Themenanmeldung

Projekt titel

Development and demonstration of a proof-of-concept for the integration of programming frameworks for high performance computing into a container-based workflow orchestrator.

Problem Definition

Im Bereich der Hochleistungsrechnung besteht aktuell ein Bedarf für robuste Workflow-Management Werkzeuge. Dieser Bedarf speist sich aus der zunehmenden Komplexität der Daten und der daraus resultierenden Prozesse, welche diese Daten verarbeiten. Um diese Komplexität zu bewältigen bedarf es zum einen einer Automatisierung und Abstraktion der alltäglichen Aufgaben welche für die Instandhaltung dieser Datensätze und deren Endprodukte nötig sind. Prozesse, welche in der Softwareentwickelung bereits seit Jahren etabliert sind, wie z.B. Versionierung, automatisierte Tests und die darauf folgende Integration in die Produktivumgebung, werden nun auch für die Verwaltung von Daten immer wichtiger. Deshalb entstehen nun auf dem Markt einige Produkte, welche versuchen sich in den Bereichen MlOps und DataOps zu etablieren.

Viele dieser Werkszeuge spezialisieren sich jedoch auf die Anforderungen der Machine Learning Community, welche nur ein Teilbereich der Hochleistungsrechnung ist. Diese Orchestratoren sind meistens auf die Verarbeitung von Daten innerhalb von Containern ausgelegt, diese die Verwaltung von Abhängigkeiten und die Isolation von Prozessen erleichtern. Um diese Werkzeuge auch für andere Bereiche der Forschung und Industrie attraktiv zu machen, bedarf es der Integration von klassischen Hochleistungsrechen-Frameworks. Diese fuer Hochleistungsrechner entwickelten Frameworks sind jedoch historisch bedingt nicht auf Container sondern ausgelegt, sonder arbeiten mit bare-metal Job-Schedulern wie Slurm. Eine Integration dieser Frameworks, wie beispielsweise Chapel oder Julia, in ein containerbasiertes Workflow Management Tool könnte helfen diese Lücke zwischen den beiden Welten zu schließen und die Vorteile beider Ansätze zu vereinen.

Um dieses Problem zu adressieren, soll nun im Rahmen der Projektarbeit ein Proof of Concept entwickelt werden, welcher anhand einer beispielhaften Workflow-Pipeline die Integration eines Hochleistungsrechnungsframeworks in ein Workflow Management Tool demonstriert.

Dabei gilt es besonders den daraus resultierend Overhead zu betrachten, welcher durch die verschiedenen Abstraktionslagen wie Containerisierung und die Differenzierung der einzelnen Komponenten in Microservices entsteht uns dieses mit den Vorteilen durch die Skalierbarkeit und die automatisierte Verwaltung der einzelnen Komponenten abzuwägen.

Ziele der Arbeit

Das Ziel der Arbeit ist es, die Herausforderungen und Anforderungen zu analysieren, die bei der Integration eines Hochleistungsrechnungsframeworks in ein containerbasiertes Workflow-Management-Tool auftreten. Diese Analyse wird der erste Schritt sein, um ein tieferes Verständnis der bestehenden Probleme und Bedürfnisse zu gewinnen. Auf der Grundlage dieser Analyse wird ein Proof of Concept (PoC) entwickelt und implementiert, der die Machbarkeit und Effektivität einer solchen Integration zeigt. Dieser PoC wird auf seine Leistung, den Overhead und seine praktische Nutzbarkeit bewertet, um die Vorteile und Nachteile eines solchen Ansatzes zu ermitteln.

Ein weiteres Ziel der Arbeit ist es, potenzielle Verbesserungen und zukünftige Forschungsrichtungen zu identifizieren.

Die Forschungsfrage, die diese Arbeit beantworten soll, ist dreifach:

1. Wie kann ein Hochleistungsrechnungsframework effektiv in ein containerbasiertes Workflow-Management-Tool integriert werden?
2. Inwiefern beeinflusst die Nutzung eines containerbasierten Workflow-Management-Tools die Leistung des Hochleistungsrechners?

3. Welche Möglichkeiten zur Verbesserung der Integration von Hochleistungsrechenframeworks in containerbasierte Workflow-Management-Tools gibt es?

Die Beantwortung dieser Fragen wird dazu beitragen, die bestehende Lücke zwischen der Hochleistungsrechen- und der containerbasierten Welt zu überbrücken und so die Vorteile beider Ansätze zu nutzen.

Methodik

Um die oben genannten Fragen zu beantworten und ein Artefakt zu kreieren wird ein iterativer Prozess verfolgt, welcher sich an den Prinzipien des Design Thinking orientiert. Dabei werden 5 Phasen durchlaufen, welche eine kontinuierliche Verbesserung des Artefakts ermöglichen.

Nutzerbedürfnisse verstehen

Zuerst wird sich mit den zukünftigen Nutzern des Artefakts auseinandergesetzt. Dabei handelt es sich um die Forscher und Entwickler, des Hewlett Packard Enterprise (HPE) Labors in Milpitas, Kalifornien. Diese werden in einem Interview befragt, um die Anforderungen an das Artefakt zu ermitteln. In zukünftigen Iterationen wird das Artefakt dann mit den Nutzern getestet, um anhand des Feedbacks die Anforderungen zu verfeinern.

Erstellen eines Szenarios

Basierend auf den Anforderungen wird ein Szenario erstellt, welches den Entwickler in das Mindset des Nutzers versetzt. Dieses Szenario wird dann als Grundlage für die Entwicklung des Artefakts verwendet und dient gleichzeitig als eine demonstrierbare Anwendung. In weiteren Iterationen kann dieses Szenario entweder erweitert oder durch ein neues ersetzt werden, welches einen anderen Anwendungsfall abdeckt um das Feature-set nuetzlich zu erweitern.

Design des Artefakts

Nun kann basierend auf dem Szenario die Architektur des Artefakts entworfen werden. In dieser Phase treffen die Anforderungen des Nutzers auf die technischen Möglichkeiten und Einschränkungen. Hier werden die Entscheidungen darueber getroffen, welche technologischen Komponenten verwendet werden und wie diese miteinander interagieren sollen. Bei weiteren Iterationen wird dieses Design dann verfeinert und an die neuen Anforderungen angepasst.

Implementierung des Artefakts

In dieser Phase wird das Artefakt moeglichst nah zu dem Design aus der vorherigen Phase implementiert. Während dieser Integration wird sich zeigen an welchen Stellen neue Anforderungen entstehen und welche Anforderungen oder Designentscheidungen nicht wie erwartet funktionieren. Diese Erkenntnisse werden dabei in die naechste Iteration mitgenommen und dort beruecksichtigt.

Wichtig ist bei dieser phase auch die Dokumentation der einzelnen Schritte, um die Entscheidungen nachvollziehbar zu machen und die Ergebnisse reproduzierbar zu gestalten. Diese Nachvollziehbarkeit ist nicht nur fuer Nutzer und Dritte wichtig, sondern auch fuer die eigene Arbeit da diese die Grundlage fuer die weiteren Iterationen bildet.

Evaluation des Artefakts

Nun kann das Resultat anhand der gewonnenen Erkenntnisse evaluiert werden. Dabei wird nicht nur das Artefakt als solches betrachtet, sonder auch die zugrunde liegenden Annahmen welche wir bei der Erstellung des Szenarios und der Entwickelung des Designs getroffen haben. In diesem Schritt findet auch die Vorbereitung fuer die naechste Iteration statt, in der die gewonnenen Erkenntnisse in die Verbesserung des Artefakts mit einfliessen.

Dabei werden neue Aspekte und Anforderungen identifiziert, welche in den naechsten Iterationen beruecksichtigt werden muessen.

Gliederung

1. Introduction
    - Problem Statement
    - Aim and Objectives
    - Research Questions
2. Methodology
    - Understanding User Needs
    - Scenario Creation
    - Artifact Design
    - Artifact Implementation
    - Artifact Evaluation
3. Related Work and Background
    - Workflow Management Tools
    - High Performance Computing Frameworks
    - Containerization
    - Microservices
4. Creation of the Artifact
    - User Needs
    - Design
    - Implementation
    - Evaluation
5. Conclusion
    - Summary
    - Future Work
6. References
7. Appendix

Zeitplan

Woche 1-2: Einführung

```
Problemstellung definieren
Ziele und Aufgaben festlegen
Forschungsfragen aufstellen
```

Woche 3-4: Methodik

```
Nutzerbedürfnisse verstehen
Szenario erstellen
```

Woche 5-7: Hintergrundarbeit und verwandte Arbeiten

```
Workflow-Management-Tools
Hochleistungsrechen-Frameworks
Containerisierung & Mikroservices
```

Woche 8-9: Entwicklung des Artefakts - Benutzerbedürfnisse

```
Durchführung von Interviews mit Benutzern
Identifizierung von Benutzeranforderungen
Synthese und Analyse der gesammelten Daten
```

Woche 10-11: Entwicklung des Artefakts - Design

Auswahl der zu verwendenden Technologien
Kozeptionierung des Systems und seiner Komponenten
Planung der Interaktion zwischen den Komponenten

Woche 12-15: Entwicklung des Artefakts - Implementierung

Codierung und Entwicklung des Systems
Behebung von Problemen, die während der Implementierungsphase auftreten

Woche 16-17: Entwicklung des Artefakts - Bewertung

Durchführung von Benutzertests
Sammlung von Feedback
Durchführung von Leistungstests

Woche 18-19: Schlussfolgerung

Zusammenfassung der Ergebnisse
Diskussion der Auswirkungen der Forschung
Erörterung möglicher Verbesserungen und zukünftiger Forschungsrichtungen
Erstellung des Abschlussberichts und der Präsentation

Woche 20: Pufferzeit

Zeit für unerwartete Verzögerungen oder zusätzliche Arbeit