Architecture Guide

Community 1.5.1 release

This guide details the architectural specifics of Jumbune, helps you understand core layers, major components, and their interactions.

Table of Contents

Introduction

Jumbune the Map Reduce Flow Profiler is intended to help Map Reduce Developers and Hadoop Cluster Administrators. It comprises of various modules, namely, Hadoop Job Flow Analyzer, HDFS Data Validator and Hadoop Job Profiler and Cluster Monitoring. This guide is intended to help the developers understand the underlying architecture and get acquainted with the design of Jumbune.

Building Blocks of Jumbune

Jumbune architecture can be classified into the following major blocks:

- Request Handler
- Execution
- Processors
- Components
- Remoting
- Jumbune Agent

Request Handler

This layer supports UI based execution. Jumbune execution is triggered by the user in a sequential workflow, expressed in a simple JSON configuration. The JSON configuration consists of instructions for individual Jumbune components. Jumbune presents intuitive self-explanatory reports for the submitted workflow.

Execution

This layer is responsible for the bifurcation of the flow of the Web or Shell based execution of the Hadoop job. As per the request received the Core Executor either calls Shell Executor if it receives request for running the job through shell(via console) or the HttpExecutor if it receives request for running the job through web, then this executors are responsible for calling the Base Processor for handling the flow of individual components.

Processors

This layer consists of the necessary processors for handling different components of Jumbune according to the specified component in the JSON. Base Processor handles the request received from the executors and directs the flow of control to the underlying component specific processors.

Components

This layer comprises of the following components:

- **Flow Debugging**: Flow Debugger verifies the flow of input records in user's map reduce implementation and provides In-depth analysis of Map Reduce flow thereby helping to identifying the bottle necks in the map reduce phase.
- **Cluster Monitoring**: Cluster Monitor provides on demand Hadoop JMX and Node resource statistics. This component provides network latency across Hadoop nodes, per file per node wise replica placement, rack aware nodes view as well as data load partition across the nodes.
- **JVM Profiling**: JVM Profiler provides Map Reduce Phase wise stats which includes per job performance of JVM, data flow rate and resource usage. This component also provides per job heap sites and CPU cycles for Mapper and Reducer.
- **HDFS Data Validation**: HDFS Data Validation validates inconsistencies in the HDFS data in the form of null, data type & regex checks.
- **Data Quality Timeline**: Data Quality timeline traces the conservation of data quality over a period of time, even in massive data offloading environment.
- **Data Profiling**: Data Profiling computes statistic assessment of data values within a data set for consistency, uniqueness and logic.

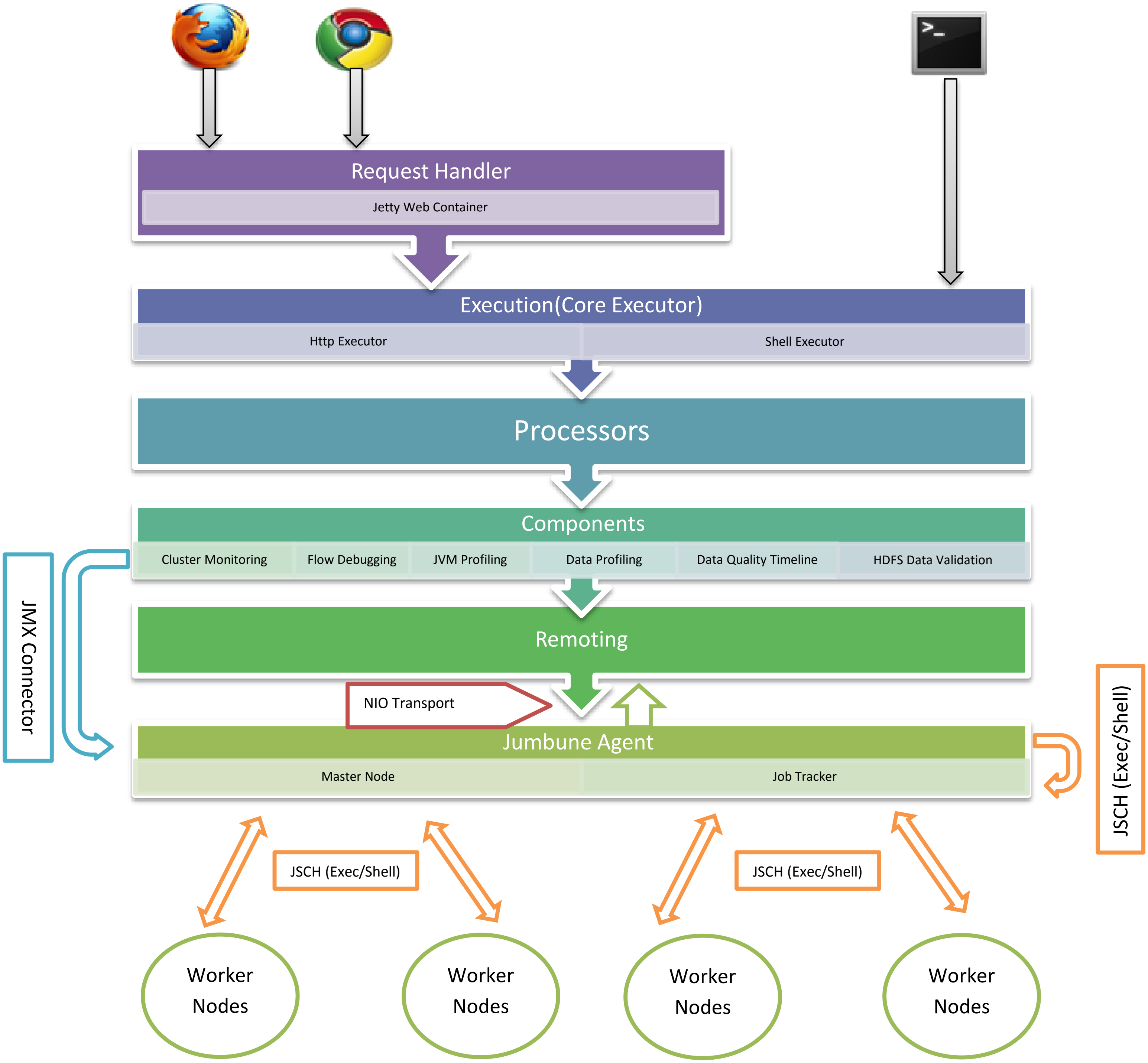
Remoting

Remoting is the underlying transport layer of the Jumbune architecture that interacts Hadoop via non-blocking NIO Socket Transport through Jumbune Agent. This layer handles all the instructions provided by the above components to process the desired outcome.

Jumbune Agent

Jumbune Agent, an ultra-light weight component is responsible for handling all the interactions between the master and the worker nodes through jsch execution/shell mode, hence giving the decoupled installation from Hadoop. This layer receives the instructions from the Remoting layer to perform various operations on Hadoop.

High Level Architecture Diagram of Jumbune



Request Handler
Jetty Web Container
Execution(Core Executor)
Http Executor
Shell Executor
Processors
Components
Cluster Monitoring
Flow Debugging
JVM Profiling
Data Profiling
Data Quality Timeline
HDFS Data Validation
Remoting
JMX Connector
NIO Transport
Jumbune Agent
Master Node
Job Tracker
JSCH (Exec/Shell)
JSCH (Exec/Shell)
JSCH (Exec/Shell)
Worker Nodes
Worker Nodes
Worker Nodes
Worker Nodes