Big Data Workflow Scheduler
Introducing Apache DolphinScheduler

Apache Software Foundation Member
Apache DolphinScheduler PMC

William Kwok
Linked-in: William Kwok
E-mail: guowei@apache.org

About Me

William Kwok

Apache Software Foundation Member

Apache IPMC Member

PMC of Apache DolphinScheduler

Mentor of Apache SeaTunnel(incubating)

Founder of ClickHouse China Community

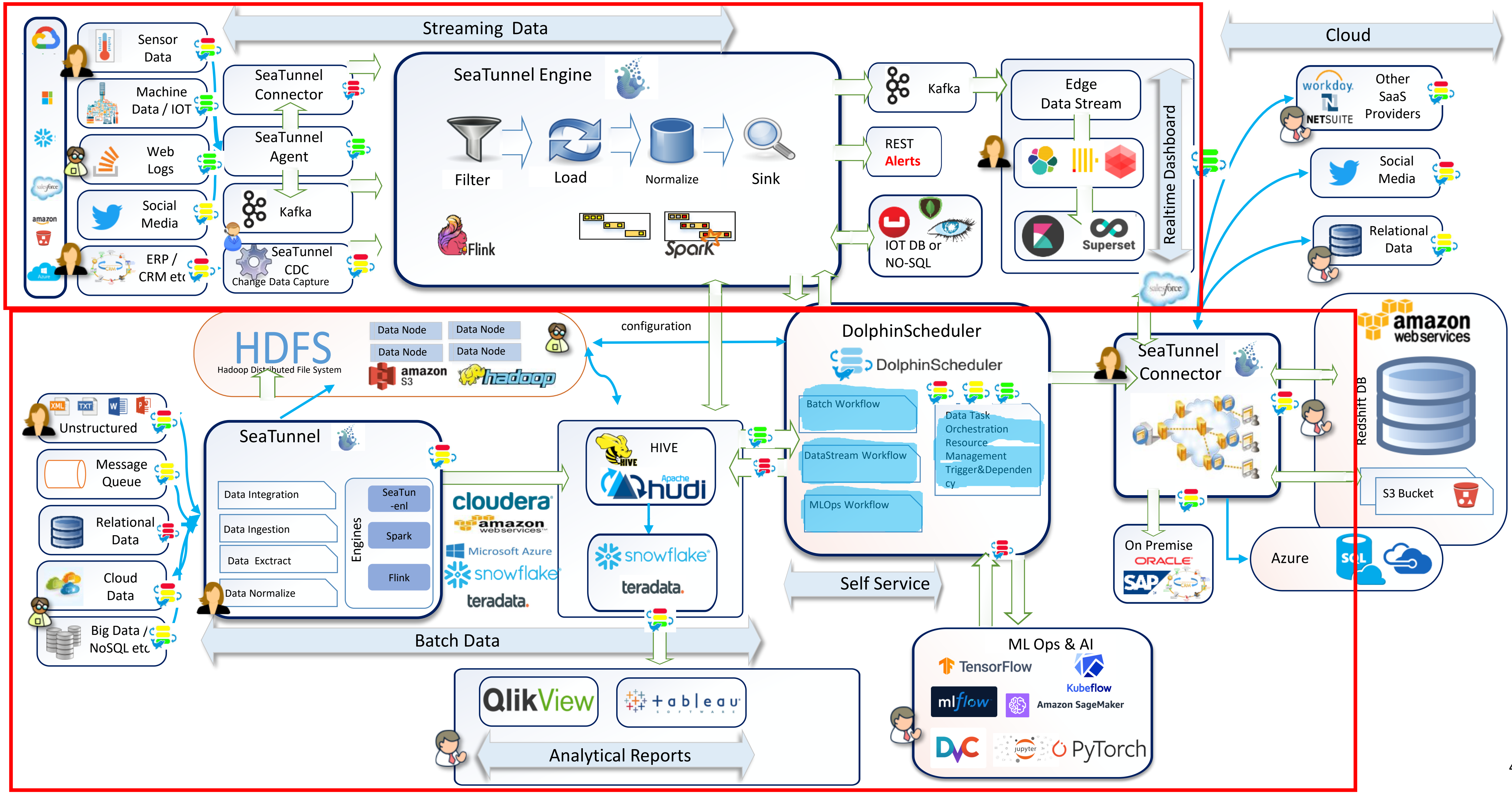
Track Chair of Workflow/Data Governance of Apache Con Asia 2021/2022

William used to be CTO of Analysys and Senior Big Data Director of Lenovo, Wanda, CICC, IBM, and Teradata.
He has more than 20+years of experience in big data technology and data management.

Agenda

- Introduction of DolphinScheduler
- 2.0 & 3.1.0 New Features
- User Case – Cisco Webx

Apache Projects in Modern Data Stack & DataOps in Enterprise



DolphinScheduler: Cloud-Native Visual Scheduler Engine with High Stability

☆ Star 8.1k

Big Data Workflow Scheduling Pain Point

1. Multiple Task Units
2. Execution in Sequential Order
3. Dependencies

+

Enterprise Pain Point in Practical Environment

1. High Frequency
2. Mass Data & Task Volume
3. Cloud-Native

Script + Open-Source Toolkit

</> + cron

- Lack of Code Reusability
- Complicated Cluster Deployment & Expansion
- Frequent Updates Cause Instability

Other Schedulers

Azkaban · Apache Airflow

- Non-visual Design with Few Intelligence
- Unscalable to adapt large volume task
- Lack of Multi-Cloud Data Management

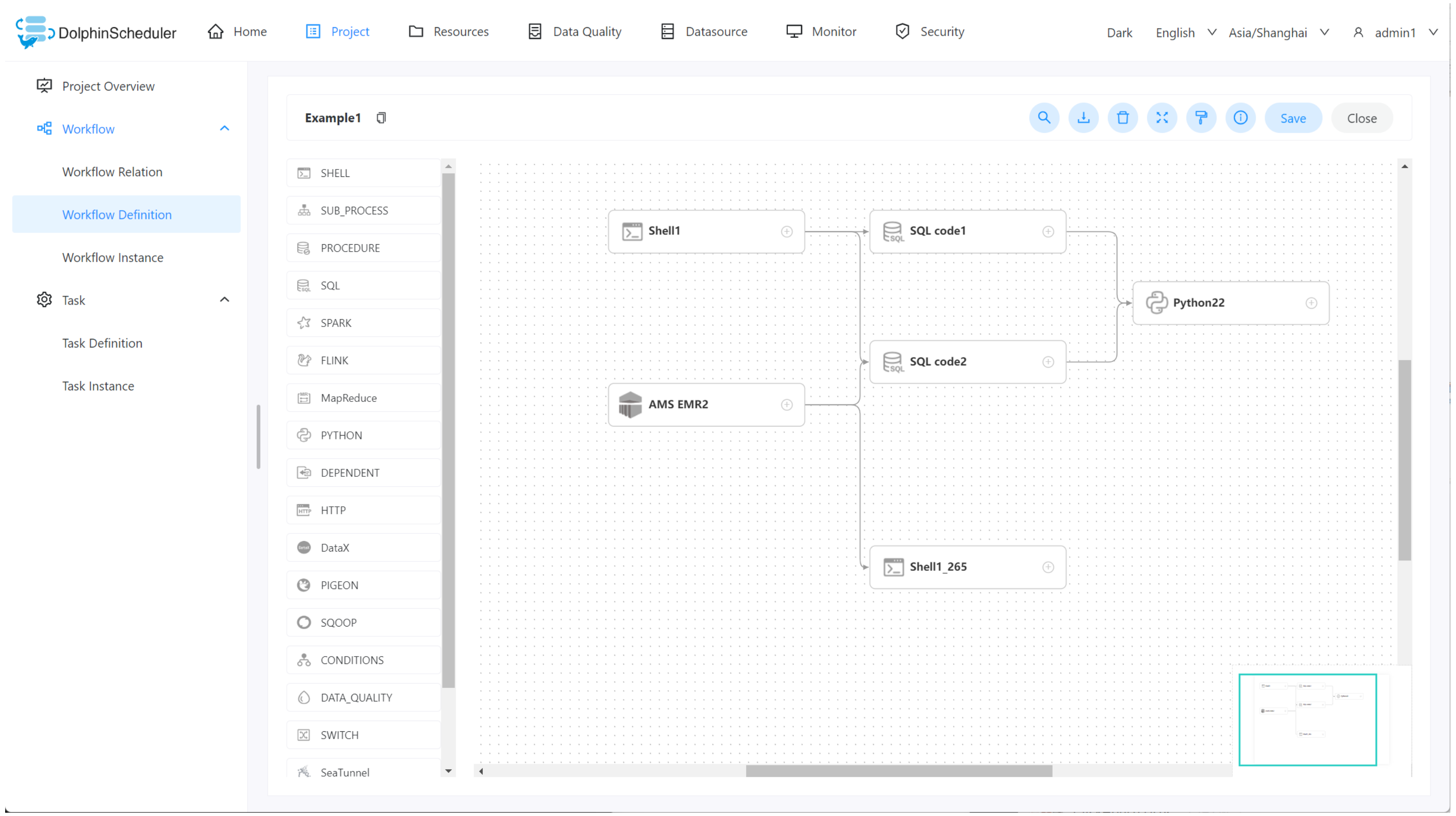
DolphinScheduler

DolphinScheduler

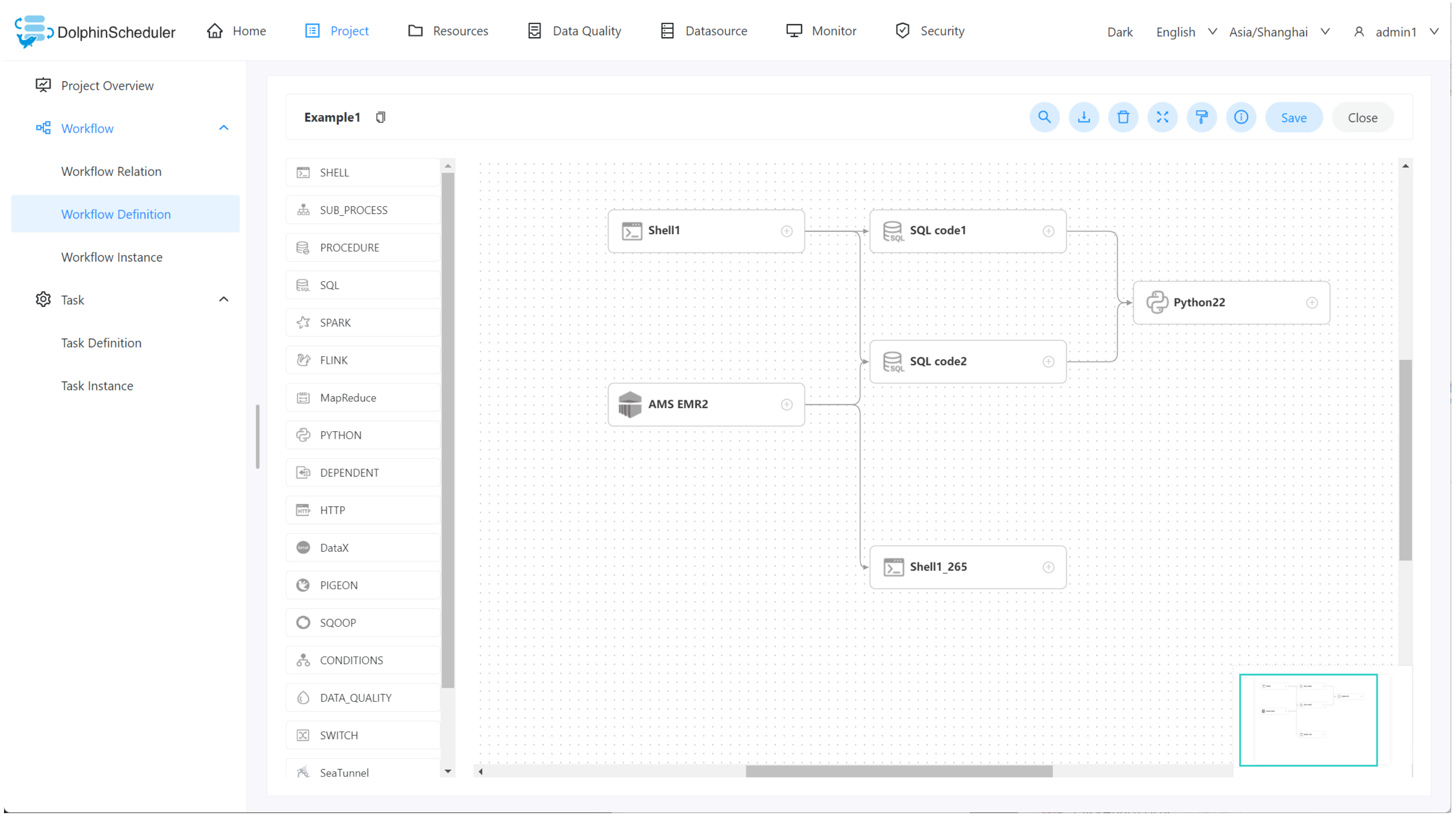
Task Scheduling with visualization and Various Tasking Categories	4 + yrs of Community
Decentralized Design Enables High Stability and Availability	350 + Contributors
Stably Supporting Millions of Data & Tasks Running Simultaneously	1000 + Active Users

Apache DolphinScheduler

DolphinScheduler is a distributed and extensible workflow scheduler platform with powerful DAG visual interfaces, dedicated to solving complex task dependencies in the data pipeline and providing various types of jobs.

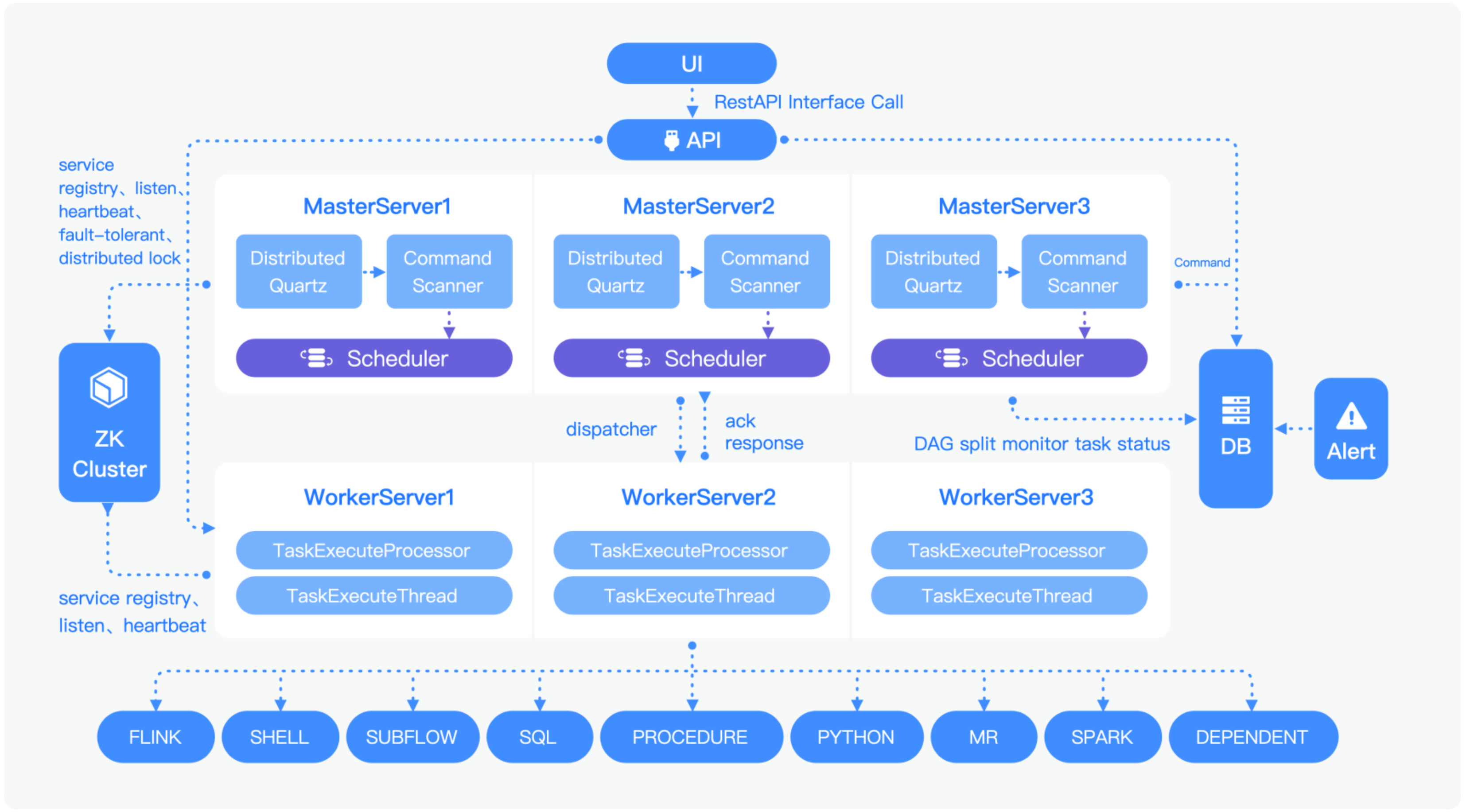


Drag&Drop To Create a WYSWYG Workflow without Code.

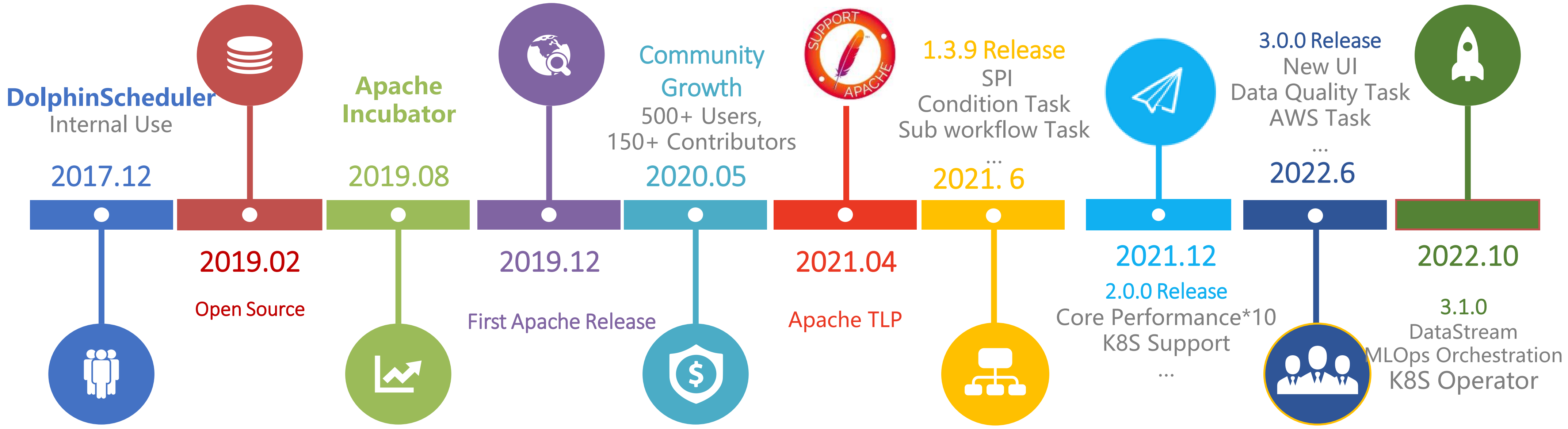


DolphinScheduler High Performance & High Stability with Multi Workers/Masters

Apache DolphinScheduler is dedicated to solving complex job dependencies in the data pipeline and providing various types of jobs available out of box.



Apache DolphinScheduler History



DolphinScheduler Is Trusted by Many Industry Leaders

DolphinScheduler Typical User Case

China Unicom originally used an enterprise scheduling system to support data processing and task scheduling of their global data platform in combination with Shell (HiveSQL) . After comparing Airflow, Azkaban, and other commercial scheduler, China Unicom finally chose DS.

- Met business and scheduling functional needs
- Met large data volume requirements
- Cost-effectiveness

1 High-Performance, High-Volume Task Scheduling

SHEIN originally used Airflow to schedule global tasks; however, Airflow has a centralized design and lack of visualization, and it was also unable to support K8S and globalized cloud-native deployment. Thus, SHEIN chose to migrate from Airflow to DolphinScheduler.

- Global cloud deployment, K8S support
- Decentralization to ensure stability
- Easy to use for data consumers without developer background

2 Global Cloud Deployment with Ease of Use for Data Consumers

Lizhi FM

Litchi FM used SQL/Shell/Python scripts and other big data components for their AI system, which was difficult both to use and to reuse. After using the AI development platform based on DS, Litchi FM abstracted the entire process of from data acquisition to model training and connected them with DAG through DS ‘s low-code IDE.

- Efficiently computing of massive big data tasks
- Reusable ML process
- DAG execution engine

3 AI/ML Orchestration

Agenda

- Introduction of DolphinScheduler
- 2.0 & 3.1.0 New Features
- User Case – Cisco Webx

DolphinScheduler 1.x and 2.x Features

- Tasks are associated as DAG form
- Real-time monitoring of task status

- Supports more than 20 task types such as Shell, MR, Spark, SQL, dependency, etc

- Supports workflow priority, task priority
- Global parameters and customized parameters

- Supports workflow scheduling, dependent, manual impacts, and pause/stop/resume

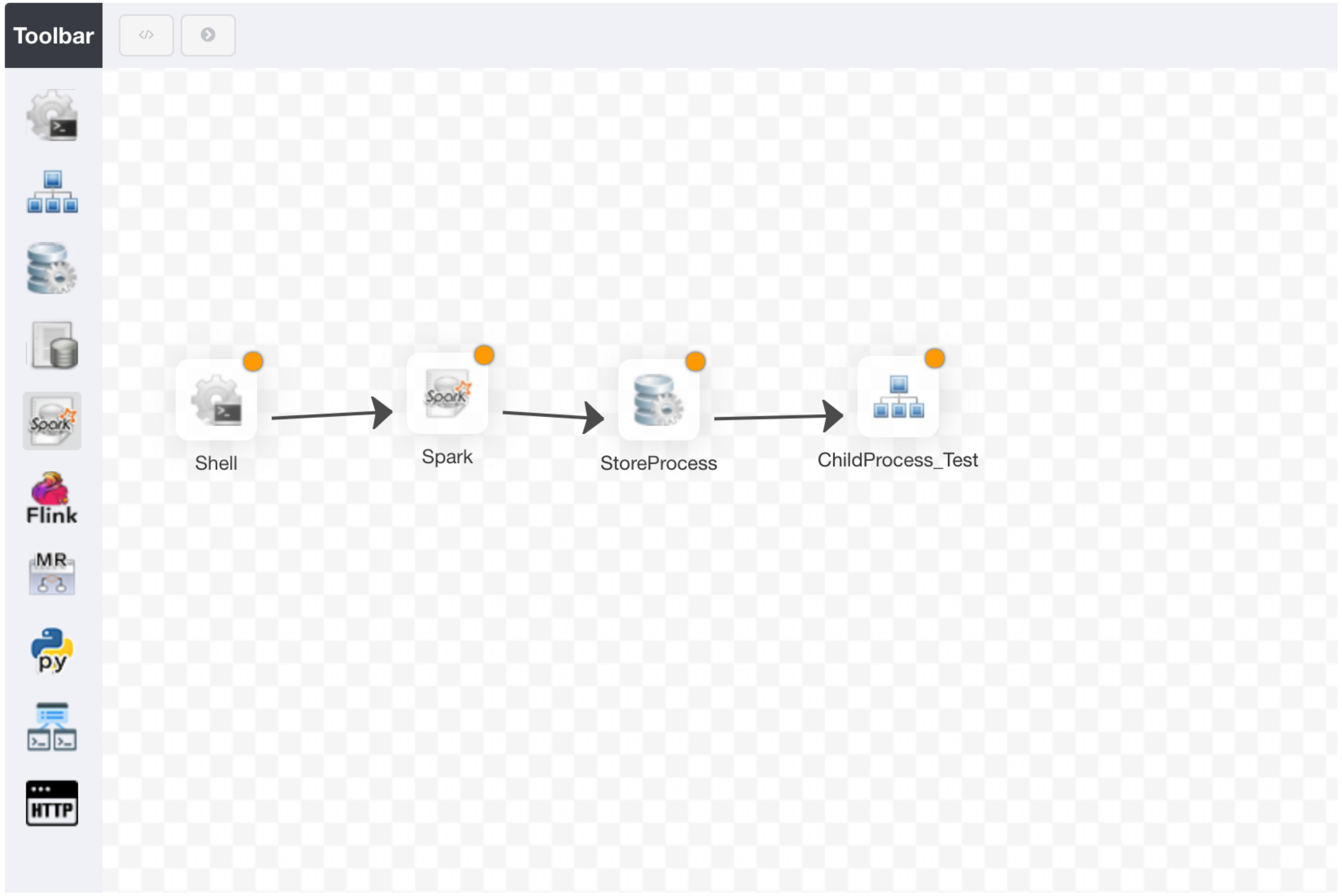
- Supports multi-tenancy, Multi-Projects
- online log viewing and resource online management

- Complete system monitoring, task timeout /failure alert.

- The decentralized design ensures the stability and high availability of the system

- Supports stable operation of 100,000 volume of data tasks per day

Workflow Management:
Visualized Drag-and-Drop Workflow Configuration

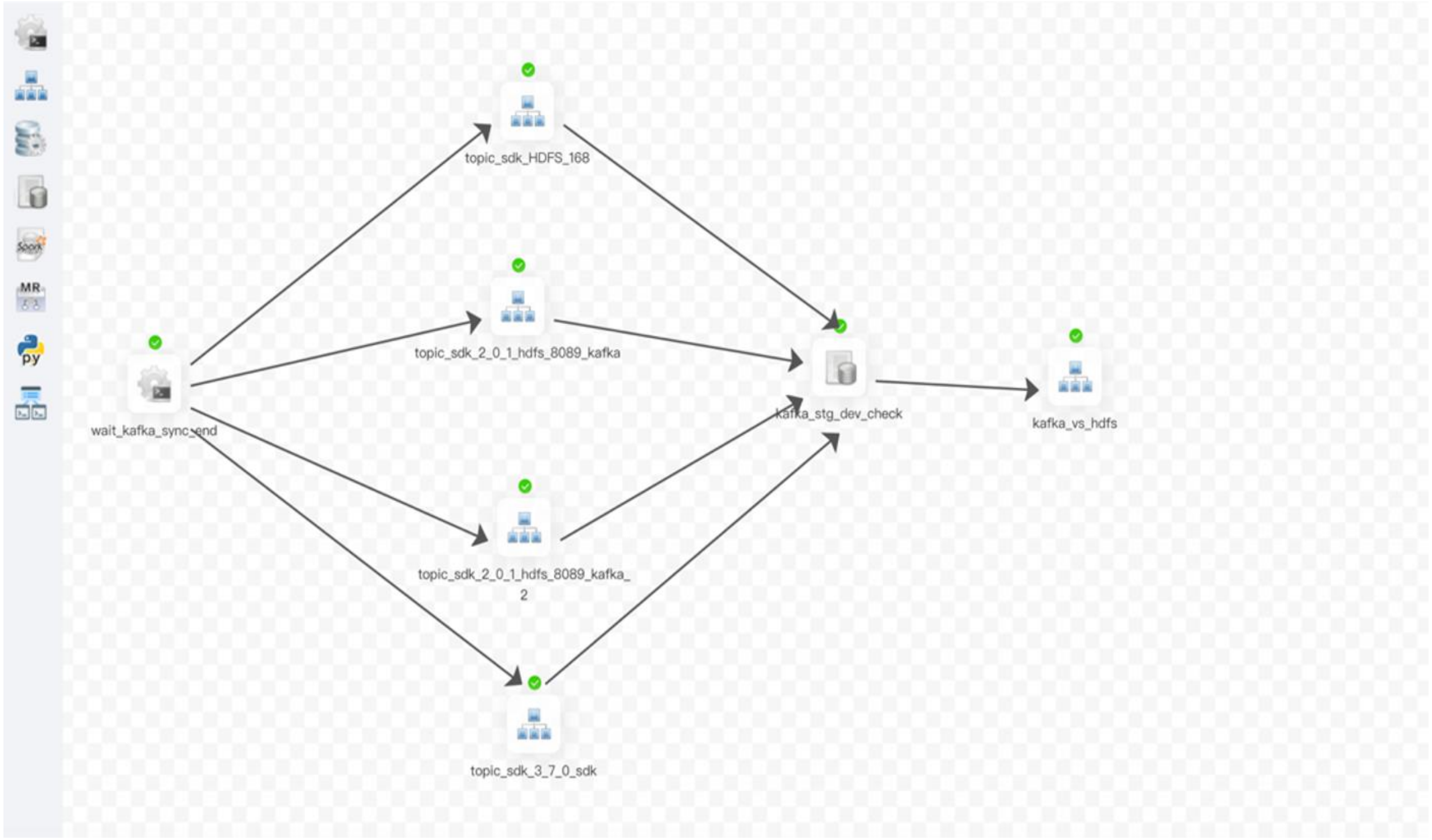


1. Visualized drag-and-drop optimizes task creation efficiency

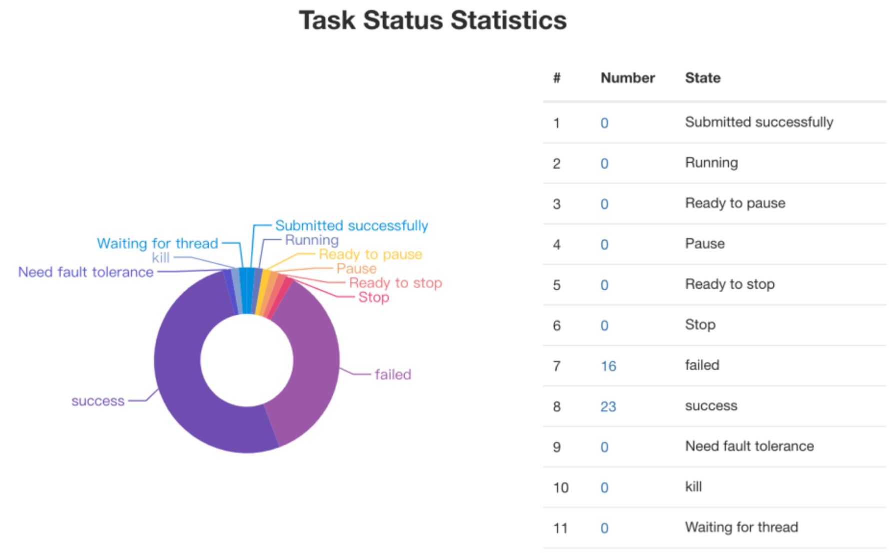
2. Support various task types: Shell, MR, Spark, SQL (MySQL, PostgreSQL, hive, spark SQL), Python, Sub_Process, Procedure, etc

3. Sub_Process
   - Sub_Process enables the reuse of **data resolve**, imports and data persistence, avoid repeated configurations

Visualization of Running Workflow



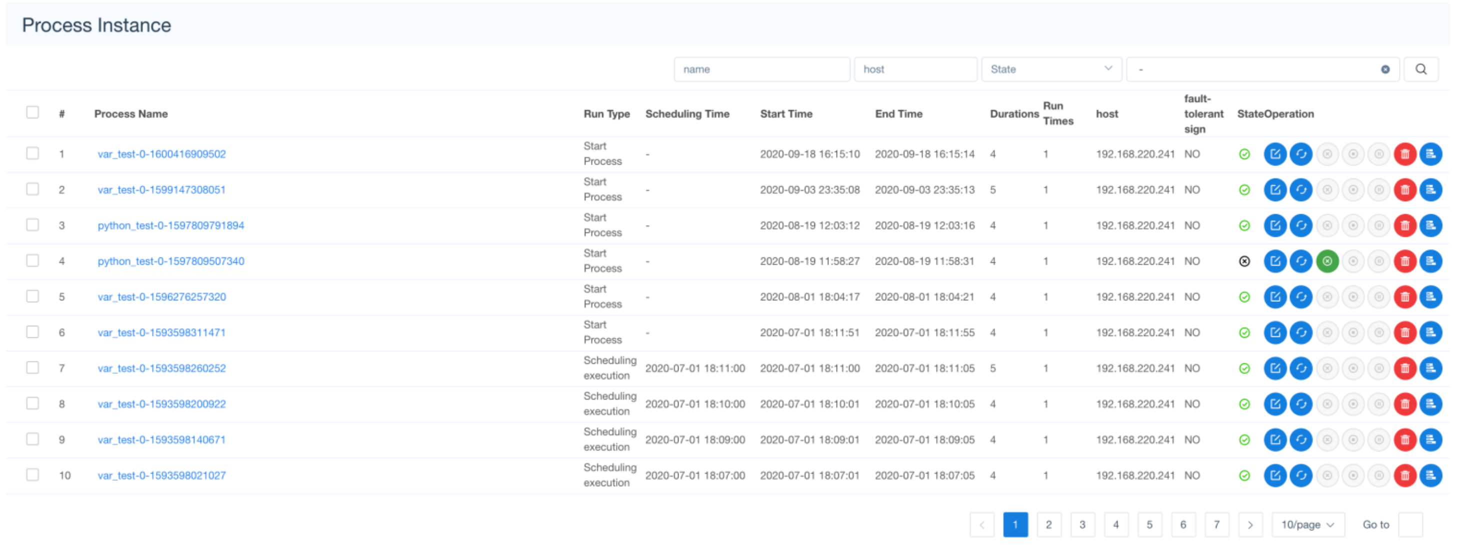
Task Management: Multi-Level Monitoring



Task Status Statistics

#	Number	State
1	0	Submitted successfully
2	0	Running
3	0	Ready to pause
4	0	Pause
5	0	Ready to stop
6	0	Stop
7	16	failed
8	23	success
9	0	Need fault tolerance
10	0	kill
11	0	Waiting for thread

Task status data statistics

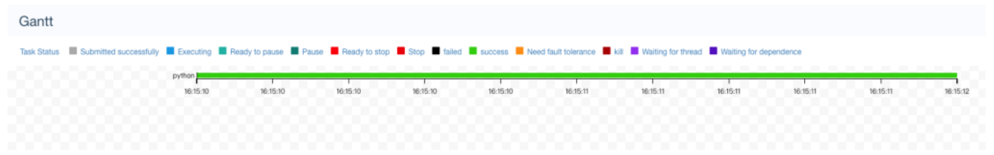


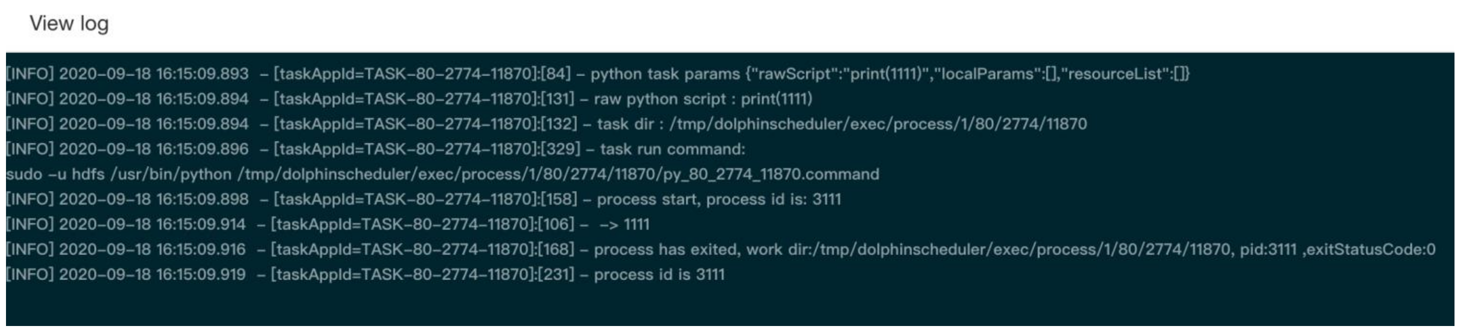
Process Instance

#	Process Name	Run Type	Scheduling Time	Start Time	End Time	Durations	Run Times	host	fault-tolerant sign	State	Operation
1	var_test-0-1600416909502	Start Process	-	2020-09-18 16:15:10	2020-09-18 16:15:14	4	1	192.168.220.241	NO	✓	
2	var_test-0-1599147308051	Start Process	-	2020-09-03 23:35:08	2020-09-03 23:35:13	5	1	192.168.220.241	NO	✓	
3	python_test-0-1597809791894	Start Process	-	2020-08-19 12:03:12	2020-08-19 12:03:16	4	1	192.168.220.241	NO	✓	
4	python_test-0-1597809507340	Start Process	-	2020-08-19 11:58:27	2020-08-19 11:58:31	4	1	192.168.220.241	NO	⊗	
5	var_test-0-1596276257320	Start Process	-	2020-08-01 18:04:17	2020-08-01 18:04:21	4	1	192.168.220.241	NO	✓	
6	var_test-0-1593598311471	Start Process	-	2020-07-01 18:11:51	2020-07-01 18:11:55	4	1	192.168.220.241	NO	✓	
7	var_test-0-1593598260252	Scheduling execution	2020-07-01 18:11:00	2020-07-01 18:11:00	2020-07-01 18:11:05	5	1	192.168.220.241	NO	✓	
8	var_test-0-1593598200922	Scheduling execution	2020-07-01 18:10:00	2020-07-01 18:10:01	2020-07-01 18:10:05	4	1	192.168.220.241	NO	✓	
9	var_test-0-1593598140671	Scheduling execution	2020-07-01 18:09:00	2020-07-01 18:09:01	2020-07-01 18:09:05	4	1	192.168.220.241	NO	✓	
10	var_test-0-1593598021027	Scheduling execution	2020-07-01 18:07:00	2020-07-01 18:07:01	2020-07-01 18:07:05	4	1	192.168.220.241	NO	✓	

< 1 2 3 4 5 6 7 > 10/page Go to

Process instance status view





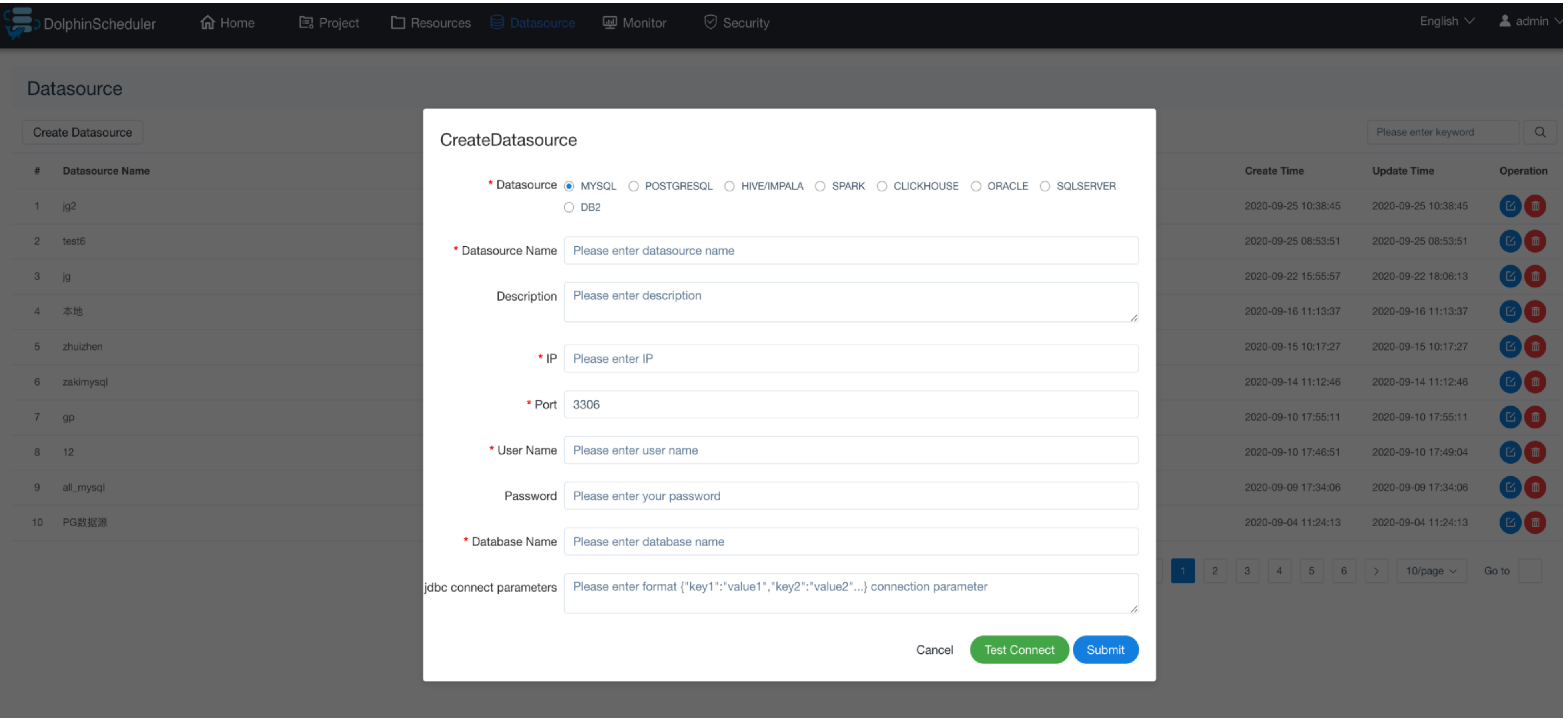
View log

```
[INFO] 2020-09-18 16:15:09.893  - [taskAppId=TASK-80-2774-11870]:[84] - python task params {"rawScript":"print(1111)","localParams":[],"resourceList":[]}
[INFO] 2020-09-18 16:15:09.894  - [taskAppId=TASK-80-2774-11870]:[131] - raw python script : print(1111)
[INFO] 2020-09-18 16:15:09.894  - [taskAppId=TASK-80-2774-11870]:[132] - task dir : /tmp/dolphinscheduler/exec/process/1/80/2774/11870
[INFO] 2020-09-18 16:15:09.896  - [taskAppId=TASK-80-2774-11870]:[329] - task run command:
sudo -u hdfs /usr/bin/python /tmp/dolphinscheduler/exec/process/1/80/2774/11870/py_80_2774_11870.command
[INFO] 2020-09-18 16:15:09.898  - [taskAppId=TASK-80-2774-11870]:[158] - process start, process id is: 3111
[INFO] 2020-09-18 16:15:09.914  - [taskAppId=TASK-80-2774-11870]:[106] -  -> 1111
[INFO] 2020-09-18 16:15:09.916  - [taskAppId=TASK-80-2774-11870]:[168] - process has exited, work dir:/tmp/dolphinscheduler/exec/process/1/80/2774/11870, pid:3111 ,exitStatusCode:0
[INFO] 2020-09-18 16:15:09.919  - [taskAppId=TASK-80-2774-11870]:[231] - process id is 3111
```

Tracking of task execution status

Task execution log online

Data-Source Management: Visualized Configuration and Multiple Data Compatibility



Datasource

Create Datasource

Please enter keyword

#	Datasource Name	Create Time	Update Time	Operation
1	jg2	2020-09-25 10:38:45	2020-09-25 10:38:45	
2	test6	2020-09-25 08:53:51	2020-09-25 08:53:51	
3	jg	2020-09-22 15:55:57	2020-09-22 18:06:13	
4	本地	2020-09-16 11:13:37	2020-09-16 11:13:37	
5	zhuizhen	2020-09-15 10:17:27	2020-09-15 10:17:27	
6	zakimysql	2020-09-14 11:12:46	2020-09-14 11:12:46	
7	gp	2020-09-10 17:55:11	2020-09-10 17:55:11	
8	12	2020-09-10 17:46:51	2020-09-10 17:49:04	
9	all_mysql	2020-09-09 17:34:06	2020-09-09 17:34:06	
10	PG数据源	2020-09-04 11:24:13	2020-09-04 11:24:13	

1 2 3 4 5 6 > 10/page Go to

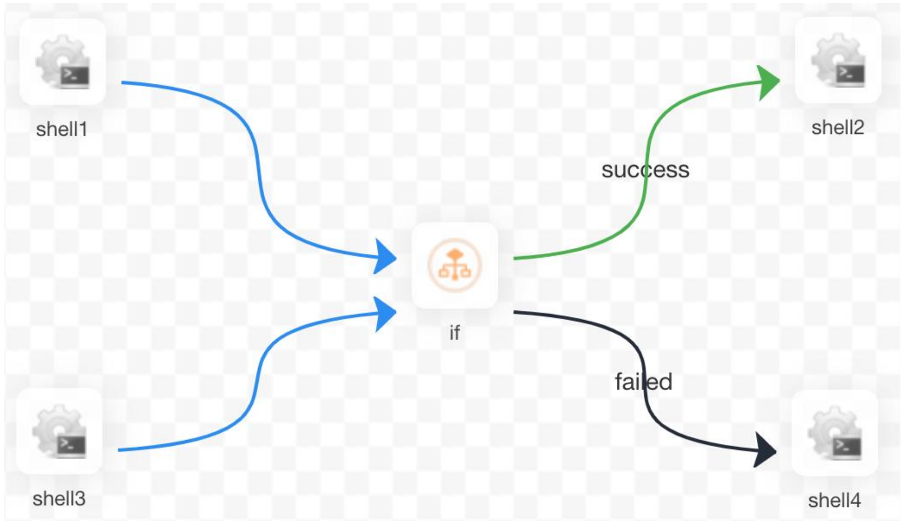
CreateDatasource

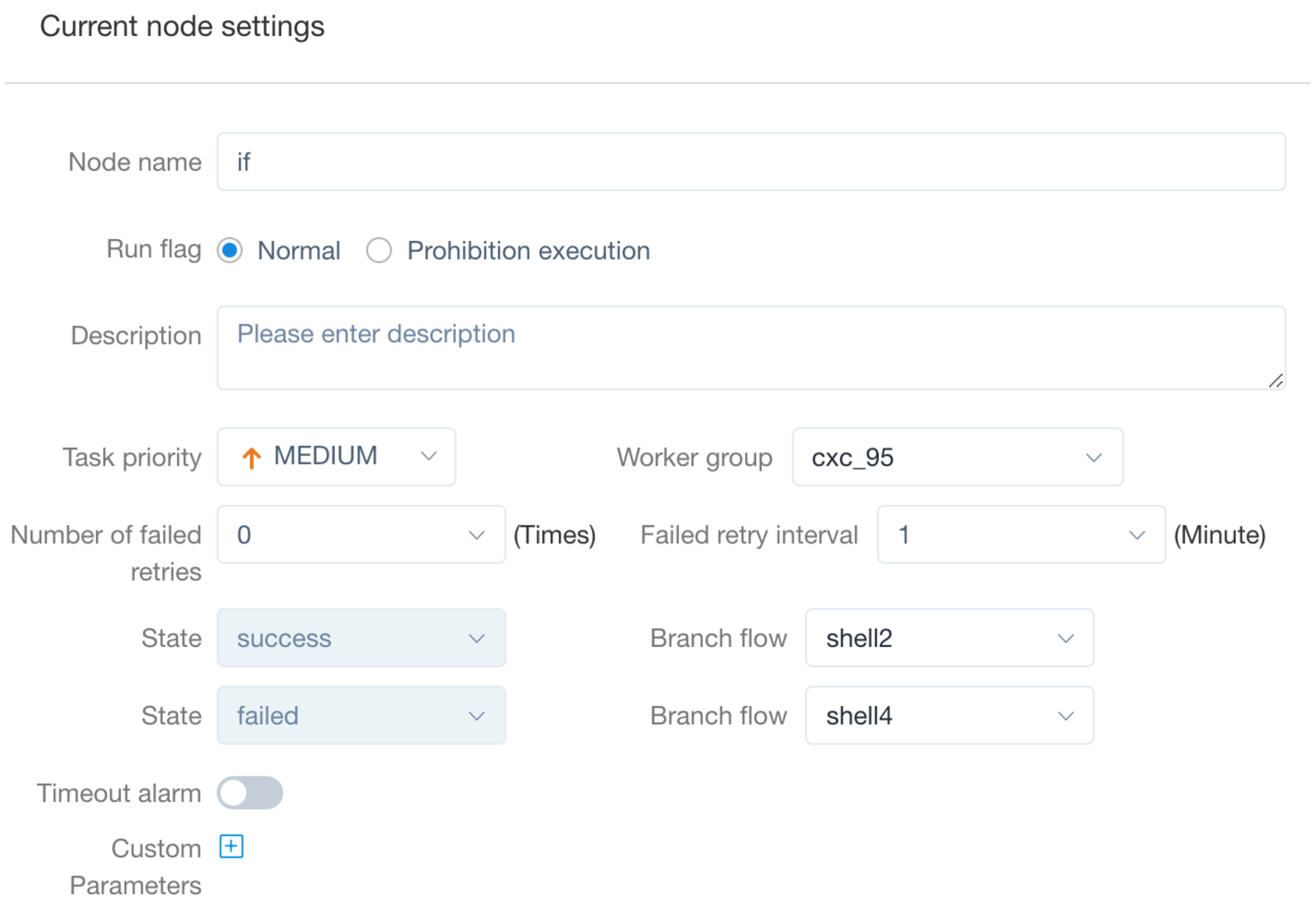
* Datasource	[✓ MYSQL] POSTGRESQL HIVE/IMPALA SPARK CLICKHOUSE ORACLE SQLSERVER DB2
* Datasource Name	Please enter datasource name
Description	Please enter description
* IP	Please enter IP
* Port	3306
* User Name	Please enter user name
Password	Please enter your password
* Database Name	Please enter database name
jdbc connect parameters	Please enter format {"key1":"value1","key2":"value2"...} connection parameter

Cancel | Test Connect | Submit

1. Visualized data-sources include： MySql、PostgerSql、Hive、Impala、Spark、ClickHouse、Oracle、SqlServer、DB2、MongoDB.
2. Supports Plugin data-source extension
3. Visualized data-source management
4. Configure once, use everywhere.

DolphinScheduler Condition Task



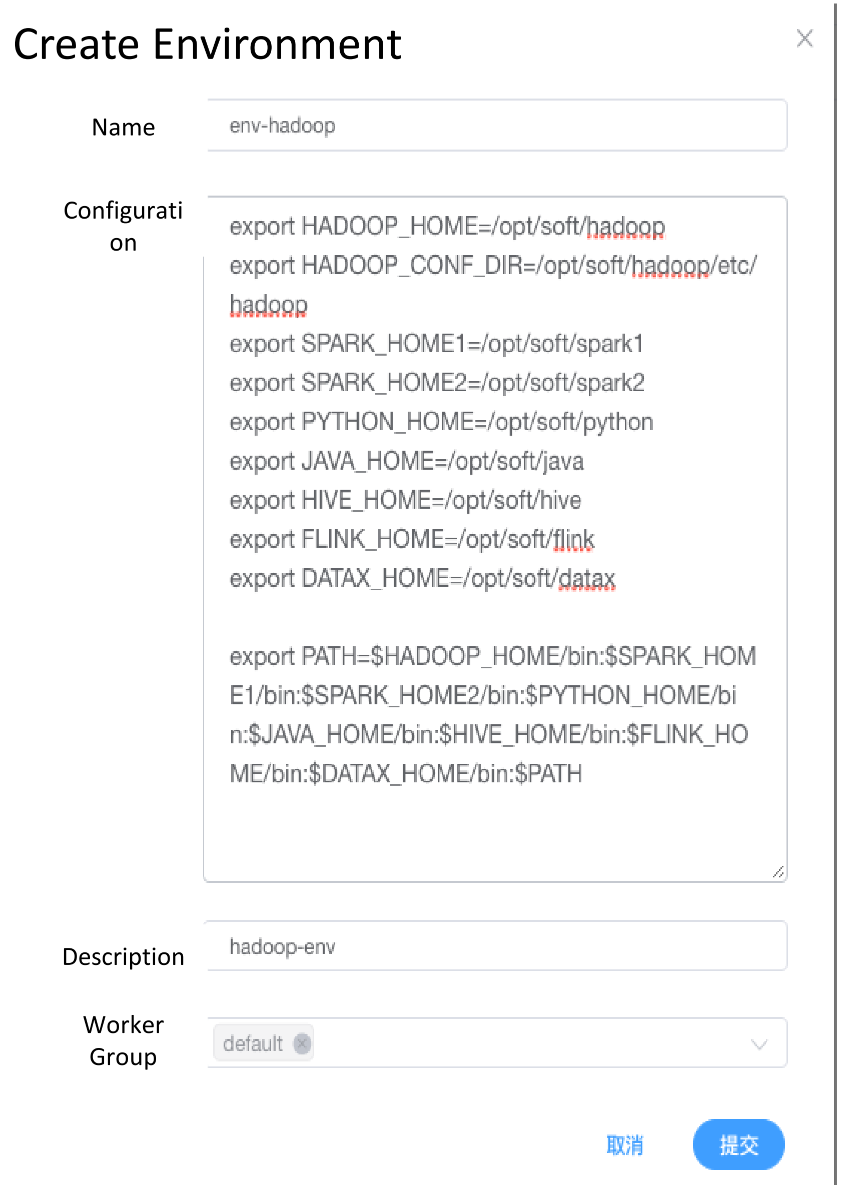


Current node settings

Node name	if
Run flag	[✓ Normal] Prohibition execution
Description	Please enter description
Task priority	MEDIUM
Worker group	cxc_95
Number of failed retries	0 (Times)
Failed retry interval	1 (Minute)
State	success
Branch flow	shell2
State	failed
Branch flow	shell4
Timeout alarm	
Custom Parameters	

Multi-Cloud, Multi-K8s, Muti Big Data Environment Support

- Different Environment Configuration
- Task mapping with Environment
- Different WorkerGroup Environment Configuration


Create Environment

Name: env-hadoop

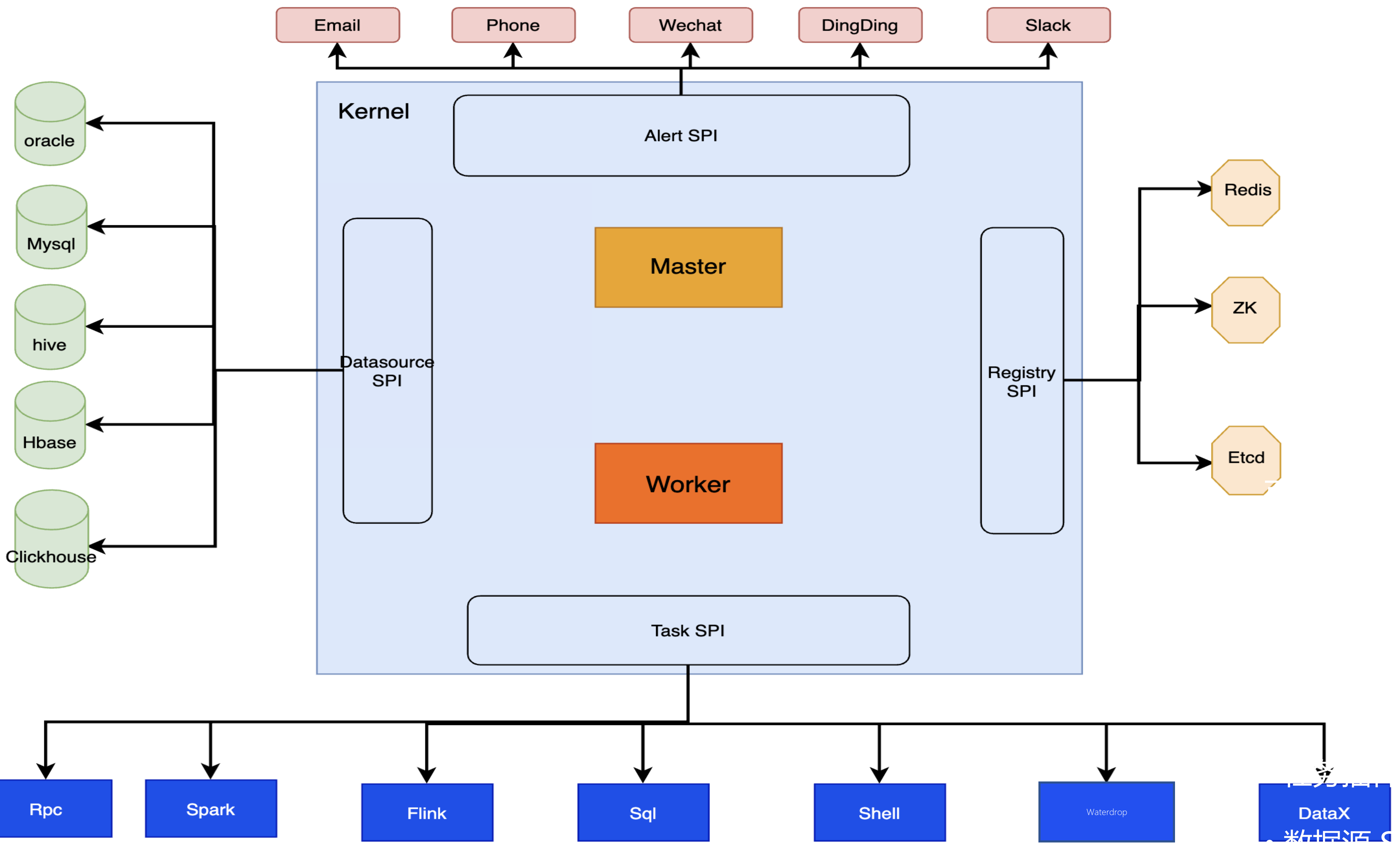
Configuration:

```
export HADOOP_HOME=/opt/soft/hadoop
export HADOOP_CONF_DIR=/opt/soft/hadoop/etc/hadoop
export SPARK_HOME1=/opt/soft/spark1
export SPARK_HOME2=/opt/soft/spark2
export PYTHON_HOME=/opt/soft/python
export JAVA_HOME=/opt/soft/java
export HIVE_HOME=/opt/soft/hive
export FLINK_HOME=/opt/soft/flink
export DATAX_HOME=/opt/soft/datax

export PATH=$HADOOP_HOME/bin:$SPARK_HOME1/bin:$SPARK_HOME2/bin:$PYTHON_HOME/bin:$JAVA_HOME/bin:$HIVE_HOME/bin:$FLINK_HOME/bin:$DATAX_HOME/bin:$PATH
```

Description: hadoop-env

Worker Group: default

Service Provide Interface – Easy to Extend Your Own Task & DataSource



DolphinScheduler 3.1.0 New Features

2.X version

Simple& WYSWYG workflow

- Drag & Drop to create workflow
- DAG Graph run-time management
- Open API to support others

High Reliability

- Decentralized multi-Masters and multi-Worker
- High performance(support 1m+ Task in production env)
- High Reliability

Rich Workflow Functions

- Support pause&resume workflow
- Support projects,multi-tenant
- Support 30+ Tasktype，Spark, Hive, MR, Python, Sub-Process, Shell,EMR, S3

Cloudnative&Extensible

- Support User-defined Task
- Condition and subworkflow
- Elastic Master & Worker dynamic on-line&off-line

+

3.1.0 New Feature

ML Orchestration

- DataPreparation+MLOps
- ML flow, Sagemaker,DVC
- Jupyter, PyTorch
- Kubeflow, TensorFlow,Bentoml…

Data Stream Support

- Flink, Sparking streaming Support
- Data Stream workflow Support
- Data Stream Management

Python, YAML Workflow Support

- Python generate Workflow
- YAML generate Workflow
- Code Review & Deployment

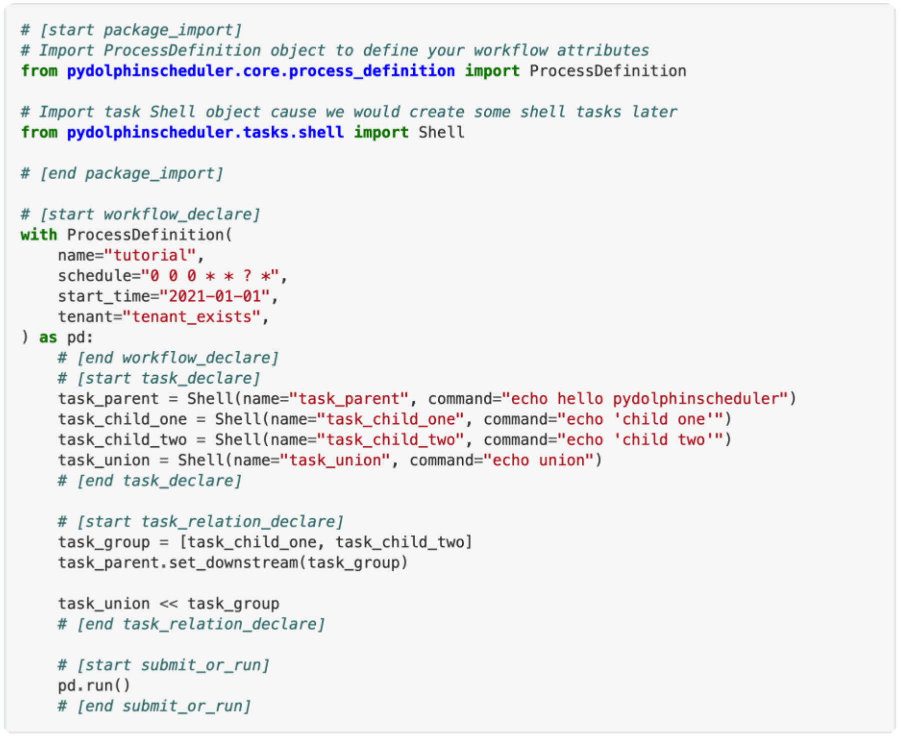
K8S Support

- K8S Operator
- K8S Task

New FeaturePyDolphinScheduler

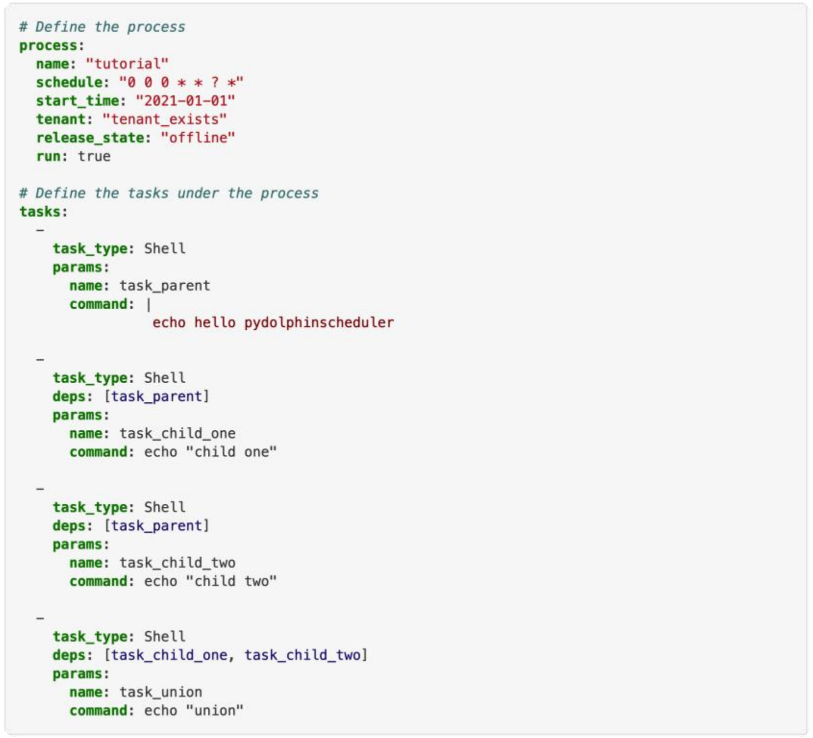
PyDolphinScheduler is Python API for Apache DolphinScheduler, which allow you definition your workflow by Python code, aka workflow-as-codes.

Python



```python
# [start package_import]
# Import ProcessDefinition object to define your workflow attributes
from pydolphinscheduler.core.process_definition import ProcessDefinition

# Import task Shell object cause we would create some shell tasks later
from pydolphinscheduler.tasks.shell import Shell

# [end package_import]

# [start workflow_declare]
with ProcessDefinition(
    name="tutorial",
    schedule="0 0 0 * * ? *",
    start_time="2021-01-01",
    tenant="tenant_exists",
) as pd:
    # [end workflow_declare]
    # [start task_declare]
    task_parent = Shell(name="task_parent", command="echo hello pydolphinscheduler")
    task_child_one = Shell(name="task_child_one", command="echo 'child one'")
    task_child_two = Shell(name="task_child_two", command="echo 'child two'")
    task_union = Shell(name="task_union", command="echo union")
    # [end task_declare]

    # [start task_relation_declare]
    task_group = [task_child_one, task_child_two]
    task_parent.set_downstream(task_group)

    task_union << task_group
    # [end task_relation_declare]

    # [start submit_or_run]
    pd.run()
    # [end submit_or_run]
```

YAML

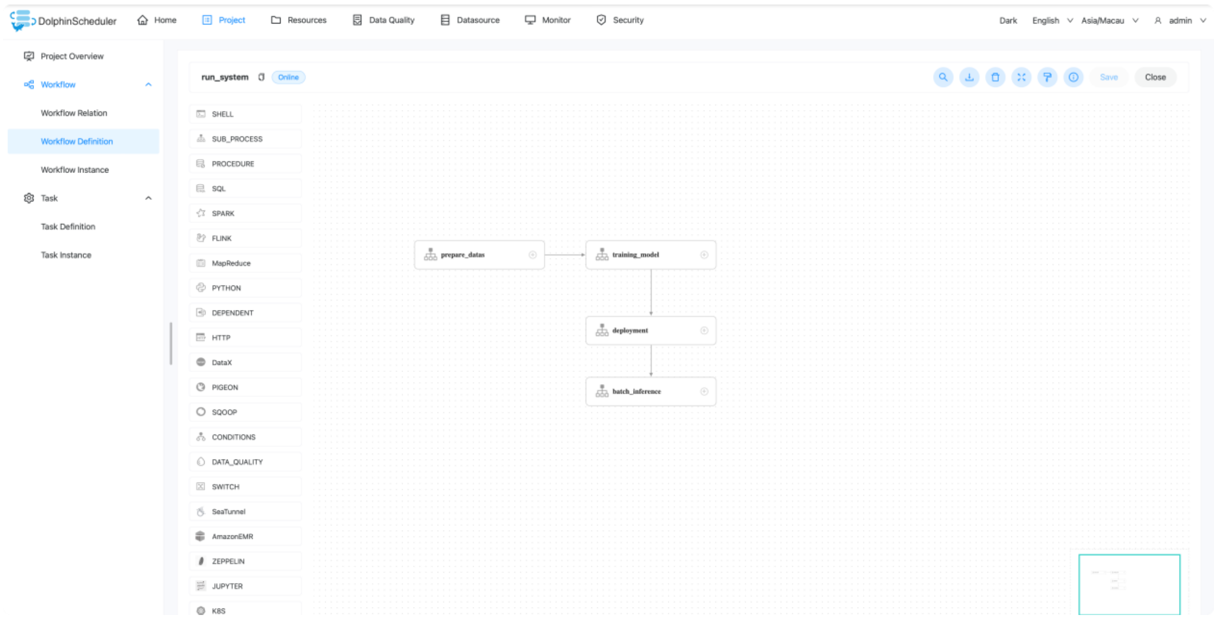


```yaml
# Define the process
process:
  name: "tutorial"
  schedule: "0 0 0 * * ? *"
  start_time: "2021-01-01"
  tenant: "tenant_exists"
  release_state: "offline"
  run: true

# Define the tasks under the process
tasks:
  -
    task_type: Shell
    params:
      name: task_parent
      command: |
                echo hello pydolphinscheduler

  -
    task_type: Shell
    deps: [task_parent]
    params:
      name: task_child_one
      command: echo "child one"

  -
    task_type: Shell
    deps: [task_parent]
    params:
      name: task_child_two
      command: echo "child two"

  -
    task_type: Shell
    deps: [task_child_one, task_child_two]
    params:
      name: task_union
      command: echo "union"
```

New Feature DolphinScheduler ML Orchestration x MLOps

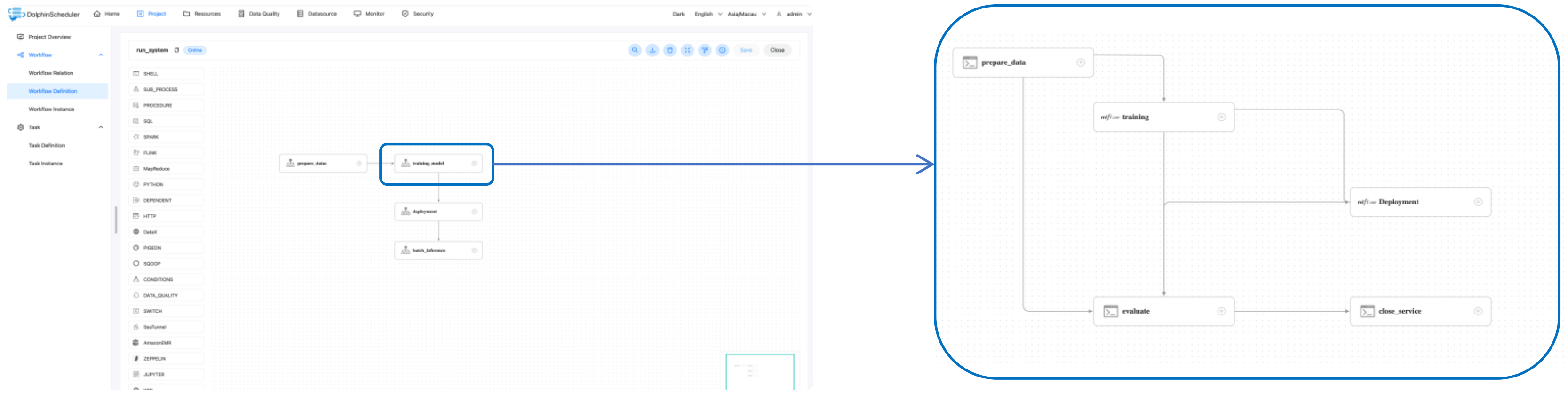
In the field of MLOps, DolphinScheduler is adding a variety of machine learning-related task plugin to help data analysts and data scientists easily use DolphinScheduler. Solve the following two problems:

The efficiency of the machine learning lifecycle
The efficiency of machine learning systems to connect with other systems

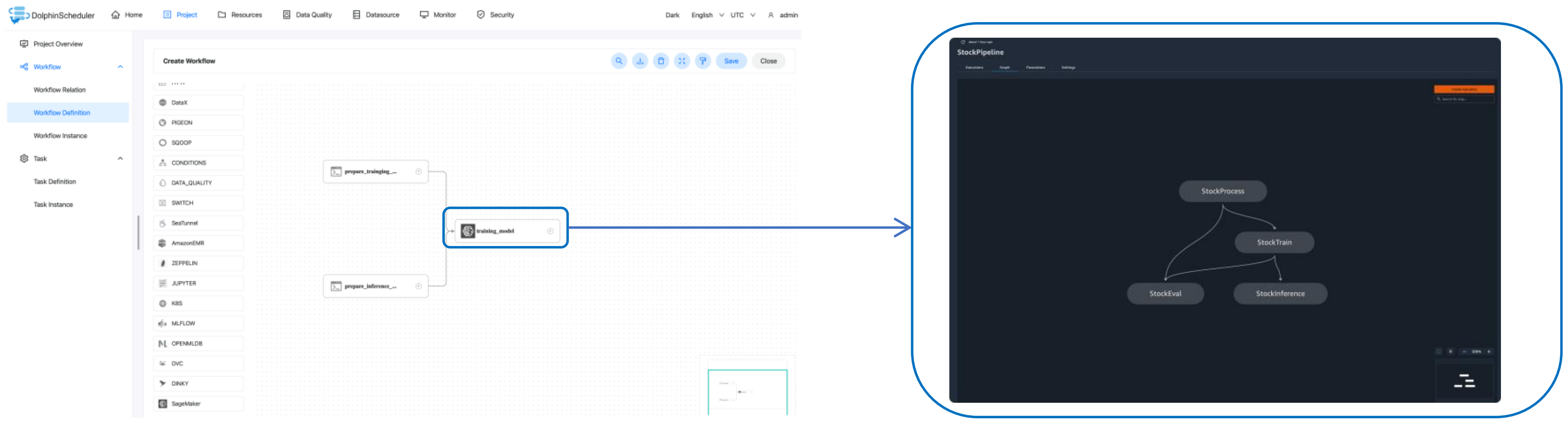


MLOps Orchestration

Machine learning workflows in DolphinScheduler



Machine learning workflows between Spark and SageMaker



MLOps Orchestration

Task plugins support for machine learning workflow

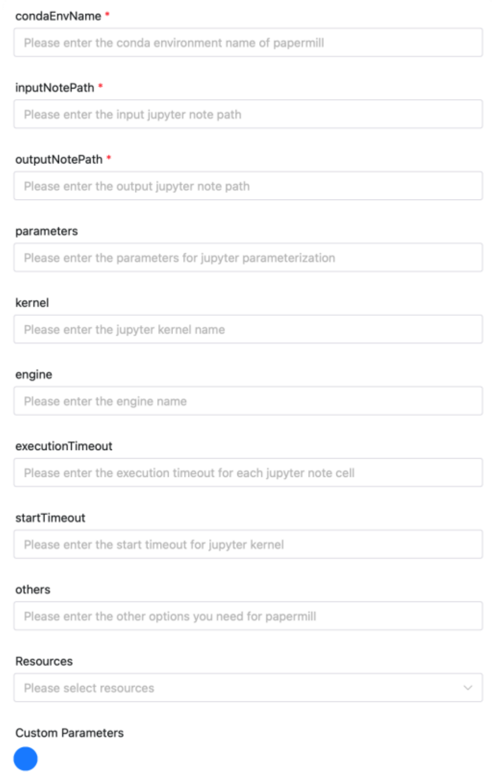
Data Management	Feature Store	Model Training	Deployment	Model Management
DVC, SageMaker	OpenMLDB, SageMaker	Shell, Python, Jupyter, MLflow, Pytorch, SageMaker	Shell, Python, MLflow, SageMaker	MLflow, SageMaker

Task plugins in DolphinScheduler MLOps Orchestration

Task Plugin	Scenario
Jupyter	Schedule the execution of model training, data analysis notebook Add the notebook to the workflow
MLflow	Run the custom MLFlow Project, built-in algorithms, AutoML Deploy machine learning models
OpenMLDB	Feature extraction and calculation for offline and online consistency
DVC	Upload and download data based on version information Large file version management based on Git repository
SageMaker	Schedule the execution of SageMaker Pipeline Connect tasks such as upstream big data analytics or some downstream tasks
Pytorch	Migrate the machine learning project to DolphinScheduler Run a Git-based machine learning project Also can run Tensorflow and other ML project

Jupyter Task Plugin

Jupyter task plugin can create a jupyter-type task and execute jupyter notes, it will use papermill to evaluate jupyter notebooks.



condaEnvName *
Please enter the conda environment name of papermill

inputNotePath *
Please enter the input jupyter note path

outputNotePath *
Please enter the output jupyter note path

parameters
Please enter the parameters for jupyter parameterization

kernel
Please enter the jupyter kernel name

engine
Please enter the engine name

executionTimeout
Please enter the execution timeout for each jupyter note cell

startTimeout
Please enter the start timeout for jupyter kernel

others
Please enter the other options you need for papermill

Resources
Please select resources

Custom Parameters

Scenario:

1. Schedule to execute machine learning notebooks such as model training

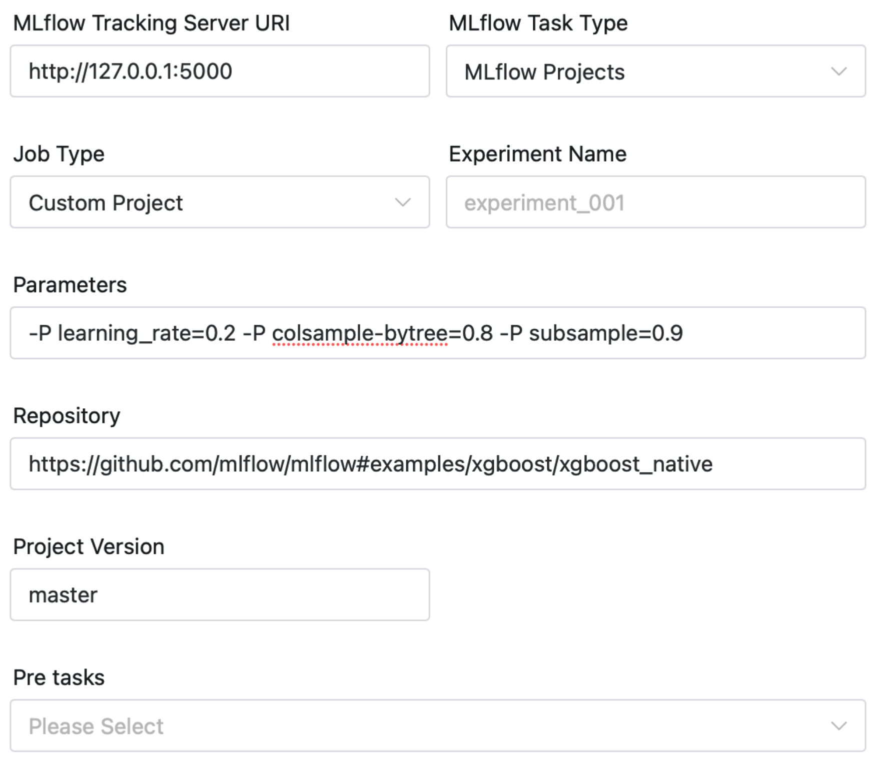
2. Schedule to execute data analysis and data visualization notebooks

3. Run the notebook with different parameters

Document links： https://dolphinscheduler.apache.org/en-us/docs/dev/user_doc/guide/task/jupyter.html

MLflow Task Plugin

MLflow task plugin used to execute MLflow tasks， Currently contains MLflow Projects and MLflow Models.



MLflow Tracking Server URI	MLflow Task Type
http://127.0.0.1:5000	MLflow Projects
Job Type	Experiment Name
Custom Project	experiment_001

Parameters

-P learning_rate=0.2 -P colsample-bytree=0.8 -P subsample=0.9

Repository

https://github.com/mlflow/mlflow#examples/xgboost/xgboost_native

Project Version

master

Pre tasks

Please Select

Scenario:

1. Run the preset algorithm

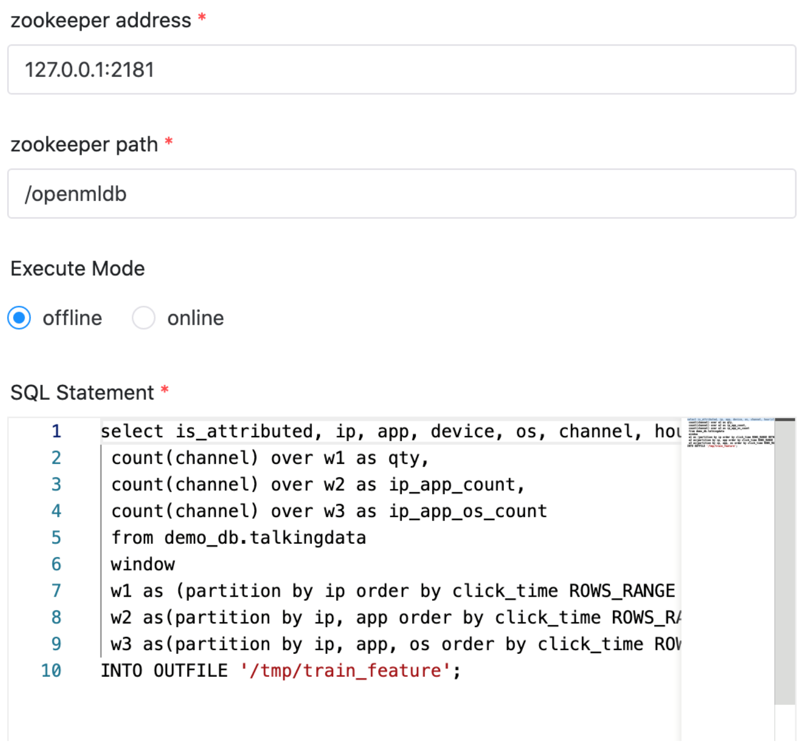
2. Run custom MLflow Project

3. Deploy the MLFlow model

Document links： https://dolphinscheduler.apache.org/en-us/docs/dev/user_doc/guide/task/mlflow.html

OpenMLDB Task Plugin

OpenMLDB task plugin used to execute tasks on OpenMLDB cluster, provide FeatureStore capability



zookeeper address *

127.0.0.1:2181

zookeeper path *

/openmldb

Execute Mode

[✓ offline] online

SQL Statement *

```
select is_attributed, ip, app, device, os, channel, hou
 count(channel) over w1 as qty,
 count(channel) over w2 as ip_app_count,
 count(channel) over w3 as ip_app_os_count
 from demo_db.talkingdata
 window
 w1 as (partition by ip order by click_time ROWS_RANGE
 w2 as(partition by ip, app order by click_time ROWS_RA
 w3 as(partition by ip, app, os order by click_time ROW
INTO OUTFILE '/tmp/train_feature';
```

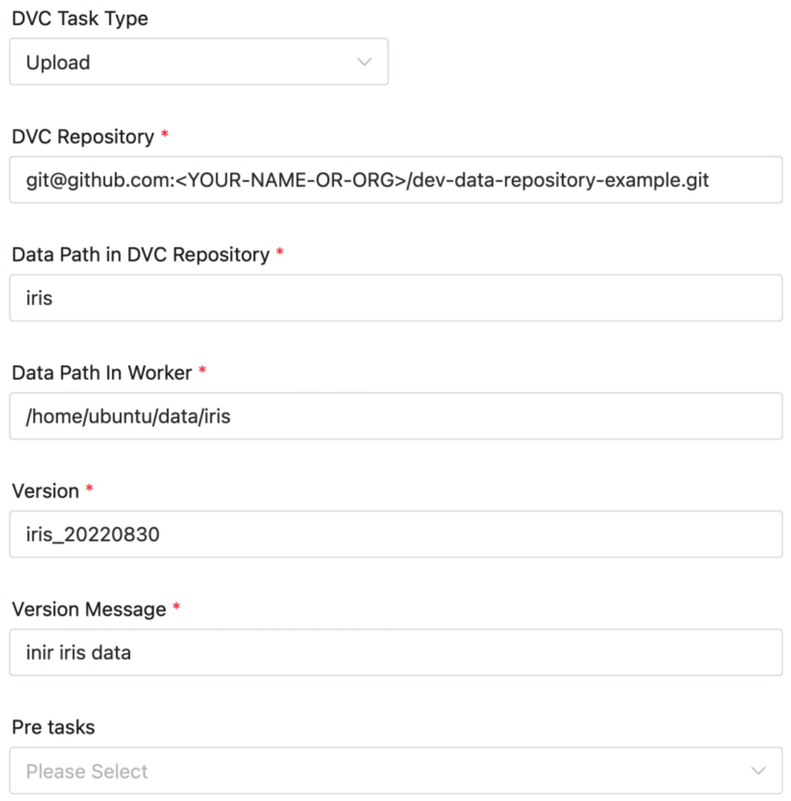
Scenario:

1. Offline feature extraction
2. Online feature extraction
3. Online and offline consistency

Document links： https://dolphinscheduler.apache.org/en-us/docs/dev/user_doc/guide/task/openmldb.html

DVC Task Plugin

DVC task plugin is used to use the data version management function of DVC on DolphinScheduler, helping users to carry out data version management easily.



DVC Task Type	Upload
DVC Repository *	git@github.com:<YOUR-NAME-OR-ORG>/dev-data-repository-example.git
Data Path in DVC Repository *	iris
Data Path In Worker *	/home/ubuntu/data/iris
Version *	iris_20220830
Version Message *	inir iris data
Pre tasks	Please Select

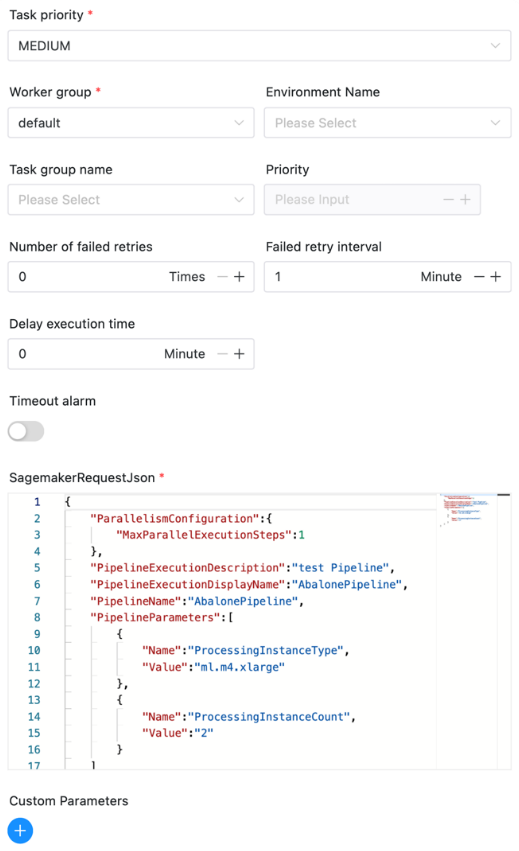
Scenario:

1. Upload the data and record the version
2. Download version-specific data
3. Large file version management based on Git repository

Document links： https://dolphinscheduler.apache.org/en-us/docs/dev/user_doc/guide/task/dvc.html

Amazon SageMaker Task Plugin

SageMaker task plugin can start a SageMaker pipeline execution and use DolphinScheduler to connect other upstream and downstream tasks.



Task priority *
MEDIUM

Worker group *
default

Environment Name
Please Select

Task group name
Please Select

Priority
Please Input

Number of failed retries
0 Times

Failed retry interval
1 Minute

Delay execution time
0 Minute

Timeout alarm

SagemakerRequestJson *

```
{
    "ParallelismConfiguration":{
        "MaxParallelExecutionSteps":1
    },
    "PipelineExecutionDescription":"test Pipeline",
    "PipelineExecutionDisplayName":"AbalonePipeline",
    "PipelineName":"AbalonePipeline",
    "PipelineParameters":[
        {
            "Name":"ProcessingInstanceType",
            "Value":"ml.m4.xlarge"
        },
        {
            "Name":"ProcessingInstanceCount",
            "Value":"2"
        }
    ]
```

Custom Parameters

Scenario:

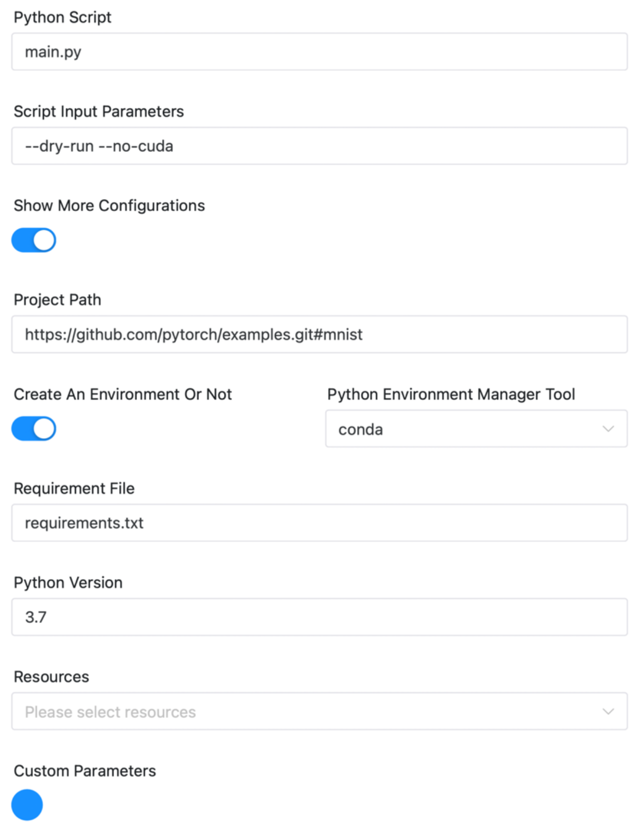
1. Schedule the execution of SageMaker Pipeline

2. Connect tasks such as upstream big data analytics or some downstream tasks

Document links： https://dolphinscheduler.apache.org/en-us/docs/dev/user_doc/guide/task/sagemaker.html

Pytorch Task Plugin

Pytorch task plugin enables users to run Pytorch projects in DolphinScheduler more conveniently. In addition, it supports handy Python environment management.



Python Script	main.py
Script Input Parameters	--dry-run --no-cuda
Show More Configurations	✓
Project Path	https://github.com/pytorch/examples.git#mnist
Create An Environment Or Not	✓
Python Environment Manager Tool	conda
Requirement File	requirements.txt
Python Version	3.7
Resources	Please select resources
Custom Parameters	●

Scenario:

1. Migrate the machine learning project to DolphinScheduler
2. Run a Git-based machine learning project
3. Also can run Tensorflow and other ML project

Document links： https://dolphinscheduler.apache.org/en-us/docs/dev/user_doc/guide/task/pytorch.html

MLOps Orchestration

Current and future supported machine learning projects.

The current support

Amazon SageMaker

mlflow

OpenMLDB

DVC

jupyter

PyTorch

The future support

TensorFlow

BENTOML

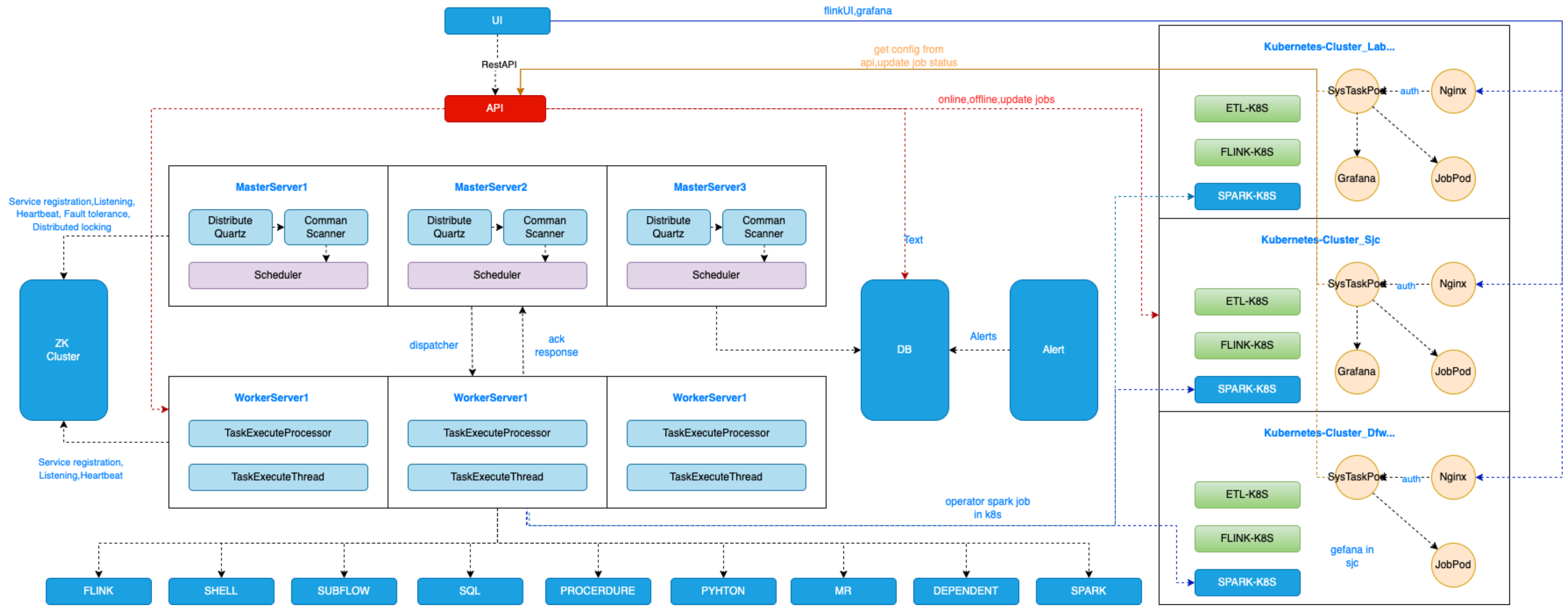
Kubeflow

CORE

Agenda

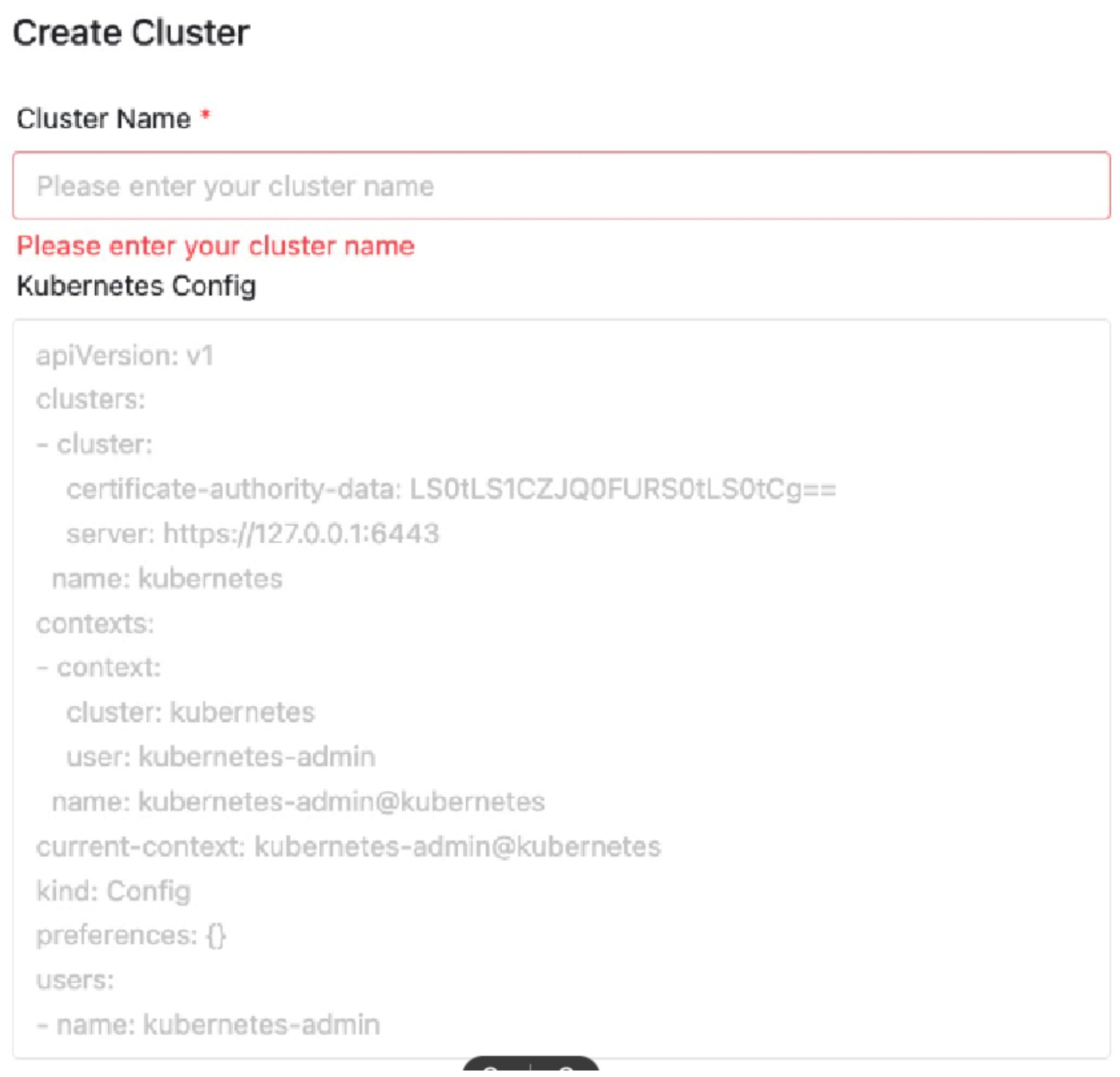
- Introduction of DolphinScheduler
- 2.0 & 3.1.0 New Features
- User Case – Cisco Webx

DolphinScheduler with Kubernetes Integration



Kubernetes Multi-Cluster Management

- Import Kubernetes cluster by input cluster config on the portal
- Support both self-built Kubernetes cluster and public cloud managed Kubernetes cluster, ex Elastic Kubernetes Service



Create Cluster

Cluster Name *

Please enter your cluster name

Please enter your cluster name

Kubernetes Config

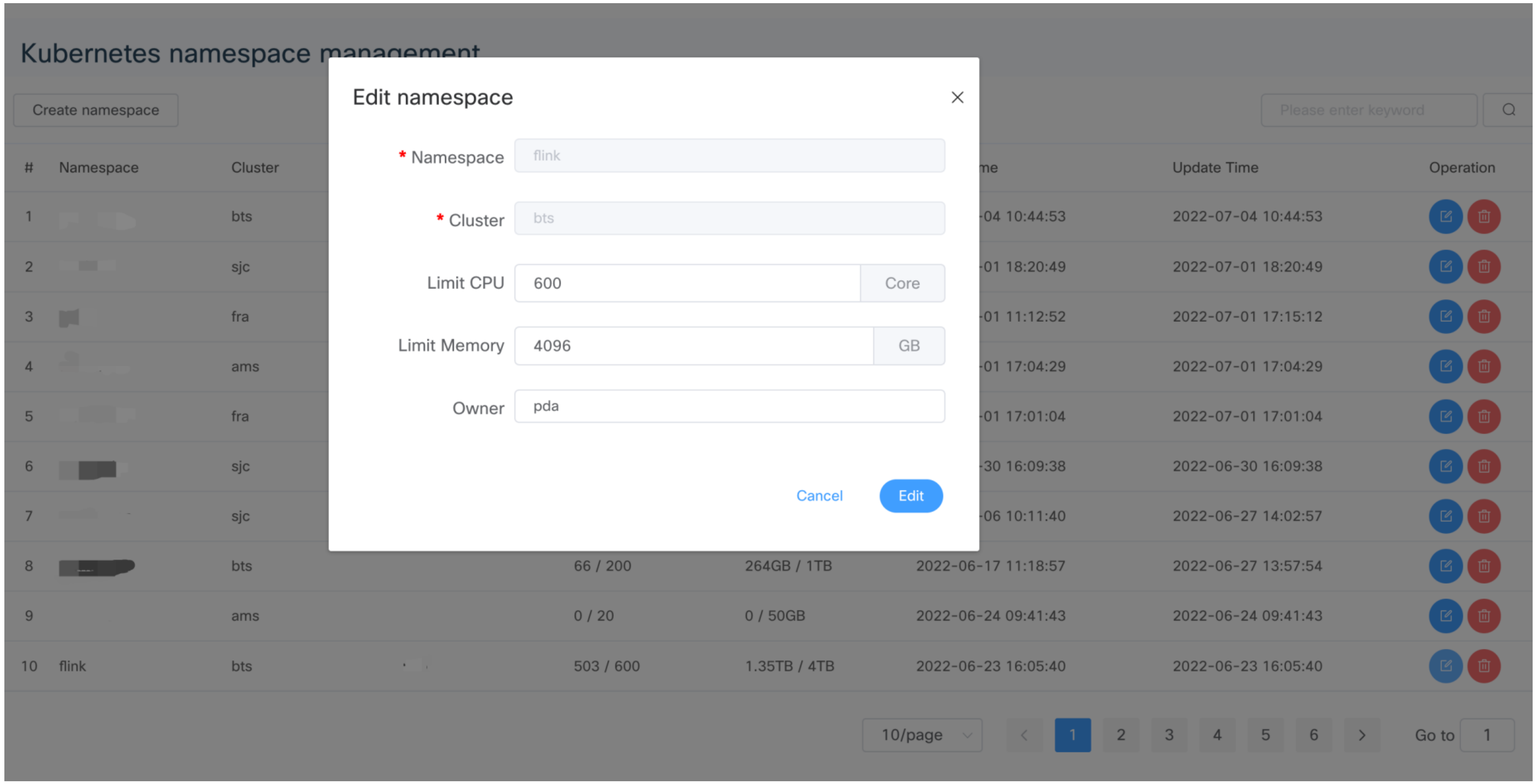
```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CZJQ0FURS0tLS0tCg==
    server: https://127.0.0.1:6443
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
kind: Config
preferences: {}
users:
- name: kubernetes-admin
```

APACHECON

Kubernetes Multi-Cluster Namespace Management

- Set CPU/Memory Limit for each namespace on different Kubernetes clusters

- Separate resource pools for multi tenancy



Kubernetes namespace management

Create namespace

Please enter keyword

Edit namespace

* Namespace	flink
* Cluster	bts
Limit CPU	600 Core
Limit Memory	4096 GB
Owner	pda

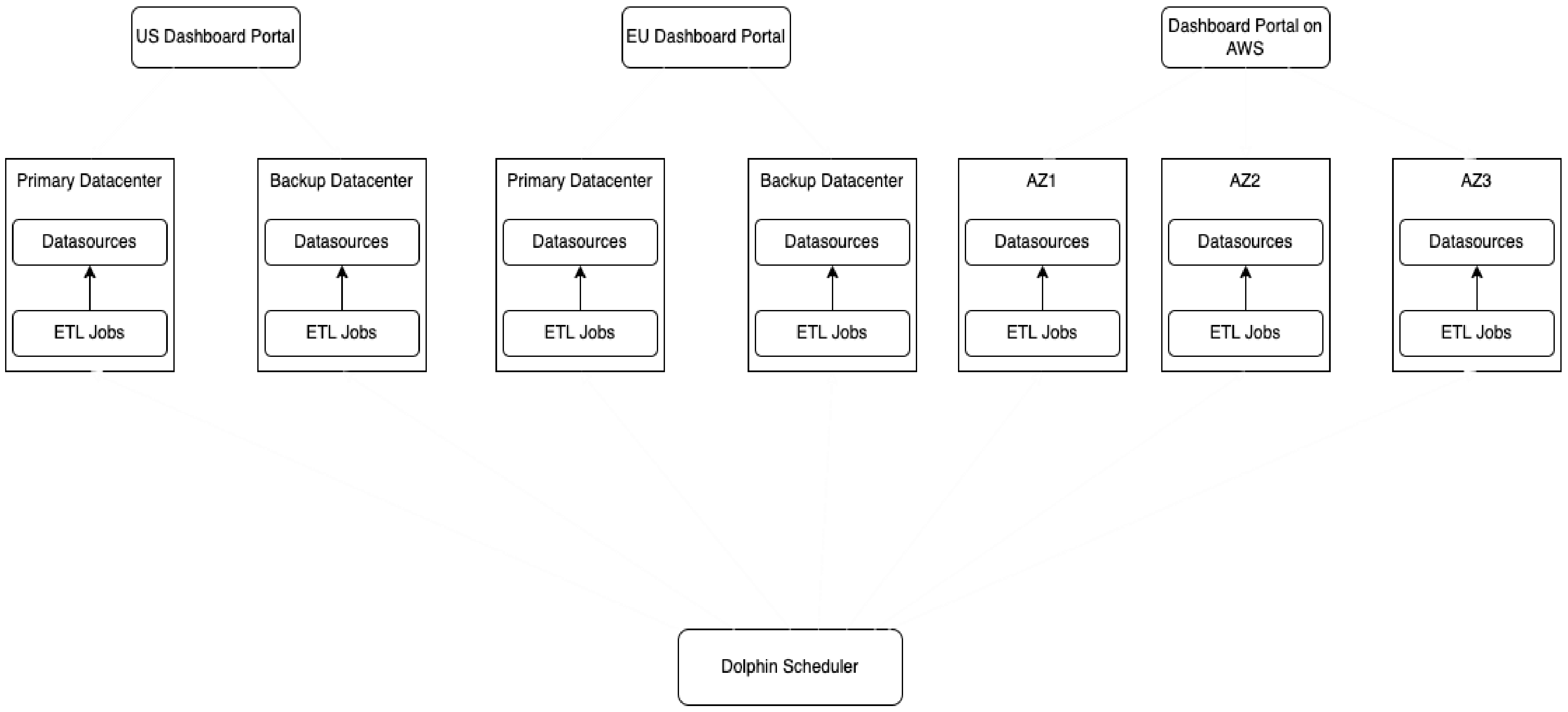
Cancel Edit

#	Namespace	Cluster				me	Update Time	Operation
1	[redacted]	bts				-04 10:44:53	2022-07-04 10:44:53	
2	[redacted]	sjc				-01 18:20:49	2022-07-01 18:20:49	
3	[redacted]	fra				-01 11:12:52	2022-07-01 17:15:12	
4	[redacted]	ams				-01 17:04:29	2022-07-01 17:04:29	
5	[redacted]	fra				-01 17:01:04	2022-07-01 17:01:04	
6	[redacted]	sjc				-30 16:09:38	2022-06-30 16:09:38	
7	[redacted]	sjc				-06 10:11:40	2022-06-27 14:02:57	
8	[redacted]	bts		66 / 200	264GB / 1TB	2022-06-17 11:18:57	2022-06-27 13:57:54	
9		ams		0 / 20	0 / 50GB	2022-06-24 09:41:43	2022-06-24 09:41:43	
10	flink	bts	[redacted]	503 / 600	1.35TB / 4TB	2022-06-23 16:05:40	2022-06-23 16:05:40	

10/page < 1 2 3 4 5 6 > Go to 1

Multi-Cluster ETL Job Management

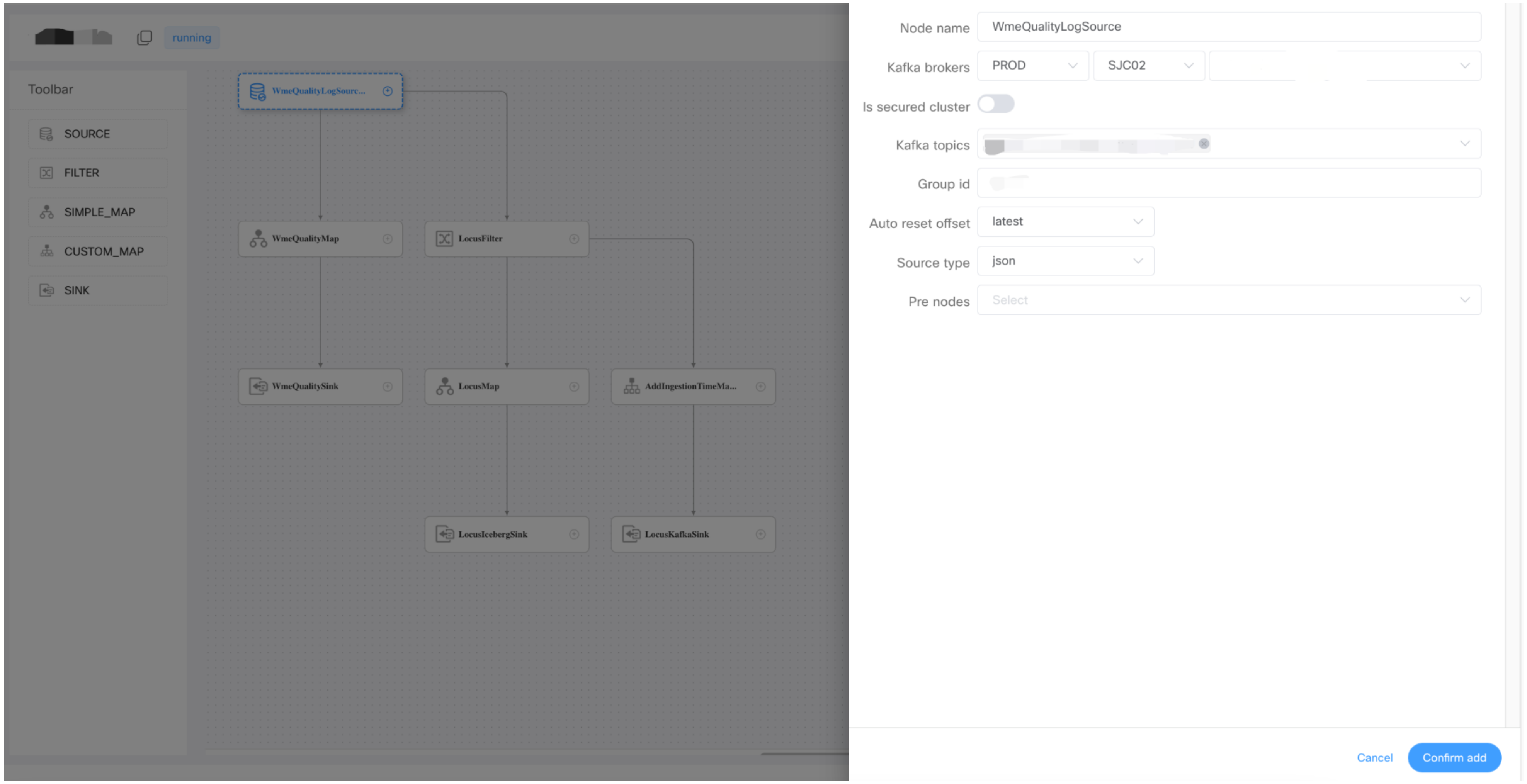
- Centralized job scheduling for multiple datacenters across the world
- Both private datacenter and public cloud support



APACHECON

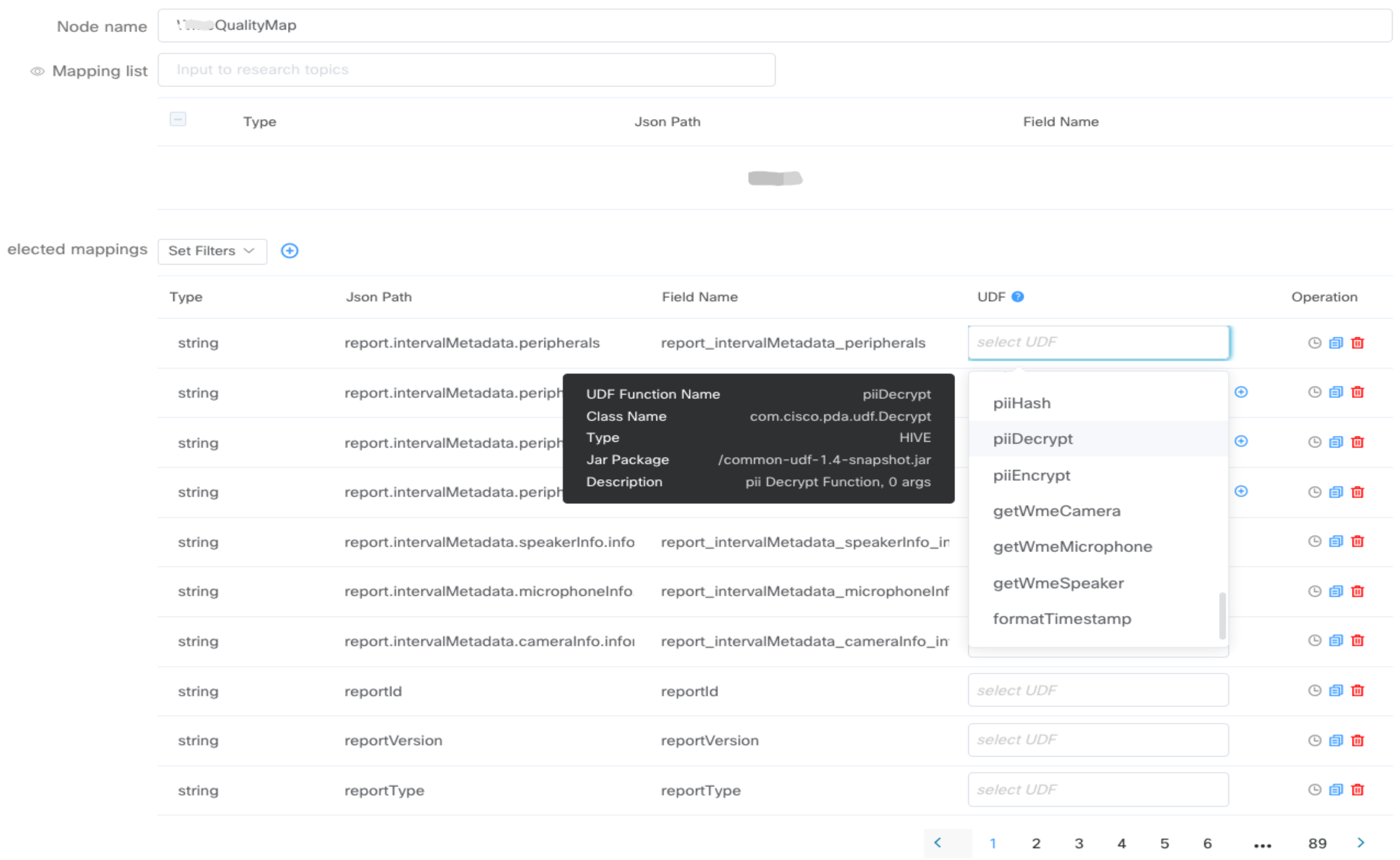
Simple ETL pipeline

- Generate a complex pipeline by Drag and Drop - No coding required
- Automatic scaling
- Stateless
- UDF support
- Job version management



Node name	WmeQualityLogSource
Kafka brokers	PROD / SJC02
Is secured cluster	
Kafka topics	[redacted]
Group id	[redacted]
Auto reset offset	latest
Source type	json
Pre nodes	Select

Simple ETL pipeline – UDF Management



Node name: [redacted]QualityMap

Mapping list: Input to research topics

Type	Json Path	Field Name

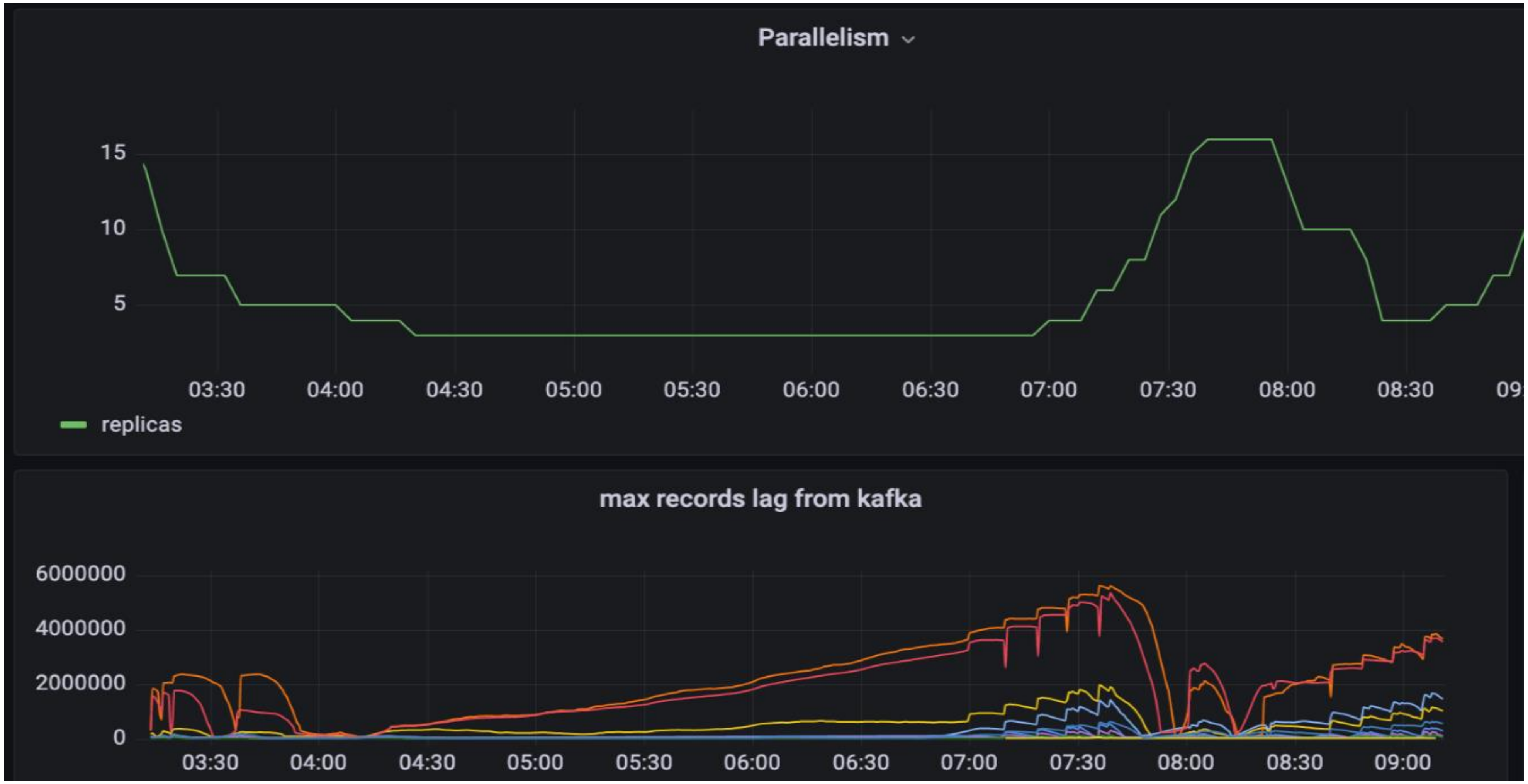
elected mappings · Set Filters

Type	Json Path	Field Name	UDF	Operation
string	report.intervalMetadata.peripherals	report_intervalMetadata_peripherals	select UDF	
string	report.intervalMetadata.periph			
string	report.intervalMetadata.periph			
string	report.intervalMetadata.periph			
string	report.intervalMetadata.speakerInfo.info	report_intervalMetadata_speakerInfo_ir		
string	report.intervalMetadata.microphoneInfo	report_intervalMetadata_microphoneInf		
string	report.intervalMetadata.cameraInfo.infoı	report_intervalMetadata_cameraInfo_in		
string	reportId	reportId	select UDF	
string	reportVersion	reportVersion	select UDF	
string	reportType	reportType	select UDF	

UDF Function Name	piiDecrypt
Class Name	com.cisco.pda.udf.Decrypt
Type	HIVE
Jar Package	/common-udf-1.4-snapshot.jar
Description	pii Decrypt Function, 0 args

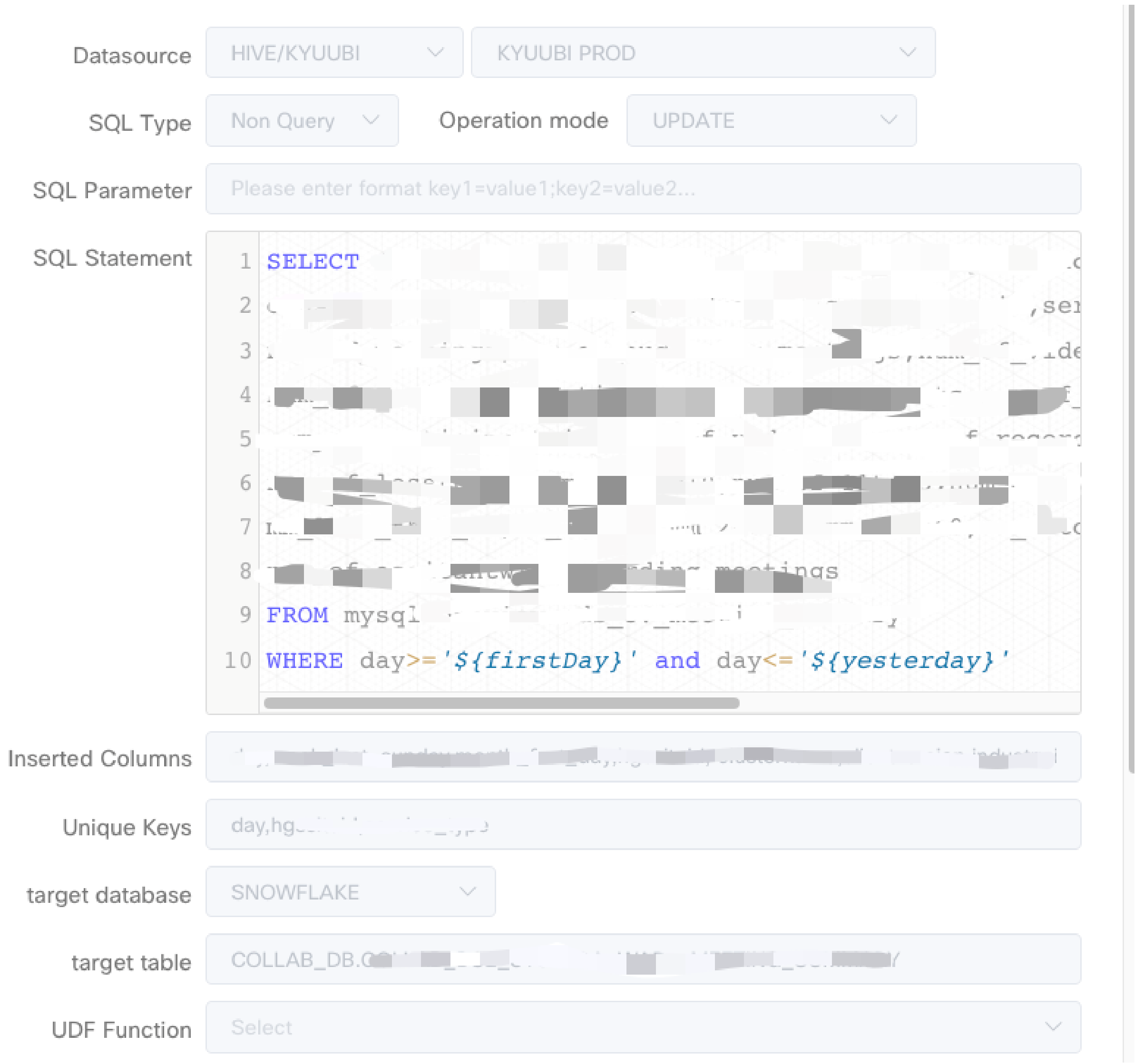
APACHECON

Simple ETL pipeline – Automatic Scaling



SQL Task Customization

- Snowflake Support in SQL Task
- Upsert feature for Snowflake Spark connector
- Sink selection



Datasource: HIVE/KYUUBI | KYUUBI PROD

SQL Type: Non Query | Operation mode: UPDATE

SQL Parameter: Please enter format key1=value1;key2=value2...

SQL Statement:

```sql
SELECT [redacted]
[redacted]
FROM mysql [redacted]
WHERE day>='${firstDay}' and day<='${yesterday}'
```

Inserted Columns: [redacted]

Unique Keys: day,hg [redacted]

target database: SNOWFLAKE

target table: COLLAB_DB.C [redacted]

UDF Function: Select

Resource

website: https://dolphinscheduler.apache.org

GitHub： https://github.com/apache/dolphinscheduler

E-mail： dev@DolphinScheduler.apache.org

Slack: https://s.apache.org/dolphinscheduler-slack

Twitter : @dolphinschedule

Demo： http://106.75.43.194:8888/

Apache DolphinScheduler

Smart, Easy and Stable Data Job Orchestration Tools

Q&A

https://www.linkedin.com/in/williamk2000/ or search William Kwok

E-mail: guowei@apache.org

Twitter: guowei_William