Institut für Parallele und Verteilte Systeme
Abteilung Anwendersoftware
Universität Stuttgart

# Master Thesis

## Orchestrating stateful database services in cloud environments using Kubernetes stateful services framework

***Bearbeiter:*** *Christian Müller*
***Prüfer:*** *Bernhard Mitschang*
***Betreuer:*** *Cataldo Mega*

**Background:** Data is a strategic asset to every enterprise and is therefore subject to data governance. The aim for data governance is to strategically managed business relevant data through its entire lifecycle from creation to disposition. The technical aspect of data governance are realized as set of policies and practices implemented to support business processes, corporate policies and regulatory compliance. Once captured all data of all types must be classified, categorized and securely stored such to meet high legal and regulatory requirements. These requirements imply the existence of an enterprise wide IT infrastructure of secure data repositories that facilitates data sharing between different user, groups or applications. Sharing data is always according to their sensitivity, relevance and jurisdictional restrictions. Given the huge amount of data and the need to interconnect and share it with potentially any other company on the globe, the thought is to align current IT with today's cloud technology to scale on demand and cut cost down. The vision is a sort of a hybrid multi-cloud approach as an alternative to a closed legacy infrastructure platform. In this context we call unstructured data = 'content' and unstructured data stores are 'content repositories' in order to distinguish it from metadata which is typically stored in relational database systems.

In the past companies have managed their 'content' using home grown monolithic data governance application stacks on top of distributed content repositories, typically operated on bare metal or at most in a virtualized IT infrastructure. And though they still offer reliable and performant content services they do lack the required flexibility of an automated continuous integration / continuous deployment (CI/CD) in a fast, efficient and cost effective way. Based on our analysis we think the major problem derives from the fact that the existing ECM systems are still using an outdated client server architecture' model and a legacy IT infrastructure and as such they are not able to exploit new cloud technologies nor be able to support CI/CD models. As a consequence, companies must entertain the effort for a 'cloud transformation' in order to benefit from the 'Cloud'.

Aside from the organizational and cultural changes required by a cloud transformation process, the technical challenge is to refactor and enhance the application design and to develop a cloud deployment model before rolling out a cloud specific implementation of the solution. In essence what this means, is to enforce a paradigm shift from 'client server' to a 'cloud computing' architecture model. Our expectation is to benefit from the exploitation of the economies-of-scale, the advantage of the inherent multi-tenancy capability and the ubiquity of the service offering in an increasingly secure and trusted cloud environment.

**Key words:** Kubernetes, stateful services, cloud native databases, elastic topologies, scale-up, scale down.

## Master Thesis Content

### Goals

This master thesis will focus on the handling of stateful database services in Kubernetes. More specifically, developing the Kubernetes (K8s) deployment models of the ECM catalog database, the content repository and the persistent storage required by ECM services. And we want all of the above orchestrated as stateful services by the K8s) runtime system. The aim is to design and deploy a set of stateful DB2 database services integrated in an ECM application stack. Put it in simple terms, we want to build stateful service wrappers around an IBM DB2 database system using the Kubernetes stateful services SDK with the goal for the database be able to at least fail-over and fail-back horizontally to kubernetes nodes in an elastic fashion and provide stateful database services be available everywhere in the configured Kubernetes cluster.

In previous master thesis, see [1] and [2], a first refactoring step was taken where the content repository application stack was decomposed into smaller self-contained components and encapsulated into Docker container images. Now we want to move the Docker database containers as stateful services in to a Kubernetes cluster making sure data consistency, high availability (HA) and disaster recovery (DR) can be achieved.

### Approach:

### Part I: Problem analysis and concept design relative to stateful services deployed and managed in clouds (2m)

To start with:

- Start with an architectural analysis by comparing traditional databases vs cloud native data bases. Look at IBM DB2 and PostgreSQL as legacy RDBMS systems and compare them against CockroachDB and Google F1/Spanner, so called cloud native databases. Explain the key differences and document the pros and cons. Then consolidate the documentation and present your intermediate results.
- Next familiarize yourself with the basic architecture design of the content repository application stack used in the previous work. Focus on the backend and related stateful services.
  Read and understand MA [1],[2], and [3] listed in the reference section, but dep dive only MA [1] and [2] as a starting point.
- Also learn about the basic architecture design of Kubernetes/Docker. Understand what virtualized and containerized cloud environments are and the aspects of cloud orchestration technology, the dynamic deployment capabilities as well the service life cycle management that the Kubernetes eco-system offers. I.e. **operators, control loop, custom resource definition (CRD)**
- Deep dive into the Kubernetes stateful services architecture design. Learn about **stateful sets, operator pattern, persistent volumes** and how to use YAML to declaratively define services templates used for an automated deployment on to a Kubernetes cluster. At this point look at the enterprise content repository components: like the catalog database, the content repositories and the persistent store and sketch out a Kubernetes specific architecture design that use the Kubernetes stateful services SDK to design required custom operators and custom resource definitions for the above mentioned components. Then develop the deployment scripts required to setup and rollout the set of stateful content repository services.

***Note: Use the self-study tutorial videos listed below.***

### Part II: Prototype implementation of the DB2 stateful database services orchestrated and managed by Kubernetes (2m)

Use the current prototype system with its basic content management application stack, its catalog database and content repository (see [1], and [2]). As shown in fig.2 in this context, IBM DB2 is used as the external database system in 2 different containers. Use DB2 and the above mentioned expert knowledge source to develop a custom DB2 operator. In addition choose an available storage system to persist database data and content like any cluster file system, NFS, Ceph or equivalent others. Then,

- Use the ECM prototype shown in figure 2 above and enhance and extend its deployment model as designed in your work of part I.
- Define a Kubernetes service consisting of multiple pods running multiple containers each of which running an ECM application component. Something similar to what is shown in figure2.
- Perform a number of tests that shows your approach is working as expected.
  - Your data base service scales horizontally over a number of nodes in an elastic fashion.
  - Your service survives a node failure
  - Your services is available from everywhere in the cluster.
- Consolidate the documentation so far and present your intermediate results

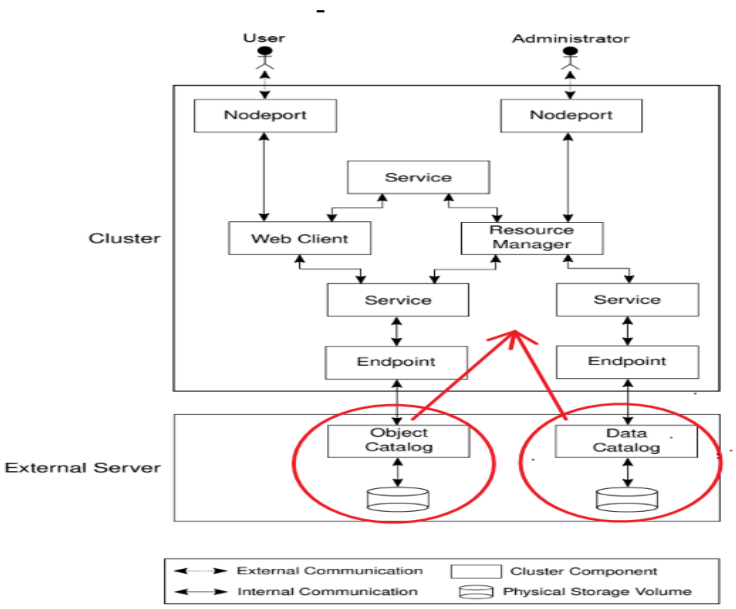


Figure 1: Phase 1 content repository prototype using external services.

**Part III: (2m)**

- Evaluate the results and consolidate documentation you have developed so far
- Explain the new cloud' delivery and the enhanced service offering model.
  Show the benefits of a possible successful content repository cloud transformation strategy.
- Final presentation and explanation if the final results

**Outcome and expected results:**

The expected results for this master thesis are:

1. A comparison between traditional databases and cloud native databases should be performed. Document their key characteristics and the differences with respect to functional and non-functional business requirements similar to the work done in this paper [5].

2. Understand and document the cloud mechanism that Kubernetes offers to achieve elasticity, resiliency scale by shrinking or expanding the overall deployment topology based on system workload load and resource component state.

3. Using the previous MA work results refactor and enhance the available Kubernetes deployment model complementing it with the set of stateful database services and put them under control of the Kubernetes orchestrator in a proof of concept and document your results. Show how your K8s enhanced deployment model can be used to roll-out in an automated fashion a set of stateful content management services in a cloud environment based on Kubernetes/Docker.

For this thesis you can use the IPVS-AS Lab on a pre-provisioned OpenStack cluster of CentOS VMs and an ECM prototype system developed using Kubernetes / Docker cloud technologies running on a Kubernetes cluster. As an alternative you can use a IBM / RedHat sponsored cloud sandbox developing the prototype on a RedHat Openshift / Kubernetes environment.

**References and useful links:**
Familiarize yourself with the current ECM system architecture design and its context.


1. C. Trybek. “Investigating the Orchestration of Containerized Enterprise Content Management Workloads in Cloud Environments Using Open Source Cloud Technology Based on Kubernetes and Docker”. Master Thesis. Germany: University of Stuttgart, September 2021.
2. P. Hagemann. “Evaluating dynamic load balancing of ECM workload pattern employed in cloud environments managed by a Kubernetes/Docker eco-system”. Master Thesis. Germany: University of Stuttgart, September 2021,
3. G. Shao. “About the Design Changes Required for Enabling ECM Systems to Exploit Cloud Technology”. Master Thesis. Germany: University of Stuttgart, December 2020.

4. Expert knowledge: Content Manager Backup/Recovery and High Availability: Strategies, Options, and Procedures Link->
5. Cloud native alternatives: What's Really New with NewSQL? Link


**Tutorial Videos:**
Please perform a self-study on Docker, Kubernetes and the Kubernetes stateful services framework using the following tutorial videos:

- Docker tutorial: https://www.youtube.com/watch?v=3c-iBn73dDE
- Kubernetes Intro: - https://www.youtube.com/watch?v=s_o8dwzRlu4
- Kubernetes architecture: https://www.youtube.com/watch?v=aSrqRSk43lY
- Kubernetes StatefulSets: https://www.youtube.com/watch?v=pPQKAR1pA9U
- Kubernetes persistent volumes: https://www.youtube.com/watch?v=0swOh5C3OVM
- Kubernetes operators - https://www.youtube.com/watch?v=i9V4oCa5f9I

- How to run HA IBM DB2 on Kubernetes:

**Related work:**


- Mega, C; Waizenegger, T. ; Lebutsch, D. ; Schleipen, S. ; Barney, J.M. Dynamic cloud service topology adaption for minimizing resources while meeting performance goals, In IBM Journal of Research and Development Issue 2 • Date March-April 2014
- Ritter T., Mitschang B., Mega C. (2012) Dynamic Provisioning of System Topologies in the Cloud. In: Poler R., Doumeingts G., Katzy B., Chalmeta R. (eds) Enterprise Interoperability V. Proceedings of the I-ESA Conferences, vol 5. Springer, London. https://doi.org/10.1007/978-1-4471-2819-9_34
- Fritz, Florian: Maximization of resource utilization through dynamic provisioning and de-provisioning in the cloud, Diplomarbeit Nr. 3078, 2011.
- Börner, Andreas: Orchestration and Provisioning of Dynamic System Topologies, Diplomarbeit Nr. 41, 2011.