Keep-up Production in JUNO Offline Data Processing

Weiqing Yin[1,2], Tao Lin[1,2], Yizhou Zhang[1,2]
[1]Institute of High Energy Physics
[2]University of Chinese Academy of Sciences
yinwq@ihep.ac.cn On behalf of JUNO collaboration

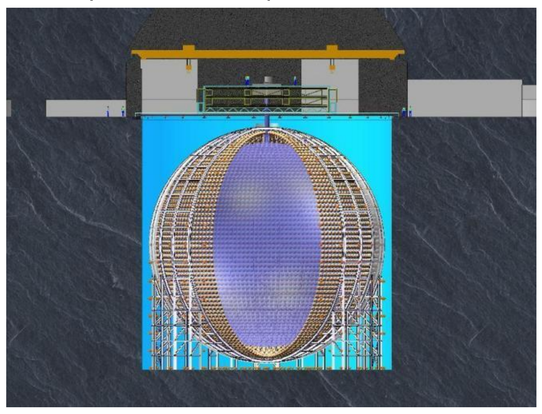
Introduction

Jiangmen Underground Neutrino Observatory (JUNO):

The Jiangmen Underground Neutrino Observatory is an important scientific experiment located in Jiangmen, Guangdong Province, China[1].

It aims to determine the neutrino mass hierarchy and provide precision measurements of neutrino oscillation parameters. Additionally, JUNO will explore other areas of neutrino physics, such as observing supernova neutrinos and studying the atmospheric, solar neutrinos and geo-neutrinos.

The observatory utilizes a massive 20 kiloton liquid scintillator detector located approximately 700 meters underground, which helps to minimize the interference from cosmic rays and other background radiation.

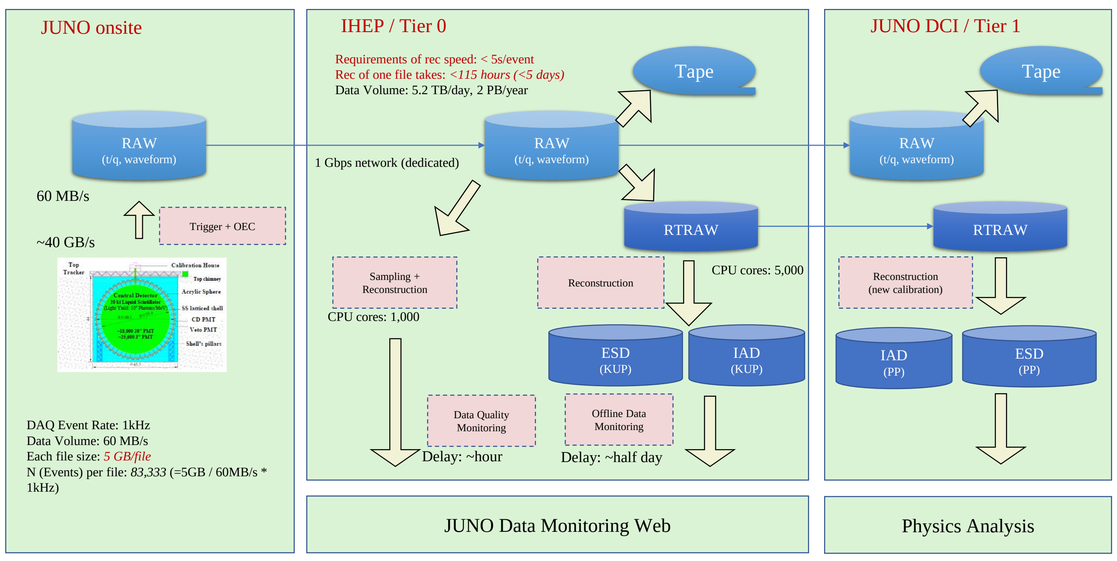


JUNO Detector Concept

Offline Data Processing

Workflow: Involving data transfer, reconstruction, grid computing, and long-term data preservation. The entire workflow embodies complexity due to the intricate relationships between these stages[1].

Data volume: Operating over 20-30 years, approximately 2PB of raw data annually.



JUNO Offline Data Processing Workflow

KUP Pipeline

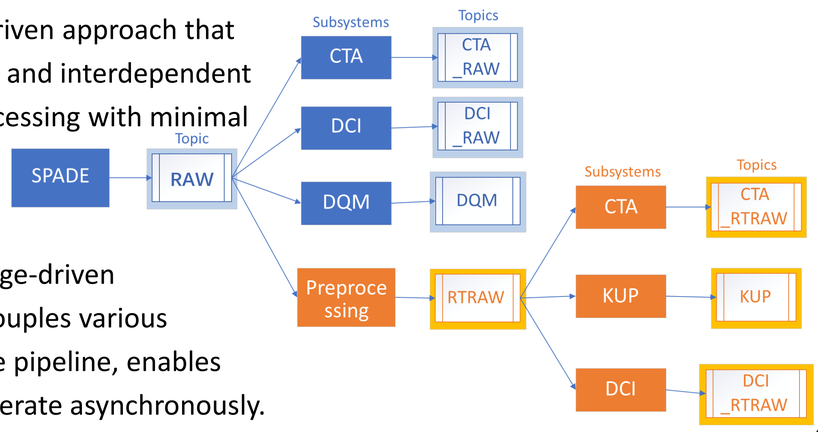
The Keep Up Production (KUP) pipeline is designed to handle the challenges posed by the complexity and volume of data.

Purpose and Objectives

Establish a pipeline-driven approach that manages the intricate and interdependent steps of raw data processing with minimal human intervention.

Architecture

KUP employs a message-driven architecture that decouples various subsystems within the pipeline, enables each subsystem to operate asynchronously.

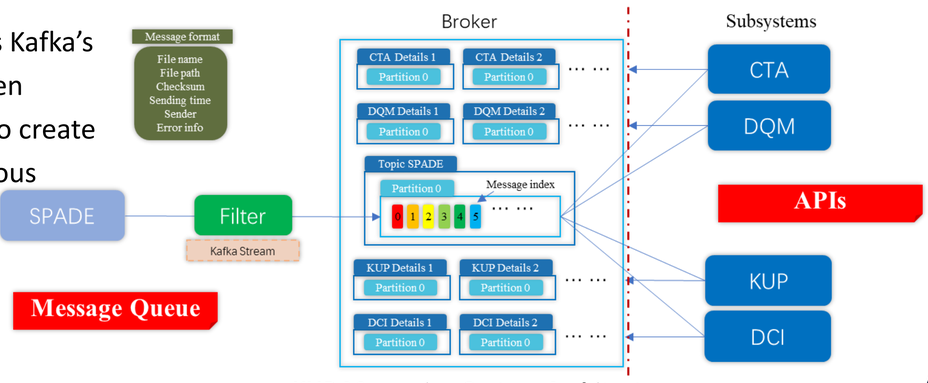


Pipeline Architecture

Kafka in KUP

Apache Kafka is a distributed event streaming platform known for its high throughput, scalability, fault tolerance, and durability[2].

KUP leverages Kafka's message-driven architecture to create an asynchronous processing environment.



KUP Messaging System Architecture

Job Management

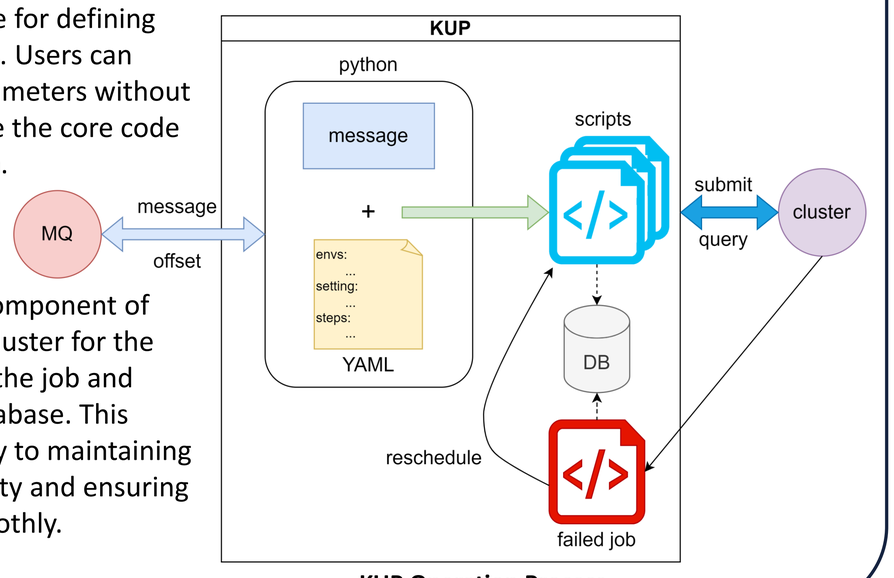
Job Submission

The Job Submission process within the KUP is critical for automating and managing the execution of data processing tasks. This process is designed to streamline job management, replacing manual work with automation.

YAML is a human-readable data serialization format that uses indentation to represent data hierarchy, making it easy to read and write for humans. The use of YAML provides a flexible structure for defining job configurations. Users can easily modify parameters without needing to change the core code of the KUP system.

Job Monitoring

The Monitoring component of KUP queries the cluster for the running status of the job and saves it in the database. This functionality is key to maintaining operational stability and ensuring that jobs run smoothly.



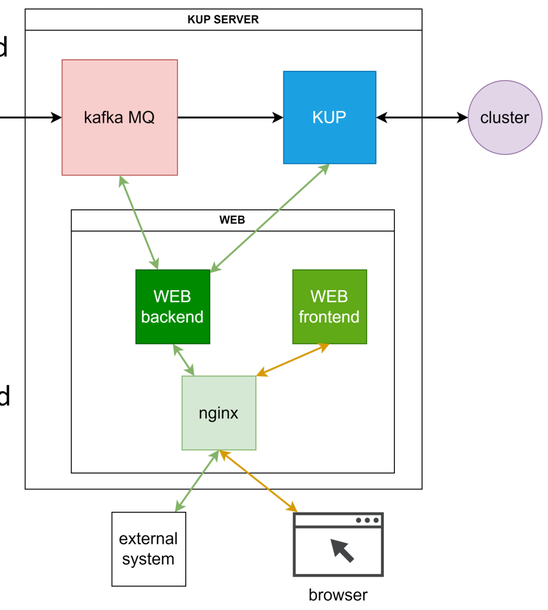
KUP Operation Process

Web Portal

The web design component of the KUP pipeline is key to providing an intuitive and efficient interface for users. This interface facilitates effective monitoring, management, and interaction with the KUP system.

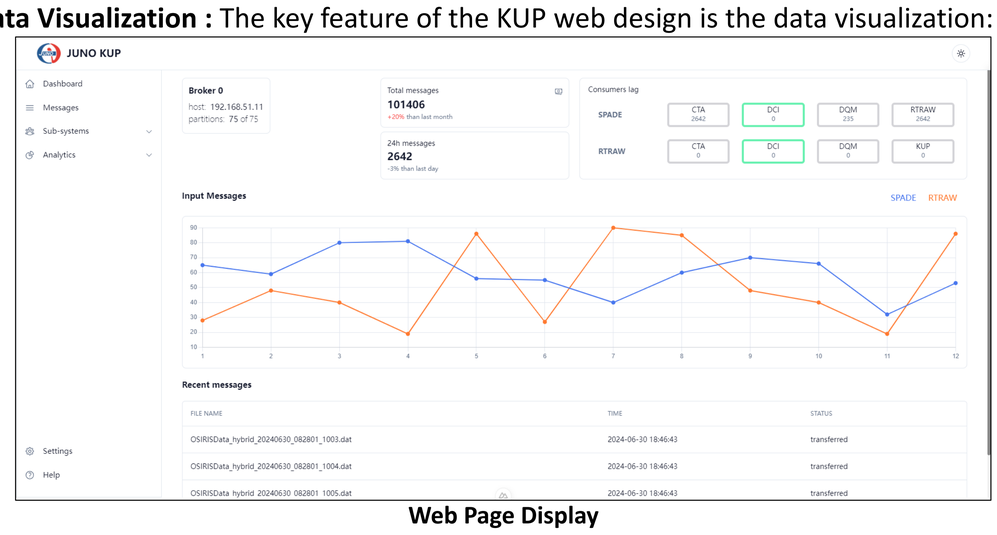
Architecture: The KUP web application is built on a separated front-end and back-end architecture , which enhances the system's modularity and flexibility.

- Front-End : The user interface is developed using Vue.js , a progressive JavaScript framework known for its simplicity and powerful features.
- Back-End : The back-end are built using FastAPI , a modern, high-performance framework for building APIs with Python.
- Reverse Proxy : Nginx is used as a reverse proxy server to manage client requests and route them appropriately between the front-end and back-end applications.
- Docker : All components of the KUP web application—front-end, back-end, and Nginx—are containerized using Docker . This approach ensures isolation, scalability and portability.



Web Architecture

Data Visualization : The key feature of the KUP web design is the data visualization:



Web Page Display

- Real-Time Dashboard : The dashboard provides real-time visual representations of message throughput, consumers lag, broker state. This includes graphical displays, enabling users to quickly assess the state of the pipeline.
- Alert Systems : Visual cues such as color-coded notifications indicate job statuses and alerts for any issues, providing instant recognition of critical situations.

Reference

[1] Abusleme A, Adam T, Ahmad S, et al. JUNO physics and detector[J]. arXiv preprint arXiv:2104.02565, 2021.
[2] Garg N. Apache kafka[M]. Birmingham, UK: Packt Publishing, 2013.