*

The recommendations are outlined in detail in the following sections, setting out a series of next steps in response to the community challenges outlined in Section 4.2.

5.1. Recommendation 1: Community building, best practises and expertise

R1.1 Align efforts nationally

Many organisations are already utilising containers and Kubernetes, and huge benefits can be derived through simply coordinating nationally and aligning our collective efforts nationally.

R1.2: Evolve the ARCOS community

A process and resourcing to continue ARCOS as a national and international engagement mechanism is recommended. Community engagement, coordination and development are crucial in supporting Kubernetes - and containers more broadly - particularly at these early stages of uptake. Building cohesion amongst operators, developers and users will ultimately benefit all through the sharing of knowledge, practices and experience. The ARCOS community will also serve as the voice to guide priorities and the ongoing shaping of a national response to Kubernetes and containers.

R1.3: Promote best practices

Promoting better and best practices in the operation of Kubernetes, and the use of containers will strongly support all community participants in realising the benefits of these technologies. It is recommended that documentation, provenance, best practices, adoption guidelines, standard operating procedures and training resources appropriate for each of the layers in Figure 2 be promoted and/or developed in concert with ARCOS community priorities.

R1.4: Develop and share expertise

Developing and sharing deep expertise to the community is recommended. Such expertise should:

- Guide/help with the establishment of infrastructure and platforms from the onset
- Help navigate the intricacies and details of more advanced topics to enable the optimisation and improvement of deployments.

The expertise can be shared through various means, such as through guides, written documentation, providing hands on training, and designing and building resources, workshops, hackathons, and so on.

However, the development of identifiable and accessible individual(s) who are embedded within the ARCOS community that are known for that role is imperative.

5.2. Recommendation 2: Develop and promote standards

R2.1: Build and maintain Kubernetes design patterns

As there are various components in building a container orchestration platform, the creation of a set of national reference research infrastructure implementations (or design patterns) is recommended. These reference implementations should:

1. Align with a broad use case, community, or other research requirement (e.g. HPC)
2. Publish an integrated collection of vetted and tested examples for toolsets, networking configuration, dependencies logging and monitoring
3. Publish configuration examples for more advanced use of Kubernetes, including networking clusters and federations of clusters
4. Develop and publish a set of research ‘gold standard’ containers as methods to test the capabilities and compatibility of different Kubernetes deployments
5. Be updated periodically in response to community requirements and the broader international efforts, new tools and processes
6. Be developed in collaboration with the research infrastructure and research software engineering communities.

Related themes across operations, support, policies and security are also key to the realisation of more coherent and aligned national efforts.

5.3. Recommendation 3: Support the intersection of research software engineering and containers.

Research platforms built on Kubernetes will ultimately support research software, and it is recommended that a series of focussed activities be undertaken to address challenges that arise at the intersection of container use, software engineering, container development, and Kubernetes. Specifically:

R3.1: Integrate the RSE community into the ARCOS community efforts.

R3.2: Promote and provide support and training for the RSE community in:

a. The adoption of containers and the promotion of containers as a deployment/distribution methodology.
b. Practices and considerations specific to developing/porting software for use in container environments
c. Domain specific elements of container ecosystem, e.g. domain repositories and their approaches.
d. FAIR elements as related to containers, e.g. describing containers through metadata, associated documentation.
e. The management of containers through their lifecycle

R3.3: Design and implement a national research container registry and repository

R3.4: Design and implement other shared infrastructure in support of the above, which could include document registries, domain specific repositories, and so on.

Permeating these recommendations should be the FAIR principles as applied to containers.

5.4. Recommendation 4: Optimise the user experience

The experience of the end user and how they engage with a Kubernetes and/or containers based platform is key to the uptake and success of this technology. The spectrum of these experiences - from using a platform that is simply hosted on K8s, through to searching for and running a containerised software tool - needs to be articulated, defined and improved.

R4.1: It is recommend that further investigatory work be undertaken with the ARCOS community, *ARDC Platforms, NCRIS Platforms, facilities and other platforms, to articulate the various ways in which end-users (researchers) engage with containers and/or K8s, and understand their challenges and roadblocks.*

Naturally, this work should consider alignment with the FAIR principles - in that an understanding how researchers find, access, interoperate and re-use containers are all relevant issues. Activities such as the RDA FAIR 4 Research Software working group[7] should be consulted as part of this.

6. Study limitations statement

The limitations of these recommendations should be noted, in that the ARCOS community that was established predominantly consisted of operators and infrastructure developers. Although some engagement with developers and user communities occurred, it is recommended that further engagement be sought specifically with these groups.

[7] FAIR 4 Research Software (FAIR4RS) WG -
https://www.rd-alliance.org/groups/fair-4-research-software-fair4rs-wg

Appendix A - Australian use cases

AARNet	● Everything is containerised. ● S3 connectivity that we provide is built on K8s - wrote own Helm charts. ● Aiming to move Cloudstor to K8s. ● Swan is run on k8s at AARNET, every new Jupyter instance is a k8s pod. ● Swan is an owncloud service developed by CERN, but CERN doesn't use k8s for deployment. ● Align AARNet's cloud offering to be consistent so it can be used by many. ● Provide vanilla compute with K8s. ● Need QA processes for "shared" or created containers to ensure security
Auscope	● Interest in containers and K8s from the simulation side ● Australian research codes that are used globally - Dockerising these, used for teaching as well. ● Need to address issues around provenance of code, citations, etc - aiming to get academic credit. ● Zenodo registry for linking publications, data, and code/containers/workflows. ● Using binder and MyBinder.org as well. ● Have been using AWS.
Australian BioCommons *www.biocommons.org.au*	● The Australian Biocommons is actively engaged in projects which are exploring requirements, services and infrastructure for software and containers in life sciences. ● ELIXIR and the BioCommons have created and started a cooperative plan to exploit international synergies between the European and Australian research infrastructures, with software and container infrastructure included. ● A workshop held at Pawsey on 2020-03-11 discussed requirements and a planned approach, which is to follow the approach used by ELIXIR and Galaxy in using biotools, bioconda and biocontainers to package all software as containers. ● Pawsey, a partner of the BioCommons, is using Singularity containers to deploy software on cloud and HPC. ● An ambition of the BioCommons is to support deployment and use of platforms and tools across many different compute platforms nationally. ● They are working with different partners - NCI and Pawsey to begin with - followed by others, to deploy tools using containers, to provide a more consistent experience of software on different platforms.
Characterisation and Imaging	● Challenging to get software installed on HPC, but would like to make this faster, easier, portable and more repeatable. ● >100 Singularity containers used in production across research software and services.

	● Public container repository maintained for production and personal researcher machine use. ● Public container recipe repository with instructions for builds and best practices for deploying on interactive HPC. ● Multiple training events on container building run for researchers and admins at national sites and NCRIS capabilities. ● So far looked at containers but not orchestration for researcher software. ● Different people have different approaches to building containers, so there is a need for best practice. ● Many users created docker containers that are not portable.
CSIRO Mineral Resources	● Creating a platform to consolidate development. ● Researcher code is containerised and made available for execution via the platform. ● Aiming to be more portable. ● Using GitLab CI as a workflow engine, taking advantage of GitLab k8s deployments and auto-scaling runners. ● Key features around auto-scaling, using serverless workloads. ● using minio not specific buckets like GCP or s3 can connect to cloudstor - object storage universally used but different interfaces in different clouds so this affects portability.
Australia's Scalable Drones Cloud (ASDC)	● Building a cloud-native infrastructure for drone data, noting it will also need to consider workloads / container-orchestration onto the UAVs themselves (the edge). ● K8s as a core infrastructure to run scalable data processing applications, web-based analysis and visualisation applications. ● Coordinates & underpins the drone data processing of 3x NCRIS capabilities (APPF, Auscope & TERN), CSIRO and a university facility (Monash Drone Discovery Platform). ● Need to scale for compute intensivity, as well as to scale for societal interaction with drone data. ● Components of the ecosystem are co-developed with groups across the globe.
EcoCommons (and amalgamation of EcoCloud, BCCVL, CSDM, SilverEye)	● EcoCloud runs on openstack using k8s for deployment. ● Duplicated EcoCloud to Tinker (HASS DeVL). ● Had problems with limitations in the Research cloud. E.g. neutron networking not fully deployed, node instabilities, performance with storage backends, missing shared file systems. ● K8s deployment with adapted kubespray to cater for Research Cloud specifics. ● EcoCommons will use K8s for all platform services - while enabling workloads to be processed on external clusters/compute allocations.

	● Maybe ARDC can sort out k8s clusters at multiple cloud nodes/sites, 1 FTE for EcoCommons just sorting out deployment.
Monash eResearch Centre	● Making software available as docker containers, for the purposes of bin-packing more users and increasingly diverse users better onto the cloud. ● Need QA for the devops processes to create containers & their orchestrations - to assure the information governance of the user's data (e.g. so they don't get hacked or leak information). ● Best practice around containers is important - want to trust researchers. ● Different users have different approaches to K8s - e.g. Magnum, or not. ● Aim to run Jupyter notebooks in k8s clusters so can use smaller VMs - users may have their own entire Jupyter instance, or be sharing an instance, depending on their compute and information governance needs. ● Want to use containers and K8s for production digital facilities, like XNAT, Mytardis, CRAMS data-dashboard, cloud-dashboard, hpc-dashboard, ELK-stacks, and project web-presences (e.g. WordPress). ● Devops, cloud-native and open-source applied to health & medical sciences, e.g. CogStack (EHR -> NLP Engine) pipelines and FHIR on k8s. ● Run machine learning pipelines on k8s. ● Deploy/serve trained ML models. ● Run/provide DevSecOps services for teams e.g. Jenkins, authentication, monitoring.
NIF / MA	● NIF and MA manage containers for software tools and some for workflows. ● NIF and MA don't use K8s yet. Some of the applications for neurosciences use a Python library called Nipype that provides an interface for many tools to work together. ● Australian Imaging Service (AIS) ARDC Platforms project is aiming to use K8s for the multi-cloud deployment of multiple instances of XNAT and associated services.
TERN	● Run all applications and data processing pipelines as containers. ● Use K8s as a core platform to run client-facing applications and services containers. ● Applications interface with ARDC Research Cloud Object storage and Volume storage, CloudStor and RDS storage. ● Run K8s nodes in multiple ARDC Research Cloud regions (QRISCloud and TPAC). ● Use k8s to run automated Apache Airflow tasks for data onboarding to the TERN data infrastructure as and when data are available. ● Applications running on K8s interface with stand-alone applications running in their own clusters including Geoserver, Postgres, Apache Spark.

	● Trusted multi-tenant setup to allow various TERN groups to manage their applications.
University of Queensland / Swinburne Neuroimaging (Swinburne University of Technology)	● NeuroDesk - The first exemplar of a new cross-platform, out-of-the-box desktop environment for reproducible and scalable analysis of imaging data (https://github.com/NeuroDesk/vnm). Currently in production at UQ. ● NeuroDesk provides access to a library of containerised neuroimaging software. (https://github.com/NeuroDesk/caid/packages) ● NeuroDesk facilitates access to publicly available containerised pipelines, e.g. fMRIprep. ● AEDAPT is an ARDC platform call proposal that adapts NeuroDesk for electrophysiology data analysis. It proposes an ARCOS-led container registry as an ideal future solution for storing software containers.

Appendix B - Containers and Kubernetes usage survey

The following survey questions were entered into an online survey form and sent to the ARCOS Working Group in order to develop an understanding of their current uses and future aspirations with respect to Kubernetes and Container Orchestration. The responses were then followed up via virtual meeting to elicit more detail of the information provided in the survey.

1. **Services**
    1.1. What services do you provide to your users (related to containers)?
    1.2. How do users request these services?
    1.3. What is your current status with containers/container orchestration?
    1.4. What is the next goal for your organisation in the containers space?
2. **Staffing**
    2.1. How much staff time are you investing in migrating and maintaining container workloads and operating container infrastructure (# of FTE)?
    2.2. How would you rate the skill level of your staff with:
        2.2.1. Kubernetes app configuration/deployment
        2.2.2. Kubernetes operations
    2.3. What would you like to be doing, but are not able to do due to resource and/or skill constraints?
3. **Community**
    3.1. Who are the relevant upstream science/technology communities you're involved with for your use cases?
    3.2. Which other organisations are you partnering or sharing knowledge with to improve your container knowledge?
    3.3. Are there any specialised resources in your local community that you think would be useful to other organisations?
4. **Containerisation**
    4.1. How do you build your containers?
    4.2. Where are your container images stored?
    4.3. How do users find them? Do you run a local catalog?
    4.4. What have been the particular pain points during your container migration process?
    4.5. Where are your workloads currently running?
    4.6. Where do you plan to run workloads in the future?
    4.7. If using (or planning to use) public cloud, what are your primary motivations behind this?
5. **Container Orchestration**
    5.1. What are your motivations for using containers / Kubernetes over other solutions?
    5.2. If using Kubernetes, what is your current Kubernetes cluster deployment tool?
    5.3. Which particular features enabled by Kubernetes are useful for your workloads?
    5.4. Which related or alternative technologies are you using (or exploring) for creating/managing/deploying containers?

Appendix C - ARCOS Working Group

The following list includes the members of the ARCOS Working Group as of the 23 September 2020 who contributed to the Survey and this document.

Name	Organisation
Allan Williams	National Computational Infrastructure
Andrew Howard	National Computational Infrastructure
Andrew Perry	Monash University
Andrew Robinson	Australian National University
Andrew Treloar	Australian Research Data Commons
Andy White	Australian Research Data Commons
Arve Sollard	Ecocommons Australia
Audrey Stott	Pawsey Supercomputing Centre
Carina Kemp	AARNet
Cheryl Hung	Linux Foundation
Crystal Chua	AARNet
Carmel Walsh	Australian Research Data Commons
Chris Hines	Monash University
Craig Hamilton	Intersect Australia
Dalia Abraham	Australian Access Federation
Darshan Nagavara	Intersect Australia
Dean Taylor	University of Western Australia
Elisa Bayraktarov	Ecocommons Australia
Geli Kourakis	Tasmanian Partnership for Advanced Computing
Gerhard Weiss	Ecocommons Australia
Ian Thomas	RMIT University
Jake Yip	Australian Research Data Commons

Jay van Schyndel	Monash University
Jafaruddin Lie	Monash University
Jenni Harrison	Pawsey Supercomputing Centre
Jens Klump	CSIRO
Jerico Revote	Monash University
Jingbo Wang	National Computational Infrastructure
Jonathan Smillie	Australian Research Data Commons
Kieran Spear	Australian Research Data Commons
Kim Mcmahon	Linux Foundation
Lance Wilson	Monash University
Manish Kumar	Monash University
Marco De La Pierre	Pawsey Supercomputing Centre
Marco Fahmi	University of Queensland
Mark Gray	Pawsey Supercomputing Centre
Michael Mallon	Queensland Cyber Infrastructure Foundation
Nigel Williams	Tasmanian Partnership for Advanced Computing
Oren Civier	National Imaging Facility, Swinburne University of Technology
Owen Kaluza	Monash University
Paul Coddington	Australian Research Data Commons
P Holmes	
Paula Andrea Martinez	National Imaging Facility, The University of Queensland
Rhys Francis	Australian BioCommons
Rob Woolcock	CSIRO
Ryan Fraser	AARNet
Ryan Sullivan	Sydney University
Sam Bradley	CSIRO

Sam Morrison	Australian Research Data Commons
Sarah Beecroft	Pawsey Supercomputing Centre
Sean Smith	National Computational Infrastructure
Siddeswara Guru	Terrestrial Ecosystem Research Network
Steffen Bollmann	Centre for Advanced Imagery
Stephen Giugni	University of Melbourne
Steven McEachern	Australian National University
Steve Quenette	Monash University
Steven Manos	Australian BioCommons
Sunny Patel	Monash University
Swe Aung	Monash University
Toan Nguyen	Monash University
Tom Close	Monash University
Terry Smith	Australian Access Federation
Wei Fang	University of New South Wales
Wilfred Brimblecombe	Australian Research Data Commons
Wojtek Goscinski	Monash University