RESEARCH PROPOSAL

SUBMITTED TO THE

UNIVERSITY OF READING

Department of Computer Science

PHD Thesis

Author:
Kyle SPINDLER

Supervisor:[1]
Dr. Julian KUNKEL

October 26, 2018

Keywords: Microservice, Serverless, HPC, Storage

High-Performance, Storage Software Architectures

[1]This thesis has been discussed with the listed supervisor.

1 Motivation

Software Architecture invovles considering multiple characteristics such as separation of concerns, quality attributes (maintainability, scalability, loose coupling, high cohesion etc...) and architectural styles. Some architecture styles are more suited for performance while others are better at maintainability and loose coupling like microservices. Microservices is a very popular architecture that is used in many domains because of the benefits it offers.

Scientic codes suffer from good software engineering practices. HPC and store applications are typically tightly coupled to utilize the available resources efficiently. While it is claimed that this provides the best performance, the benefit and drawbacks of alternative software architectures for HPC software is not thoroughly investigated. Microservices, for example, provide a scalable architecture and ease the software development process by providing separation of concerns by applying techniques from Domain Driven Design. When deciding a software architecture not only performance and scalability matters, but also flexibility and maintainability of the software.

In this regard, the HPC community struggles to recruit sufficient developesr to keep up with the development of software which can often be seen in important utility tools. For example, existing tools for pre/post-processing of HPC workflows and the analysis of HPC data are typically not the main focus of scientists and developers; hence, they are implemented in a way that shows limited scalability, i.e. are executed sequentially in bash scripts.

2 Research question

Understand the impact of modern day software architectures (microservices, event driven) has on HPC and particularly the climate/weather domain
The goal of this thesis is to see if HPC applications and storage systems can be redeveloped using modern day software architecture such as microservices with minimal or no overhead while gaining the benefits from the loosely coupled architecture.

1. What parts of the HPC and Storage Solution could benefit from microservices or other software architectures?
2. How to make HPC and Storage Solution more maintainable, scaleable, loosely couple, more cohesion and more independent?
3. What areas within the HPC and Storage Solution that could improve it's efficiency through the use of Software Architecture?

3 Related work

Microservices are becoming very popular in today's world due to the main benefits and problems it solves. The application of microservices are used in many domains where maintainability, scalability and resilience is very important as opose to scientific applications that requires performance is their primary attribute. Although microservices are found in HPC applications the majority of research found has applied microservices in storage, pre/post processing, middleware, scheduling, workflow and caching services that wrap around the main HPC processing.Also storage services like iRODS harness microservices, however, these systems are not as performance-criticial as a typical HPC application which might be one of the reasons why iRODS is not used in HPC environments.

Relevant work can be classified into: a) LaTeX studies, b) performance analysis in HPC, ...

Caching Microservices. Microservices were used as a distributed interpolation-based memoization cache. (Jenkins et al., 2017).

Containers for High Performance Computing. Thoughts on how containers may or may not be used in HPC. Containers are commonly used in microservice architectures. (Jackson, 2018).

Distributed Virtual Machine Cloud Microservice for HPC:SPMD Applications How Virtual Machines were used in a cloud based microservice architecture in an HPC environment:SPMD. (Fatéma et al., 2017).

Scheduling Scientific Workflows in HPC How to dynamically approach to scheduling reconfigurable scientific workflows in heterogeneous HPC environments. (Cheptsov, 2016).

Middleware cloud based microservice in HPC Shows how middleware has made use of microservices within a HPC environment. (Benchara et al., 2016).

iRODS integrated Microservice Rulebook Shows how iRODS uses Microservices for storage. (Rajasekar et al., 2015).

Microservices in Ocean Climate Data Shows how Ocean-Derived Climate Data uses Microservices. (Johanson et al., 2016)

Software Engineering in Computational Science How Software Engineering practices can be used in Computational Science environments. (Johanson and Hasselbring, 2018)

Workflow-Oriented Cyberinfrastructure for Sensor Data Analytics

http://eprints.uni-kiel.de/42726/1/2018-04-19GeomarDataScience.pdf

4 Research methodology

Firstly, I must understand the limitations and performance characteristics of alternatives software architectures. This is performed in two steps:

1. Benchmarking of microservice communication protocols in contracts to HPC communication path.
2. Modelling of systems with different hardware / software architectures

Next, from HPC use cases and scenarios must be derived and characterised with their pros/cons and performance chatacterisitcs.
To prove the expectations, a prototype of selected applications must be made and compared to native applications. This may involve to adjust an existing package or create a new application that behaves similar but has limited feature set (so called mini-app).

An orthogonal aspect is to conduct surveys with scientists / scientific developers to understand the reason for the architectural choices made and identify strategies to adjust the existing practice.

5 Required infrastructure

Access to HPC systems to benchmark existing software and the developed prototypes. Access to the scientific network to foster discussions and explore with scientists in co-development the benefits of the prototypes.

6 Workplan

The following sketches the workplan for the different years of the PhD.

First year: Setup of work environment, researching related work, Start Investigating system under study, send out surveys, analyise system under study, writing the chapters introduction and related work of the thesis.

Second year: Continue researching related work, Add more details in thesis based on previous findings, Start Implementing Prototype.

Third year: Continue Implementing Prototype, Add Quality Assurance and testing to prototype, continue researching, add implementation details to thesis

Fourth year: Run Benchmark Tests, Adjust Prototype, add benchmark findings into thesis, provide critical thinking.

Fifth year: Finalise thesis, Prepare thesis for final stages and delivery

A Appendix

Add here any appendix, if needed

References

Benchara, F. Z. et al. (2016). "A new efficient distributed computing middleware based on cloud micro-services for HPC". In: *2016 5th International Conference on Multimedia Computing and Systems (ICMCS)*, pp. 354–359. DOI: 10.1109/ICMCS.2016.7905644.

Cheptsov, A. (2016). "Dynamic Approach to Scheduling Reconfigurable Scientific Workflows in Heterogeneous HPC Environments". In: *2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS)*, pp. 7–14. DOI: 10.1109/CISIS.2016.146.

Fatéma, Zahra et al. (Aug. 2017). "Toward a New Massively Distributed Virtual Machine based Cloud Micro-Services Team Model for HPC: SPMD Applications". In: *International Journal of Advanced Computer Science and Applications* 8, pp. 238–249. DOI: 10.14569/IJACSA.2017.080831.

Jackson, Joab (2018). "Containers for High Performance Computing". In: URL: https://thenewstack.io/roadmap-containers-for-high-performance-computing/.

Jenkins, John et al. (2017). "A Case Study in Computational Caching Microservices for HPC". In: DOI: 10.1109/IPDPSW.2017.40.

Johanson, A. and W. Hasselbring (2018). "Software Engineering for Computational Science: Past, Present, Future". In: *Computing in Science Engineering*, pp. 1–1. ISSN: 1521-9615. DOI: 10.1109/MCSE.2018.108162940.

Johanson, Arne et al. (2016). "OceanTEA: Exploring Ocean-Derived Climate Data Using Microservices". In: *Proceedings of the Sixth International Workshop on Climate Informatics (CI 2016)*. NCAR Technical Note NCAR/TN, pp. 25–28. URL: http://eprints.uni-kiel.de/34758/.

Rajasekar, Arcot et al. (2015). *The Integrated Rule-Oriented Data System (iRODS 4.0) Microservice Workbook*. 1st. USA: CreateSpace Independent Publishing Platform. ISBN: 1511732776, 9781511732772.