

# Design and Implementation of Workflow Scheduling Platform for Big Data

Zhifei Tan<sup>1,2,\*</sup>, Chen Li<sup>1,2</sup>, Xia Hou<sup>2</sup>, Junlin Du<sup>1</sup> and Haibo Wang<sup>1</sup>

<sup>1</sup>Institute of Software Chinese Academy of Sciences, Science and Technology on Integrated Information System Laboratory  
Beijing, China

<sup>2</sup>Beijing Information and Science Technology University Beijing, China

\*Corresponding author

**Abstract**—Data analysts use big data processing technology to analyze rich content and find statistical law. The demand for analyzing data is soaring. This paper designs a visualization big data workflow scheduling platform. It can simplify processes of analyzing data via using user interfaces to edit workflow. It integrates data processing algorithm for Hadoop and Spark, also provides various data analysis services and management interfaces for workflow programs and files.

**Keywords**-big data; workflow scheduling platform;hadoop

## I. INTRODUCTION

With the rapid development of the Internet, data analysts need to use big data processing technology for analyzing data. But learning process of Hadoop is extremely complex and its learning cycle is very long, so that it is difficult to be used. On the other hand, that the manual input command analyzes data often leads to task failure or error due to various subjective factors. Although there are many products based on the workflow engine for Hadoop, these products are usually operated by command lines. It makes these products very complex to be used and deployed. Both of them lack the visualized operating environment. Furthermore, the ability of managing big data tasks is so far from perfect that it causes reducing efficiency when confronting various processes of computing and processing programs. To solve these problems, there are several workflow engines provided, such as Oozie, Azkaban, Zeus etc. Jie Chen and others focus on the process and effect of big data processing technology platform for gaming [2] that adopts the open-source workflow engine system, named Zeus, and improve efficiency of managing and scheduling the workflow task greatly. Similarly, Wenjun Zhang designed the visualized data processing platform [3] by using Oozie to improve convenience and reduce the error rate for data analysts, but it is only aimed at features of the MapReduce task. And there are also other open-source platform for big data processing technology, such as Hue. The Hue is an Apache Hadoop UI system which Cloudera contributes to open-source community. We can use the Hue to interact with Hadoop clusters and process data through user interfaces, such as the operation on the HDFS, running MapReduce Job etc. But user

interfaces of the Hue are so complex and the user experience is so terrible that resulting in very few users. Therefore, we summed up the state of these products, then design and implement a workflow scheduling platform which is more convenient and practical. The system is a second exploitation to the workflow engine of Oozie. Contents of this paper include several aspects as follows:

1) Implementation of a B/S architecture Web service system, the system features include:

- a). Editing workflow visually;
- b). Managing components based on user type;
- c). Monitoring status based on the workflow node;
- d). Supporting to upload custom components.

2) Putting forward design ideas of each system module;

3) Testing and analyzing the result.

This paper is organized as follows. The first section introduces the overall architecture of the system. The second section discusses the design and implementation of the system, including the database design and the function of each module. The third section shows the system testing case and process. The fourth section gives a conclusion.

## II. OVERALL ARCHITECTURE

This system is based on J2EE technology architecture to develop a workflow scheduling platform. The system adopts the workflow engine of Oozie and B/S mode. Its system architecture is shown in Figure 1.

By the user interface, users can drag and drop the visual component to draw a directed acyclic graph (DAG). For the computing and processing program, users are supposed to package the source code into JAR file and upload to the component library, then it can be used in every workflow. After users submit the task, JAR files required in the workflow will be copied from the component library to the workspace of the current task. The task execution flow is shown in Figure 2.

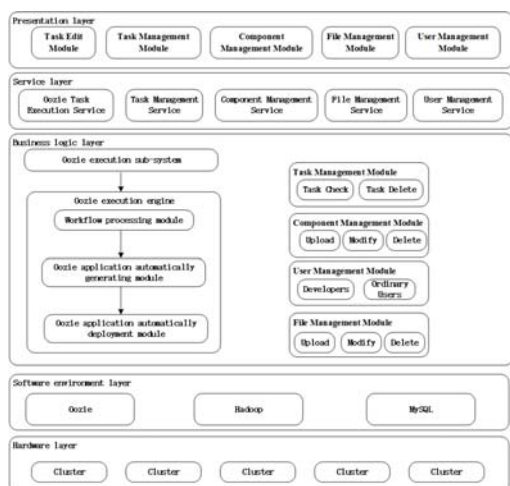


FIGURE I. SYSTEM ARCHITECTURE DIAGRAM.

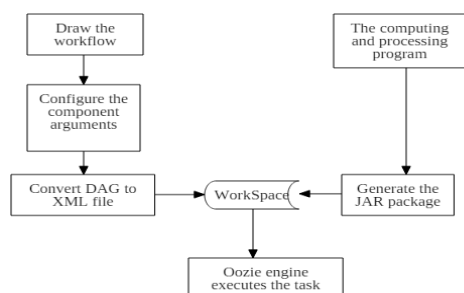


FIGURE II. TASK EXECUTION FLOW.

### III. DESIGN AND IMPLEMENTATION OF THE SYSTEM

The system consists of five modules: user management, task edit, task management, component management and file management. Each module has its own function. The use case diagram of the system is shown in Figure 3.



FIGURE III. USE CASE DIAGRAM OF THE SYSTEM

#### A. User Management

The System is aimed at two types of user: developers and ordinary users. Developers can use the component management module to manage JAR files and design a specific workflow processing model for ordinary users. For ordinary users who are not familiar with workflow, they can directly use the workflow processing model that developers have designed and input required arguments to run the workflow. There are few professional operations. Besides, users can modify the password and personal information by this module. The use case diagram of the user management module is shown in Figure 4.

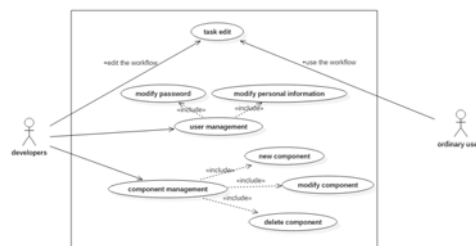


FIGURE IV. USE CASE DIAGRAM OF USER MANAGEMENT MODULE.

#### B. Component Management

The component management module contains three sub-functions: upload, delete and modify. It's for developers. Developers package the computing and processing program into the JAR file and then upload the JAR file to the component library on HDFS platform. Properties of the component are stored in MySQL database and the detail is shown in Table 1. Components are divided into two types: private and public. Private components are only used by the submitter and can't be the public component until the administrator agrees. Public components can be used by all users, but ordinary users don't have the permission to upload, delete, and modify.

TABLE I. PROPERTIES OF COMPONENT

name	data type	constraint	description
id	INT	PRIMARY KEY	The component id
name	VARCHAR	NOT NULL, UNIQUE	The component name
type	VARCHAR		The component type
arguments	VARCHAR		The set of Arguments
description	VARCHAR		The component description
hdfspath	VARCHAR		the path of the component on HDFS

#### C. Task Edit

The task edit interface is mainly responsible for drawing the workflow graph. There is a menu of nodes on the left. Nodes what can be dragged and dropped by users are divided into two parts: the control node library and the action node library. The control node library provides nodes to control execution process of workflow, including the node of start, end, decision, fork, join etc. The action node library is exactly

the component the user uploaded. It contains the node of MapReduce, Java, Pig, Spark and so on. Users can drag and drop the node into the editing area to draw the workflow graph in accordance with the definition of DAG. DAG of the workflow is shown in Figure 9. Then users can set arguments for these action nodes. After the task is submitted, there are following steps to be done by the system. Initially, information of DAG and arguments will be converted into a JSON string. Furthermore, with the help of AJAX, the JSON string will be asynchronously sent to the background server. And the analysis algorithm is converting the JSON string to a XML configuration file with a special language, named hPDL, which can be understood by the Oozie engine. Then, the task is going to be executed by the Oozie engine. Finally, the feedback mechanism of Oozie will return the result to the user interface. It forms a management mode of "drawing → definition → submit → monitor". The task execution flow is shown in Figure 2, the task data flow diagram is shown in Figure 5.

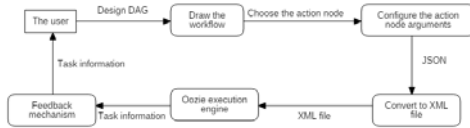


FIGURE V. TASK DATA FLOW DIAGRAM.

The tool of drawing the workflow graph is JsPlumb which is used in the interface. It is a powerful JavaScript connection library that can connect elements of HTML with the arrow, the curve, the line etc. As mentioned before, the process of converting DAG into a XML file is executed by the analysis algorithm in the background. With the help of Jsplumb, the source point and the target point of each connection can be recorded. Content of each connection will be saved as a JSON string and sent to the background. After analyzing, the following information can be obtained as in (1):

$$N_i(E_i, V_{i-1}, V_{i-2}) \quad (1)$$

$i$  means the  $i$ -th,  $E_i$  means the  $i$ -th edge,  $V_{i-1}$  means the source

---

Algorithm 1 Generating XML file for Oozie engine.

---

```

Iteration:
for all  $N_i \in N$  do
    if  $N_i.V_{i-1} = \text{end}$  then
        insert the config code of end to XML file;
    else
        switch( $N_i.V_{i-1}$ ) then
            case start :
                insert the config code of start to XML file;
                break;
            case decision :
                insert the config code of decision to XML file;
                break;
            case fork :
                insert the config code of fork to XML file;
                break;
            case join :
                insert the config code of join to XML file;
                break;
            case action node type :
                insert the config code of action node to XML file;
                break;
        endswitch
    endif
endfor

```

---

FIGURE VI. ALGORITHM OF GENERATING XML FILE.

vertex of the edge,  $V_{i-1}$  means the target vertex of the edge and total edges and vertices is represented by a set as in (2):

$$N = \{N_1, N_2, N_3, \dots, N_i\} \quad (2)$$

Implementation of the analysis algorithm is by iterating the set  $N$  to determine what kind of content should be written into XML file by IO stream and according to hPDL language standard. Then it is going to generate a complete XML file that the workflow requires. Algorithm pseudocode is shown in Figure 6 as follows.

#### D. Task Management

The task management module provides ability to monitor tasks and query real-time status for all tasks. The task status contains PREPARE, RUNNING, KILLED and SUCCEEDED. Besides, it also contains the task name, start time, duration and so on. It is more intuitive to allow the user to know details of the single task or multiple tasks. The database will store related information for every workflow. The E-R diagram of the task management module is shown in Figure 7. The task management module contains three sub-functions: check, edit and delete.

- 1) Check: After running the task, users can use this function to check details of the current task and the real-time feedback mechanism can locate which node is running. If the exception or error occurs, users can click the node to view the cause of the exception or error.
- 2) Re-edit: Users can re-edit finished or not tasks. Ordinary users use this function to re-edit the workflow graph which developers have drawn. After setting arguments, the user can run the workflow.
- 3) Delete: Deleting the workflow task can facilitate to manage the task list for users.

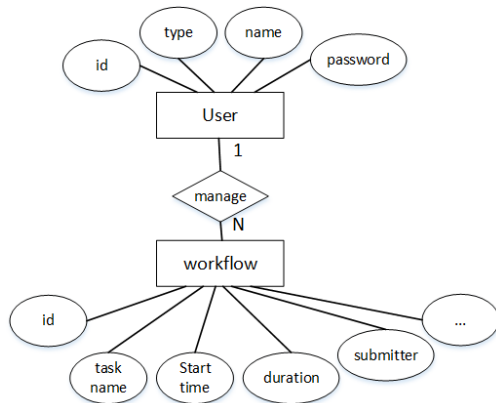


FIGURE VII. TASK MANAGEMENT MODULE E-R DIAGRAM.

### E. File Management

When a user creates an account, it is assigned a private workspace on HDFS and the workspace belongs to the current user. Users can remotely manage files on HDFS by Hadoop's Java API by the file management, including viewing the workflow execution file, the workflow output file and so on. The file management module picture is shown in Figure 8.

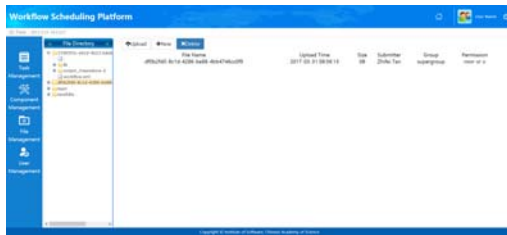


FIGURE VIII. FILE MANAGEMENT MODULE

## IV. TESTING CASE

The testing case is running a single WordCount program of the Spark type in two different ways: the system and the command line.

### A. The System

Users drag and drop the visible component to draw the workflow graph. DAG is shown in Figure 9.

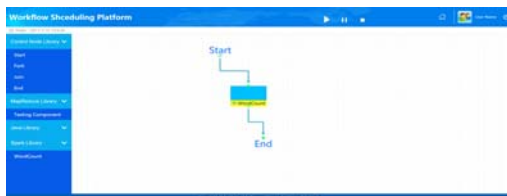


FIGURE IX. DAG OF WORKFLOW.

When editing the workflow, users also need to set arguments for computing and processing program. The configuration box is shown in Figure 10.

FIGURE X. CONFIGURATION BOX

After submitting the task to the background server, users can use the function of task management module to view the task information such as the running status. It is shown in Figure 11 as follows.

Task Name	Task ID	Start Time	End Time	Status	Operation
Spark Testing	Spark-2017-01-10-10-10-10	2017-01-10 10:10:10	2017-01-10 10:10:10	Success	Cancel
Map Testing	Map-2017-01-10-10-10-10	2017-01-10 10:10:10	2017-01-10 10:10:10	Success	Cancel
Reduce Testing	Reduce-2017-01-10-10-10-10	2017-01-10 10:10:10	2017-01-10 10:10:10	Success	Cancel

FIGURE XI. TASK MANAGEMENT MODULE.

### B. The Command Line

Users are supposed to prepare two files: workflow.xml and job.properties. The xml file is mentioned before. It mainly defines the procedure of executing task. The properties file configures the environment of cluster. It is required by the command line. Users must put them to the HDFS platform. Steps of running task by command lines is shown as follows.

- step1. Put workflow.xml to HDFS:  
hadoop fs -put /user/root/oozie/tanzhifei/workflow.xml
- step2. Put job.properties to HDFS:  
hadoop fs -put /user/root/oozie/tanzhifei/job.properties
- step3. Put JAR file to HDFS:  
hadoop fs -put /user/root/oozie/tanzhifei/lib/WordCount.jar
- step4. Run the task:  
oozie job -oozie http://124.16.136.145:11000/oozie -config /user/root/oozie/tanzhifei/job.properties -run

FIGURE XII. STEPS OF RUNNING TASK.

### C. Result Of Testing

The testing is executed under the experimental environment (deployed five clusters: 12 kernel 47GB,

RedHat6.4 OS, 100Mbit/s network card). Testing data is size of 10M TXT document and a total of 10 groups. The average uptime of the system testing and the command line testing is shown in Table 2 as follows.

TABLE II. UPTIME OF TESTING

Ways	Time(s)
The system	24
The command line	20

Under the same experimental conditions, the task execution time of the command line is shorter than the system. The system does not improve efficiency of the computing and processing program for processing data. Because the system is dependent on the Oozie underlying engine service and it has to call necessary functions. But users don't have to prepare XML and properties files, they will be automatically generated by the background of the system.

## V. CONCLUSION

Although the system can't improve the efficiency, it can reduce the error rate of manual operation. Moreover, it provides visual interfaces and services of monitoring, management, login and so on that make the system easily to be used to manage the workflow task for users. Therefore, if the process of analyzing data is complex, the system will be superior to the command line. After all, manual input command lines are very inconvenient and error-prone.

## REFERENCES

- [1] Y. Li. Design and Implementation of SSM Framework in Web Application Development [J]. Computer technology and development, 2016, 26(12):190-194.DOI:10.3969/j.issn.1673-629X.2016.12.041.
- [2] J. Chen, J. Wu, Y. Su, et al. The Research and Practice of Workflow Engine for Big Data Platform [J]. Telecommunications Science, 2016, 32(2):134-139.DOI:10.11959/j.issn.1000-0801.2016080.
- [3] W. J. Zhang. Design and Implementation of Visual Data Processing Platform Based on MapReduce [D]. Xi'an Electronic and Science University, 2014.
- [4] David F. Taobao front-end team. JavaScript authoritative guide [M]. Beijing Machinery Industry Press, 2016
- [5] Islam M, Huang A K, Battisha M, et al. Oozie: towards a scalable workflow management system for Hadoop[C]// ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies. ACM, 2012:4.
- [6] Vaddeman B. Creating and Scheduling Workflows Using Apache Oozie[J]. 2016.
- [7] Porritt, S.: jsPlumb JavaScript library: Visual connectivity for webapps (2011),