Integration of cloud computing paradigms for performant analysis of simulated process engineering applications

A R&D report (Dissertation) submitted in partial fulfilment

of the requirements for the degree

of

Bachelor of Engineering (Hons)

at

The University of Waikato

by

Caleb Archer

Supervised by:

Tim Walmsley, Mark Apperley

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

2024

Abstract

Acknowledgements

Declaration of Authorship

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The process systems engineering (PSE) field is critical to enabling detailed analysis and optimisation of industrial chemical processes. Despite this, modernisation of accessible optimisation software tools and techniques has slowed, with most software-based analysis performed via desktop applications constrained by the resource limits of local systems. More efficient optimisation techniques are available, but require software programming proficiency not held by the common process engineer. The cloud computing sector has significantly advanced the ability to run and deliver software in a distributed fashion, without requiring clients to procure and maintain physical hardware. Cloud-native software systems are better positioned to respond to dynamic usage and allocate computational resources efficiently based on need. There has been minimal convergence of the PSE and cloud computing fields.

This work details and evaluates the development of a cloud-native PSE software platform (the Ahuora Digital Platform) on a physical Kubernetes cluster, comparing the performance of the different software components running on the cluster versus a local deployment of said components.

1.1 Background

1.2 Summary of Literature Review

Chapter 2

Research and Development Approach

2.1 Kubernetes cluster architecture

2.2 Performance analysis experiment structure

2.2.1 Controlling variables

2.2.2 Preliminary performance modelling

2.2.3 Empirical comparison

2.2.3.1 Grafana k6 load-testing

2.2.3.2 Grafana system monitoring

2.2.4 Key metrics

2.2.4.1 Response time

2.2.4.2 Throughput

2.2.4.3 Quantile comparison

2.3 Performance analysis experiments

2.3.1 Benchmark

2.3.2 Test scenarios

2.3.2.1 Average-load testing

2.3.2.2 Stress testing

2.3.2.3 Soak testing

2.3.2.4 Spike testing

2.3.2.5 Breakpoint testing

2.3.3 Resource allocation

2.3.4 Horizontal scaling policies

2.3.4.1 Minimum replicas

Chapter 3

Implementation

3.1 Raspberry Pi hardware and network provisioning

For ease of access and cost minimisation purposes, a set of eight Raspberry Pi 5 computers was obtained to run the Kubernetes cluster on. Each device has an active cooler component installed to effectively cool the CPU (Central Processing Unit) and prevent system throttling. The usage of these devices allows for the construction of a physically compact computing cluster at low cost.

A headless (sans desktop interface) version of Raspberry Pi OS (operating system) was loaded onto eight corresponding SD cards, which each device uses as primary storage. The headless version of the OS strips the resource consumption of the desktop user interface, which is not required, as most interaction with each device will be automated over the network, requiring no more complication than a remote CLI (Command-Line Interface) provides.

3.2 Ansible playbook automation

3.3 Isolated cluster access

3.3.1 Network airgapping

3.3.2 ZeroTier Virtual Private Network (VPN) usage

3.3.3 Private container image registry mirror

3.3.4 Cloudflare Tunnel ingress point

3.4 Kubernetes manifest configuration

3.4.1 Deployments

3.4.2 Services

3.4.3 Ingresses

3.4.4 Authentication

3.5 Kubernetes deployment automation

3.5.1 ArgoCD manifest synchronisation

3.5.2 GitHub Actions platform release pipeline

Chapter 4

Results and Discussion

4.1 Outcomes

4.1.1 Resource allocation

4.1.2 Horizontal scaling policies

4.1.2.1 Minimum replicas

4.1.2.2 Target resource utilisation

4.1.2.3 Variable load conditions (stabilisation window)

4.1.3 Clustered vs. local PSE optimisation performance

4.2 Impacts

Efficiency, accuracy, feasibility (examples of demonstrable impacts).

4.3 Threats to validity

Chapter 5

Conclusion and Future Work

References