Chapter 2
Zero-Loss Sensor Data Aggregation: A Rust-Based Operating System Core for Concurrent Data Processing

Hong Rui Zhou
The Hong Kong Polytechnic University, Hong Kong

Xiang Li
The Hong Kong Polytechnic University, Hong Kong

Min Hao Ling
The Hong Kong Polytechnic University, Hong Kong

Yi Ran Wu
http://orcid.org/0009-0004-8040-8379
The Hong Kong Polytechnic University, Hong Kong

Tong Yao Li
The Hong Kong Polytechnic University, Hong Kong

Cong Wu
http://orcid.org/0000-0002-8652-3501
Liaoning Technical University, China

ABSTRACT

This project initially aims to construct a consequent data processing pipeline. It should have the ability to collect readings from multiple simulated sensors, and perform a real-time statistical aggregation. It should also store the results durably, and serves them through a web interface. The system architecture is organized as a Rust workspace with four key components. The sensor_sim works for data generation, os_lib for buffer management, gateway for the main entry point, and dashboard for the web presentation. The core innovation in this project is the synchronization implementation. A lock-free, non-copy circular buffer is used, and enables the zero-blocking data transfer between the producer and consumer threads.

DOI: 10.4018/979-8-2600-1101-0.ch002

1. INTRODUCTION

1.1 Problem Statement and Motivation

The modern infrastructure systems mostly relies on the interconnected sensor networks, and the continuous data collection and analysis. These systems are often compose of hundreds of sensors which can generate high speed data streams. These streams must be collected, stored, processed and presented with minimal latency and zero data loss.

Therefore, at the core of the system, there is a fundamental operating system challenge: managing the data flow between system components with different and mismatched speeds and capacities. Sensors often contain small internal buffers because of hardware constraints, the buffers can only hold the readings for several seconds then they may overflow. Moreover, other processing components like the aggregation engine and the storage system often operate at different speed, their capacities are also different. The system should have control of the reading speed. This problem is even more serious when concurrency is required. Multiple sensors generate data together, and multiple processing threads have to coordinate when accessing the shared buffers. The web server should serve aggregated results to clients, and new data is arriving at the same time.

The distributed sensor data aggregation platform is implemented to solve these problems. The core of the implementation is a lock-free non-copy circular buffer. It works as a fundamental synchronization helper. This chapter aims to solve the real-world data ingestion problem. It makes use of the advanced OS concepts like lock-free synchronization, atomic operations, and producer-consumer pattern to achieve this.

1.2 Chapter Objectives

To construct a complete sensor data aggregation platform is the overall objective of this chapter. It should fulfill the strict requirements for data integrity, concurrency, and efficiency. The system should collect data from multiple simulated sensors timely (to prevent overflow). Then it should process the data through a pipeline without any data loss. This overall objective can be decomposed into four component-level objectives.

First, the buffer management component. It must provide a lock-free and thread-safe queue which can perform the sensor reading and data processing separately. It must create dedicated reader threads for each sensor, and poll them sufficiently to prevent overflow. It should also maintain a high-performance circular buffer which can support the non-copy data types.