



UMDISW: A Universal Multi-Domain Intelligent Scientific Workflow Framework for the Whole Life Cycle of Scientific Data

Qi Sun, Yue Liu^(✉), Wenjie Tian, Yike Guo, and Bocheng Li

School of Computer Engineering and Science, Shanghai University,
Shanghai 200444, China

sunqichn@163.com, yliu@staff.shu.edu.cn, tianwenjie1997@163.com,
y.guo@imperial.ac.uk, 1124129934@qq.com

Abstract. Existing scientific data management systems rarely manage scientific data from a whole-life-cycle perspective, and the value-creating steps defined throughout the cycle constitute essentially a scientific workflow. The scientific workflow system developed by many organizations can well meet their own domain-oriented needs, but from the perspective of the entire scientific data, there is a lack of a common framework for multiple domains. At the same time, some systems require scientists to understand the underlying content of the system, which virtually increases the workload and research costs of scientists. In this context, this paper proposes a universal multi-domain intelligent scientific data processing workflow framework (UMDISW), which builds a general model that can be used in multiple domains by defining directed graphs and descriptors, and makes the underlying layer transparent to scientists to just focus on high-level experimental design. On this basis, the paper also uses scientific data as a driving force, incorporating a mechanism of intelligently recommending algorithms into the workflow to reduce the workload of scientific experiments and provide decision support for exploring new scientific discoveries.

Keywords: Scientific workflow · Intelligent · Scientific data · Universal framework

1 Introduction

Scientific research has entered the era of big data. Instruments and equipment with ever-increasing data collection capabilities and ever-evolving computing facilities and simulation methods are important sources of scientific big data, causing explosive growth in data size, which is happening in different scientific domains [1–3]. At the same time, because scientific data faces greater “breadth” and “depth” than commercial data, the processing patterns and methods of scientific data are also diversified. It can be said that the management and processing of scientific data now face enormous challenges.

In this case, many organizations have carried out a series of work on the structure and system of scientific data management from different perspectives, and developed many scientific data management systems or analytical frameworks. Such as SkyServer for managing SDSS data [4], Paradise for storing geographic information [5], Google's virtual digital earth system Google Earth for visualization [6], Apache's big data high-performance computing framework Hama [7], etc. These systems perform well at some point in management, analysis, or visualization, but lack a complete solution from the perspective of the whole life cycle of scientific data. Scientific data creates value, and the whole lifecycle of scientific data defines the various steps in creating value, including importing, storing, processing, visualizing, analyzing, re-storing, and so on, as well as the conditions associated with each step. These steps essentially constitute a workflow, or data flow, information flow.

The concept of workflow has been widely used in many fields, such as business processes, industrial manufacturing, scientific research, medicine, etc. [8]. For scientific workflows, the most common representation is to create a high-level graph composed of directed graphs, related nodes and edges that define the sequence and interaction between the various steps associated with the scientific workflow, and this graphy defines the sequence and interaction between the various steps. In addition, because of the diversity and complexity of scientific research, only one directed graph is not sufficient to represent the processing flow of multiple scientific domains, so additional descriptors are needed to specifically identify and control the nodes of a series of steps. The information exchange standard proposed by the Open Provenance Model (OPM) core specification [9] for data traceability and the concepts and terminology defined in the S88 standard [10] for production recipe process can be applied to the scientific workflow. The scientific workflows that will be presented later in this paper also refer to the relevant content of these two standards.

It turns out that it is feasible to use workflow to represent the processing of scientific data, many organizations continue to develop their own field-oriented scientific workflow system based on their needs. From the combination of scientific workflows, it is divided into text-based combinations such as BPEL4WS [11], DAGMan, SCUFL; graphics-based combinations such as Triana [12], VisTrails, Kepler [13]; semantic-based combinations such as K-WF, Pegasus, Taverna [14]. These scientific workflow systems have brought great convenience to the processing of scientific data to a certain extent, but there are also some problems:

- It is more difficult to meet the needs of many types of users. Domain experts are more focused on domain-related research than on the scheduling of underlying resources related to specific calculations.
- Most of the existing scientific workflows are only for a single domain, lacking a universal framework.
- Lack of a more efficient method of scheduling. Users usually need to choose different methods to compare to select the better one, so that the workload of scientific experiments is increased.

In summary, based on the previous research [15], this paper proposes a Universal Multi-Domain Intelligent Scientific data processing Workflow (UMDISW), which solves the problems of current scientific workflow from the perspectives of model, structure and application. The second section of this paper describes the state and status of the proposed workflow in previous research. In the third section, the model of the UMDISW is analyzed. And in the fourth section, we describes the structure of the UMDISW. What's more, the fifth section introduces the application scenario of the UMDISW, highlighting the characteristics of intelligence, and the sixth section is to introduce the workflow implementation in combination with the previous research. Finally, we have a conclusion in the seventh section.

2 The Status of UMDISW in System Architecture

In previous research, we proposed a scientific big data management system architecture for multiple domains and roles, The architecture is divided into four areas: Basic Service Function Area (BSFA), Storage and Access Area (SAA), Query Function Area (QFA), and Analysis Function Area (AFA). Each area has its own corresponding function. For example, the AFA is responsible for processing scientific data in different domains using machine learning or domain methods. On this basis, we integrated UMDISW proposed in this article in the AFA, and the AFA is updated to have three component: the Asset Loader (AL), the Pipeline Manager (PM), and the Pipeline Tool (PT). These three components show the composition of the UMDISW in the architecture, where the AL is used to get the data and algorithms needed by the experiments; the PM is responsible for the design, build, and integration of the workflow; the PT is responsible for providing the operating environment and execution engine. Each component has its own port and interface to interact with outside. The model, structure and implementation of the UMDISW in the AFA will be detailed later.

3 The Model of UMDISW

The UMDISW proposed in this paper consists of the following modules: workflow and task, data flow and information flow, data node and algorithm node. The functions of each of the modules will be described in the next few subsections.

3.1 Workflow and Task

The processing of scientific data can be divided into modules that depict the processing details, and a workflow is the sum of all these modules. On this basis, the workflow is essentially a container that defines the scope of data processing under the user's decision. Typically, scientific data processing will have one or more tasks, and the workflow should also consist of at least one task.

A task represents a collection of steps performed on or by an allocated resource. In science, these resources often refer to scientific devices, instrumentation, scientific software, scientific data and algorithms, and are distinct in different scientific domains. When a resource is shared by multiple tasks, the allocation of resources to perform a given task depends on its availability, suitability, and priority of the task to be completed. The start and execution of the task will be postponed until the resources are allocated to it.

3.2 Data Flow and Information Flow

Depending on the structure of the workflow, its execution will involve the transfer of entities (data, information, etc.) because they are generated or used during the execution of the workflow, and this transfer forms the data and information flows.

The data flow usually refers to the process or location of the data in the workflow, so it's necessary to define the nodes to represent the start and end of the data flow; define the input and output locations of subtasks to represent the current progress of the data; define connect lines to represent the movement of data between two subtasks, which is also a factor driving the workflow; define work areas to distinguish between different workflows. In order to describe the model of UMDISW in a graphical way in the following discussion, the data flow is represented by solid lines, which start from the start node (represented by a solid circle), or from the output of the subtask module (represented by rectangle), or from the data node (represented by rounded rectangle), and terminate at the end node (represented by triangle) or subtask module input. What's more, different workflows are divided into different areas (represented by lanes).

The information flow in the workflow also requires the above definition to represent the start and end, movement, and the classification of information. At the same time, the definition of data nodes and algorithm nodes is needed to display the development and changes of information during the execution of the workflow, such as data generation and update of algorithm parameters, and these changes based on the definitions form the workflow's information flow. Similarly, for graphical description, the information flow is designated as a dashed dotted arrow that begins at the subtask module and ends at the information module (represented by dashed rectangle).

3.3 Data Node and Algorithm Node

Defining data nodes and algorithm nodes for two purposes: first, they contain data inputs and algorithm inputs for a module in the workflow, or metadata information that is used to standardize the module's output; second, they contain addresses that point to the actual stored data and algorithms in the databases. An algorithm node and several data nodes combine to form a module of the workflow, that is, a subtask in the task.

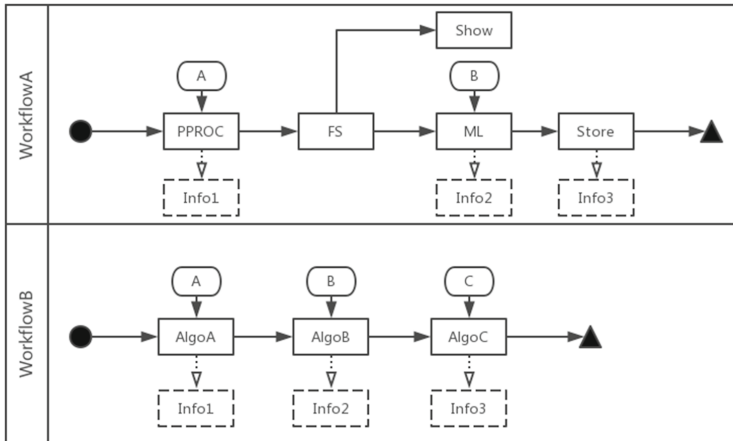


Fig. 1. Workflow sample diagram

Algorithm Node. According to the above description, the number of algorithm nodes in a workflow should be equal to the length of the workflow (remove the start and end nodes). In addition to the metadata information of the corresponding algorithm and the actual stored reference address, the algorithm node needs to indicate what format of input and output are needed in this step to improve the efficiency of the user to build the workflow. The algorithm corresponding to the algorithm node here may be a file or an algorithm program that has been integrated in the system.

Data Node. Data nodes are divided into data input nodes and data output nodes. The data input node can be combined with the algorithm node to become a subtask or a module of the workflow, and it may be a data file, a database table or a value; the data output node is an online result set or image (available for download), but it is not displayed in the workflow, and will only be displayed in the user interface when clicking on an algorithm node.

3.4 Example

Figure 1 shows two sample workflows that use machine learning methods to illustrate the workflow symbols mentioned earlier. There are two workflows, WorkflowA and WorkflowB, separated by lanes.

WorkflowA is a branched workflow that includes five subtasks: PPROC, FS, Show, ML, Store. Each subtask uses the corresponding algorithm for calculation and processing. WorkflowA starts from the subtask PPROC, which needs to input the data set pointed by the data node A. After a period of processing, the processing result is pushed to the subtask FS, and the corresponding information module Info1 is generated.

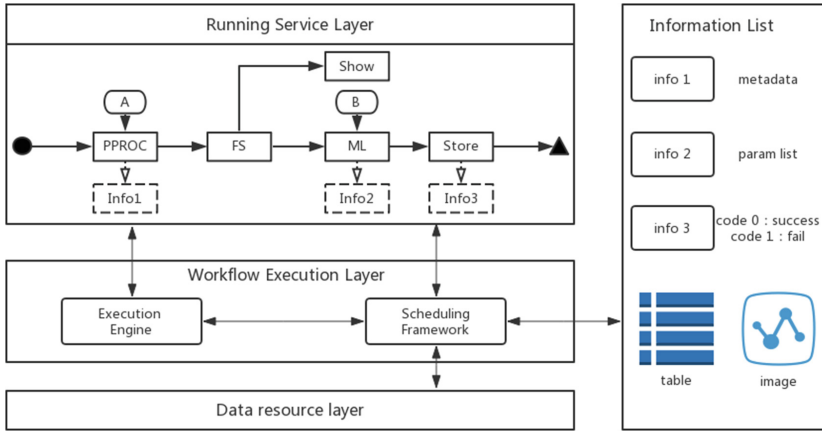


Fig. 2. Hierarchical structure of UMDISW

When the subtask FS gets the output of the previous step, it takes itself as its own input and starts executing. It should be noted that when the FS is executed, the output will be passed to the two subtasks, Show and ML, the former displays the results of the previous step to the page, and the latter combines the result of the previous step with the newly entered data B as its own input, and the result of the operation is passed to the next Store, and the corresponding information module Info2 is generated.

After executing the ML, the Store subtask will receive the result of the ML, which will save it to the database and generate the information module Info3, and end the WorkflowA.

The solid line in Fig. 1 reflects the flow of data during the execution of the workflow, which shows the process of data changes; the combination of the information modules corresponding to the dotted lines is the embodiment of the information flow in the workflow, which shows the gain process of the information.

WorkflowB is a linear workflow, and its execution process is similar to WorkflowA, except that each subtask has data input and information output, and there is no branch.

Although this sample workflow is only for machine learning processing, the nodes, modules, and regions shown in the figure provide the necessary functions to model the workflow of various scientific domains and non-machine learning methods. In addition, the graphical representation of the workflow provides a very compact view of the features and controls used to implement the workflow. In principle, since any organized process can be projected as a workflow, the proposed workflow model can facilitate the development of a common framework for managing related processes in different domains. Such a framework can promote understanding and application of cross-domain knowledge processing.

4 The Structure and Execution of UMDISW

In order to eliminate the inconsistency between scientific data and methods in different domains, the UMDISW proposed in this paper is transparent to the bottom and inside, and can be divided into three layers to meet the needs of different domains.

As shown in Fig. 2, UMDISW consists of three layers: Running Service Layer (RSL), Workflow Execution Layer (WEL), and Data Resource Layer (DRL). Each layer interacts with each other to complete the execution of the workflow and save the information.

4.1 Running Service Layer

The Running Service Layer is the only open part of the UMDISW, located at the top of the entire structure, providing a series of methods to interact with the user. Its core function is to instantiate user-created workflow models and display workflow information.

The RSL is web-based, thus allowing users to dynamically design workflows on the browser and to support visual editing operations such as adding, copying, dragging, and double-clicking on graphical elements representing components. When the page is loaded, an area is first drawn in the page according to the initialization incoming data as the working area of the workflow, and the related components are instantiated. At this time, the user needs to click the component to complete the binding operation of the data and the algorithm. When the above operation is completed, the components and connect lines in the entire area will be locked, and the workflow will be instantiated by the RSL and passed to the WEL.

In addition, as a layer of display information, the RSL also supports the relevant data information transmitted on the underlying layer and rendered on the page.

4.2 Workflow Execution Layer

Workflow execution typically involves one pass of all modules modeled in the workflow or the whole life cycle of scientific data. For such a loop, the workflow starts from the initial state and is converted by all or part of the finite state for each module. Workflow execution may be affected by a number of specific factors, such as the value of various parameters, the final application, decision logic, the execution engine, and the user's input at a particular point in the workflow. WEL is to implement the built-in or implicit logic in the workflow, and to perform a cycle of the workflow. The WEL consists of two core modules: Execution Engine and Scheduling Framework.

The Execution Engine's responsibility is to obtain the workflow objects instantiated in the RSL and combine them with the data and algorithms passed by the Scheduling Framework to form an executable workflow. When the combination is completed, the user selects whether to start executing the workflow.

If start, the Execution Engine starts executing the workflow until to the end or an abnormality occurs in the middle. Therefore, workflow execution is based on user-triggered events. When the execution ends, the execution engine passes the results to the Scheduling Framework for further processing.

According to the previous description, one duty of the Scheduling Framework is to pass data and algorithms to the Execution Engine to form an executable workflow. In addition, the Scheduling Framework is responsible for further processing the execution results, including passing it up to the RSL for visualization, and passing it down to the DRL for storage. However, no matter what processing is performed, the Scheduling Framework can form an information flow corresponding to the workflow as shown in the right half of Fig. 2 according to the original information and the result information, including the state of the data, the update of the parameters, and the results of each subtask.

4.3 Data Resource Layer

Facing heterogeneous multi-source scientific data, the DRL implements the maintenance of multi-domain scientific data storage and algorithms. Due to the different types of scientific data in different domains, this layer provides a variety types of databases, including relational databases, non-relational databases, and graph databases, and provides a unified access interface for data to achieve transparent operations, including data input and output interface, algorithm call interface, model save interface, etc. The information required by the other two layers are stored in the DRL.

5 The Application Scenario of UMDISW

Depending on the application scenario, the UMDISW can be transformed into different forms depending on the selection, including fully automated workflow, semi-custom workflow, and fully custom workflow. The difference between these three forms is that the amount of user interaction is different when building the workflow.

5.1 Fully Automated Workflow

In many scientific fields, a set of specifications may have been formed for some processes, and experts agree that the process is reasonable and will not change the process when doing experiments. Based on this situation, the UMDISW can form a template workflow based on domain specifications. The template workflow defines the corresponding tasks and subtasks according to the requirements of the experts, and the location of the data nodes and algorithm nodes also have corresponding requirements, and the data flow and information flow in the template workflow need to meet the domain specifications.

For the template workflow, the user only needs to select the workflow template in advance, and then the fully automated workflow as shown in the Fig. 3

will appear on the RSL. Figure 3 shows the gravitational wave data processing workflow. Each subtask, data node, algorithm node and information flow in the workflow are pre-defined. The user only needs to specify the data address corresponding to the data node, and then click to start running. The template workflow automatically obtains the data of the specified address from the DRL, and then passes the template instance and data to the WEL to complete the workflow execution, and finally return the result to the RSL to complete the display of the information flow.

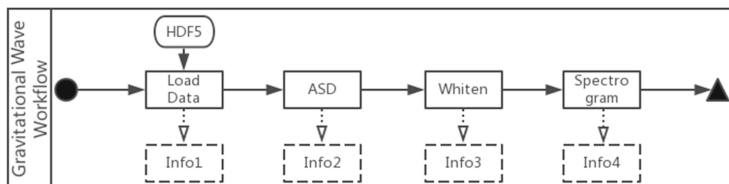


Fig. 3. Fully automated workflow

One use of this fully automated workflow is to examine the generalization capabilities of the same set of algorithms or processes for different data. Because the subtasks of the template workflow are fixed, the effect of the workflow in this case can be compared when accepting different data.

5.2 Semi-custom Workflow

In addition to the above-mentioned domain specification process, scientists may also customize a scientific data analysis process. The workflow in this application scenario is called a semi-custom workflow. The workflow requires the user to define the number of steps of the workflow, data nodes and algorithm nodes, and then RSL will generate the specified workflow based on these parameters. However, it should be noted that the algorithms and data corresponding to the subtasks of the created workflow are empty, and the user needs to select the required algorithms and data. The selection steps are the same as mentioned above. The information flow is also displayed by the WEL.

One function of this semi-custom workflow is to compare the results of the same data in the case of different algorithms. As shown in Fig. 4, these two semi-custom workflows built for users have the same structure, including the number of steps and input data, the only difference is that the subtasks, the subtask in Superalloy Experiment 1 is to analyze data A using the SVM algorithm, while the subtask in Superalloy Experiment 2 is to analyze data A using the random forest algorithm, returning the results to Info1 and Info2 respectively, to compare the effects of using different algorithms when processing the same data.

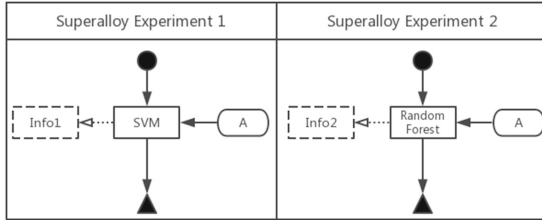


Fig. 4. Semi-custom workflow

5.3 Fully Custom Workflow

Fully custom workflows have a higher degree of freedom than semi-custom workflows. This is reflected in the fact that such workflows do not provide users with the generation of related components, and users can drag and drop to display on the RSL with their own needs, and connect the components using connectors. Of course, each node of the workflow at this time is still empty, so the user needs to select the corresponding data and algorithm. It is worth noting that after the user uses such a workflow and selects the data corresponding to the data node, the Scheduling Framework in the WEL intelligently recommends the appropriate machine learning algorithm according to the data selected by the user. So this kind of workflow is designed to provide users with an intelligent algorithm selection tool compared to the above two workflows, reducing the workload of users when conducting scientific experiments.

The recommended logic for machine learning algorithms is shown in Fig. 5. The data is first preprocessed into a conforming format, and then the data is parsed to see if feature reduction processing is required. If necessary, workflow will recommend the feature dimension reduction method of unsupervised learning class, such as FA, PCA, LDA and other topic model algorithms, or select Lasso, Ridge which are depending on the number of samples. If feature dimensionality reduction is not required, it will check if the dataset have decision attributes. If not, it will recommend clustering methods of unsupervised learning class, and recommend clustering algorithms such as k-means, hierarchical clustering, and FCM as needed. If there is a decision attribute, it will check whether the decision data belongs to a discrete class or a continuous class. If it belongs to the continuous class, workflow will recommend the regression method of supervised learning class, and recommend SVR, RF, Adaboost and other algorithms depending on the sample attributes. If the decision attribute is a discrete class, it will be recommended according to the sample type, if sample is image data, CNN is recommended, if it is time series data, RNN is recommended, and so on.

Of course, these recommended algorithms are just to give users a reference to help users make decisions. However, in practical applications, it can be found that this intelligent and fully custom workflow with recommendation mechanism does reduce the experimental steps, experiment time and workload for users who

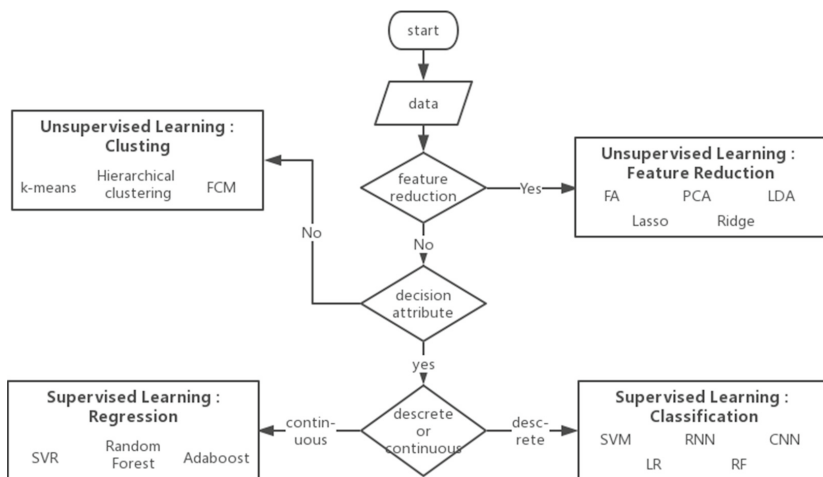


Fig. 5. The mechanism of intelligently recommending algorithms

do scientific experiments. Most importantly, it also plays a great role in exploring new scientific discoveries.

6 The Implementation of UMDISW

Since the proposal of UMDISW is based on previous research on the architecture of multi-domain scientific big data management system, the implementation of UMDISW is also dependent on the implementation and deployment of the architecture.

After integrating the content of the UMDISW into the architecture, the deployment of the entire architecture is updated as shown in Fig. 6. And as mentioned earlier, our proposed UMDISW is located in the AFA area.

As shown in Fig. 6, AFA consists of two subsystems: Spark as a workflow tool and gooFlow as a workflow manager. GooFlow is a UI component used to design flowcharts on the web page. It is based on JQuery development, and the great user experience makes the interface very easy to use. As a new generation of distributed processing framework, Spark's memory-based computing can speed up the execution of workflows, and it has an excellent machine learning library MLlib, which can be used as a tool for data analysis. The asset loader in the component diagram is replaced by an adapter "QFA-AFA Adapter". GooFlow also integrates the Spark tool with another adapter, the "Spark-gooFlow Adapter".

As can be seen from the deployment diagram, AFA connects to Bootstrap through the adapter "QFA-AFA Adapter" and the interface "asset provider". Through adapters and interfaces, gooFlow can get the data and algorithms stored in SAA that scientists want to experiment with. For the visualization of the workflow, gooFlow can be presented directly to the "Chinese visCloud" or Bootstrap via the interface "pipeline provider" and the adapter "Visualization

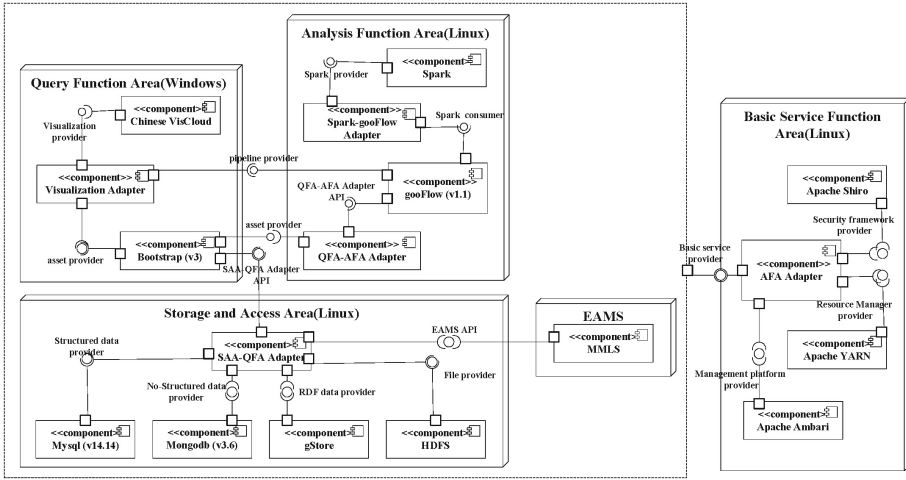


Fig. 6. The deployment of architecture and the implementation of UMDISW

Adapter”. At the same time, the UMDISW also obtains the basic services and resource management functions in the BSFA through the interface “Basic service provider” and adapters.

7 Conclusion

Managing scientific data from a whole life cycle perspective is very helpful in mining the value of it. This paper proposes a General Multi-Domain Intelligent Scientific Workflow to help scientists create scientific value by dividing the whole life cycle steps.

The highlight of this article is: (1) using graphics and descriptors to model scientific workflows, eliminating the differences in heterogeneous scientific data, and building a universal framework to support consistency in multi-domain data processing; (2) transparency of the underlying architecture enables scientists to focus on high-level experimental design, reducing the time cost of learning and using the workflow; (3) incorporating a data-driven mechanism to intelligently recommend algorithms, reducing the amount of labor required for scientists to experiment and providing decision support for exploring new scientific discoveries.

Acknowledgments. This work is supported by the National Key Research and Development Plan of China (Grant No. 2016YFB1000600 and 2016YFB1000601).

References

1. Andreeva, J., Campana, S., Fanzago, F., Herrala, J.: High-energy physics on the grid: the ATLAS and CMS experience. *J. Grid Comput.* **6**(1), 3–13 (2008)

2. Chen, J., Wang, W., Zi-Yang, L.I., An, L.I.: Landsat 5 satellite overview. *Remote Sens. Inf.* **43**(3), 85–89 (2007)
3. Bengtsson-Palme, J., et al.: Strategies to improve usability and preserve accuracy in biological sequence databases. *Proteomics* **16**(18), 2454–2460 (2016)
4. Ivanova, M., Nes, N., Goncalves, R., Kersten, M.: MonetDB/SQL meets SkyServer: the challenges of a scientific database. In: *International Conference on Scientific and Statistical Database Management*, p. 13 (2007)
5. C. T. P. Team: Paradise: a database system for GIS applications. In: *ACM SIGMOD International Conference on Management of Data*, p. 485 (1995)
6. Patterson, T.C.: Google earth as a (not just) geography education tool. *J. Geogr.* **106**(4), 145–152 (2007)
7. Suchanek, F.M., Weikum, G.: Knowledge bases in the age of big data analytics. *Proc. VLDB Endow.* **7**(13), 1713–1714 (2014)
8. Schwartz, D.G., Te'Eni, D.: Encyclopedia of knowledge management. *Online Inf. Rev.* **5**(3), 315–316 (2006)
9. Moreau, L., et al.: The open provenance model core specification (v1.1). *Future Gener. Comput. Syst.* **27**(6), 743–756 (2011)
10. Batch control part 1: models and terminology (1995)
11. Reichert, M., Rinderle, S., Dadam, P.: On the modeling of correct service flows with BPEL4WS. In: *EMISA 2004, Informations system in E-Business und E-Government, Beiträge des Workshops der GI-Fachgruppe EMISA, 6–8 October 2004, Luxemburg*, pp. 117–128 (2004)
12. Taylor, I., Shields, M., Wang, I., Harrison, A.: Visual grid workflow in Triana. *J. Grid Comput.* **3**(3–4), 153–169 (2005)
13. Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., Mock, S.: Kepler: an extensible system for design and execution of scientific workflows. In: *SSDBM*, pp. 423–424 (2004)
14. Turi, D., Missier, P., Goble, C., De Roure, D., Oinn, T.: Taverna workflows: syntax and semantics. In: *IEEE International Conference on e-Science and Grid Computing*, pp. 441–448 (2008)
15. Sun, Q., Liu, Y., Tian, W., Guo, Y., Lu, J.: Multi-domain and sub-role oriented software architecture for managing scientific big data. In: Ren, R., Zheng, C., Zhan, J. (eds.) *SDBA 2018. CCIS*, vol. 911, pp. 111–122. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-5910-1_10