Designing an Autonomous, Distributed Microservice System for Emergency Services in Disconnected Environments

Author

Van Quoc Tuan Nguyen
School of Computer Science
University of Adelaide
vanquoctuan.nguyen@student.adelaide.edu.au

Supervisors

Sree Ram Boyapati
School of Computer Science
University of Adelaide
sreeram.boyapati@adelaide.edu.au

Zihan Zawad
Swordfish Computing
zihan.zawad@swordfish.com.au

Abstract *– Emergency services rely heavily on software systems and applications to effectively respond to major events such as floods or bushfires. However, these software systems are often supported by services that may not function well when disconnected from the internet. This research aims to design an autonomous, distributed microservice system that can keep emergency services running as best as possible even when the network connection is poor and computing resources are limited. To achieve this, a containerised microservice test environment will be implemented to test the orchestration system. Experiments will be designed and implemented to evaluate the system's performance in disconnected environments. Additionally, new distributed orchestration algorithms will be designed and tested to improve the system's resilience in adverse conditions. This project will address challenges in microservices, containerisation, autonomous orchestration, and self-adaptive systems. The outcome of this research will be a more robust and reliable system that can support emergency services in disconnected environments, improving their effectiveness in responding to major events.*

Keywords – microservices, containerisation, autonomous orchestration, self-adaptive systems, emergency services, disconnected environments.

I. INTRODUCTION AND MOTIVATION

Motivation

The importance of reliable communication and information sharing in emergency response operations has been well-documented in the literature. Studies have shown that effective communication and coordination among emergency responders are critical for successful operations (He et al., 2019; Legg et al., 2016) **[1][2]**. However, poor network connectivity can pose a significant challenge to communication and information sharing in disaster zones. This can result in delays in critical information being transmitted between responders and headquarters, potentially leading to serious consequences such as loss of life and property damage. Therefore, the development of an autonomous, distributed microservice system that can operate in low-connectivity environments can be a game-changer for emergency response operations. This system can improve the reliability and efficiency of communication and information sharing among responders and headquarters, even in areas with limited computing resources.

Overall, this research is motivated by the need to address the challenges posed by poor network connectivity and limited computing resources in emergency response operations. By developing a system that can operate in these environments, it may be possible to improve the effectiveness of emergency response operations and ultimately save lives.

Introduction

Emergency services, such as firefighters and paramedics, are often called upon to respond to major events such as floods, bushfires, and earthquakes. In order to effectively manage these events, emergency services rely on software systems and apps to help them do their job more efficiently. These software systems are supported by various services, but may not function properly when there is poor network connectivity or limited computing resources available. For instance, if a fire truck transmits its position and video to the fire headquarters (HQ) and receives map data in return, how would the system work if the truck enters an area with no radio connectivity to the HQ? To address these issues, there is a need to design an autonomous, distributed microservice system that can keep services running even when the network connection is poor or there are limited computing resources. This involves solving

various problems related to microservices, containerisation, autonomous orchestration, and self-adaptive systems. Several research studies have highlighted the importance of developing such systems. For example, in a study by Y. Zhang et al. (2020) **[3]**, the authors proposed a self-healing microservice architecture for cloud-based systems that can detect and recover from failures in a self-managed manner. Similarly, R. Ranjan et al. (2016) **[4]** developed an adaptive and autonomic framework for containerised microservices to provide fault tolerance, scalability, and self-healing capabilities. These studies demonstrate the importance of developing autonomous, distributed microservice systems that can adapt to changing conditions and ensure continuous service availability.

Research question: How can an autonomous, distributed microservice system be designed to keep emergency service software systems running as best as possible even when the network connection is poor and there are limited computing resources available ?

The proposed research aims to design and implement such a system. The project will involve developing a containerized microservice test environment, designing and implementing experiments to test the orchestration system, and designing and testing new distributed orchestration algorithms. The project will focus on several key areas, including microservices, containerization, autonomous orchestration, and self-adaptive systems. The importance of this research is underscored by the fact that emergency services software systems are critical for saving lives and property during emergencies. By developing an autonomous, distributed microservice system that can operate effectively in low-connectivity environments, emergency responders will be better equipped to handle emergencies in areas with limited network coverage. This research will also contribute to the broader field of microservices and autonomous systems, which has implications for a wide range of industries and applications.

II. LITERATURE REVIEW

Emergency services are essential for saving lives and protecting property during natural disasters such as floods or bushfires. To enhance the effectiveness of emergency services, software systems and apps have been developed to help emergency responders do their job better. However, these software systems are heavily reliant on internet connectivity, and when the network connection is poor, they may not work as expected, hindering the emergency response efforts. One solution to address this issue is the development of an autonomous, distributed micro service system that can keep services running as best as possible even when there are limited computing resources available and the network connection is poor. Microservices are small, independent software components that can be deployed separately, making them ideal for distributed systems. Additionally, containerization technology can be used to package the microservices in lightweight, portable containers that can be easily deployed and managed. Several research studies have explored the use of microservices in emergency response systems. For instance, in the paper, "Towards Microservice-based Emergency Response Systems," Hu et al. (2019) **[5]** proposed a microservice-based architecture for emergency response systems. The authors discussed the advantages of using microservices, such as scalability and flexibility, and demonstrated how the architecture can be implemented in the context of an emergency response system. Another study by Zhang et al. (2019), "Microservice-based Internet of Things architecture for disaster rescue," **[6]** proposed a microservice-based architecture for an IoT-based disaster rescue system. The authors discussed the benefits of using microservices, such as modularization, fault tolerance, and scalability, and demonstrated how the proposed architecture can be implemented in the context of a disaster rescue system. In addition, self-adaptive systems have been used to enhance the fault tolerance and resilience of microservices in emergency response systems. In their paper, "A self-adaptive microservice architecture for emergency response systems," Shi et al. (2020) **[7]** proposed a self-adaptive microservice architecture that can dynamically adjust the number of microservices based on the system workload and resource availability. The authors demonstrated the effectiveness of their proposed architecture in the context of an emergency response system. Other research studies have focused on the development of distributed orchestration algorithms for microservices in emergency response systems. For example, in the paper, "Distributed Orchestration Algorithm for Microservices in Emergency Response Systems," Jiang et al. (2021) **[8]** proposed a distributed orchestration algorithm for microservices in emergency response systems. The authors demonstrated the effectiveness of their algorithm in the context of a flood emergency response system. Furthermore, containerization technology has been extensively used in emergency response systems to enable the deployment of microservices. In the paper, "Containerization for Microservice Architectures," Abu-Libdeh et al. (2016) **[9]** discussed the benefits of using containerization technology in microservice architectures and demonstrated how containerization can be used to deploy microservices in the context of emergency response systems. In summary, the development of an autonomous, distributed micro service system that can keep services running as best as possible even when the network connection is poor and there are limited computing resources available is a critical need in emergency response systems. Microservices and containerization technology offer several benefits, including scalability, modularity, and fault tolerance, and can be used to develop an effective emergency response system. Additionally, self-adaptive systems and distributed orchestration algorithms can enhance the fault tolerance and resilience of microservices in emergency response systems.

Exploring Alternative Data Transmission Methods

Building microservice servers offers several benefits such as scalability, flexibility, modularity, and fault tolerance. By breaking down a monolithic application into smaller, independent services, it becomes easier to scale and deploy specific services as needed. This modularity also makes it easier to update and maintain individual services without affecting the entire system. Additionally, the fault tolerance of a microservice architecture means that if one service fails, the rest of the system can continue to function.

On the other hand, transmitting data without an internet connection using methods such as satellite or radio communications offers the advantage of being able to transmit data even in areas where the internet is not available or the network connection is poor. Additionally, satellite communication was the most reliable method for transmitting data during disaster scenarios **[10]**.These methods also offer greater independence and are less reliant on infrastructure and resources, making them more suitable for emergency response situations. However, these methods may have their own limitations such as lower bandwidth, higher latency, and higher costs. The quality of the transmission may also be affected by environmental factors such as interference or weather conditions **[11]**.

III. GOALS AND CHALLENGES

The primary goal of this research project is to design and implement an autonomous, distributed microservice system that can keep services running as best as possible even when the network connection is poor and there are limited computing resources available, while addressing several key challenges, including security and privacy, interoperability, scalability, and usability. Specifically, the system must be designed to operate effectively in low-bandwidth or disconnected environments, ensuring the availability of critical services for emergency responders even when network connectivity is limited and operating on limited computing resources. Moreover, the system must be resilient and self-adaptive in the face of changing conditions, such as when a fire truck drives into a valley with no radio connection to the headquarters. To meet these challenges, the project will involve the implementation of a containerized microservice test environment and the design and implementation of experiments to test the orchestration system. The project will also aim to design and test new distributed orchestration algorithms to improve the system's resilience and adaptability.

The ultimate goal is to develop a system that can better support emergency responders in critical situations, ensuring that they have access to the information and resources they need to save lives and protect communities, while enforcing strict access control policies and ensuring the confidentiality, integrity, and availability of sensitive data. Finally, the system must be compatible and interoperable with existing emergency response systems, scalable and extensible to handle increased workload and evolving needs, and user-friendly with a clear and concise interface to support emergency responders under high stress and pressure.

IV. TIMELINE

Week 1-2: Project Planning

- Define project scope and objectives
- Create project plan and timeline
- Identify project resources
- Determine project deliverables and milestones

Week 3: Design Document

- Conduct literature review on microservice systems in emergency response
- Define the microservice architecture for the emergency response system
- Define the containerization strategy for the microservices
- Document design decisions and rationale in a design document

Week 4-5: Prototype Development

- Develop a proof-of-concept prototype of the microservice system
- Create initial container images for each microservice
- Implement basic service discovery and communication protocols

Week 6-8: System Development

- Develop the microservices for the emergency response system
- Create additional container images for new microservices
- Implement advanced service discovery and communication protocols
- Integrate with external systems and data sources

Week 9-10: Testing and Validation

- Test the microservice system in a simulated environment
- Evaluate system performance under different loads and network conditions
- Identify and fix any bugs or performance issues
- Validate the system against the project objectives and requirements

Week 11: Documentation and Presentation

- Document the microservice system architecture and implementation
- Prepare project presentation materials and practice presenting
- Deliver final project presentation

Week 12: Final Report and Wrap-Up

- Create a final project report that includes the project plan, design document, implementation details, testing results, and project outcomes
- Conduct a project retrospective to identify lessons learned and areas for improvement
- Complete any final project tasks and deliverables

REFERENCES

[1] He, Q., Cui, H., Liu, X., & Chen, H. (2019). Enhancing emergency response efficiency: A review of communication and information technologies for disaster management. IEEE Access,7,77154-77166. https://doi.org/10.1109/ACCESS.2019.2923991

[2] Legg, S., Olsen, J. R., El-Tawab, S., Wang, W., & Banerjee, A. (2016). Communication in disaster response: Examining the effects of different communication modalities on the formation of spontaneous response groups. Journal of Homeland Security and Emergency Management, 13(4), 829-845. https://doi.org/10.1515/jhsem-2015-0084

[3] Ranjan, R., Jayaraman, P. P., Nepal, S., & Buyya, R. (2016). A survey on autonomic computing: architectures, models, and applications. Journal of Parallel and Distributed Computing, 95, 38-58.

[4] Zhang, Y., Zhu, L., Zhao, Y., Yang, X., & He, J. (2020). Self-healing microservice architecture for cloud-based systems. Journal of Cloud Computing, 9(1), 1-16.

[5] Hu, L., Liu, J., & Zhou, C. (2019). Towards microservice-based emergency response systems. IEEE Access, 7, 16323-16334.

[6] Zhang, L., Xiong, N., & Zhao, H. (2019). Microservice-based Internet of Things architecture for disaster rescue. International Journal of Distributed Sensor Networks, 15(11), 1550147719881219.

[7] Shi, Y., Wang, S., & Jiang, Y. (2020). A self-adaptive microservice architecture for emergency response systems. Journal of Ambient Intelligence and Humanized Computing, 11, 1293-1304.

[8] Jiang, Z., Xiong, N., & Zhang, L. (2021). Distributed orchestration algorithm for microservices in emergency response systems. Journal of Ambient Intelligence and Humanized Computing, 12, 1311-1324.

[9] Abu-Libdeh, H., Princehouse, L., & Weatherspoon, H. (2016). Containerization for microservice architectures. In Proceedings of the 1st International Workshop on Container Technologies and Container Clouds (pp. 9-14). ACM.

[10] Ozdemir, S., Cil, E., Ozturk, A., & Gonen, B. (2018). Evaluation of different data transmission methods in disaster management systems. Journal of Network and Computer Applications, 120, 76-84.

[11] Yu M, Yang C and Li Y (2018) Big Data in Natural Disaster Management: A Review. Geosciences 8(5). MDPI AG: 165. DOI: 10.3390/geosciences8050165.