International Journal of Control and Automation
Vol.9, No.3 (2016), pp.77-88
http://dx.doi.org/10.14257/ijca.2016.9.3.08

Visual Flow Design in Autonomous Underwater Vehicle Collaborative Design Platform

Zhenzhen Xu, Jie Dai, Xiaowei Zhao and Xiujuan Xu*

School of Software, Dalian University of Technology
xzz@dlut.edu.cn, ddadlolita@gmail.com,
xiaowei.zhao@dlut.edu.cn, xjxu@dlut.edu.cn

Abstract

Autonomous underwater vehicle (AUV) collaborative design platform which is a novel system to solve collaborative design problem of AUV is established. The platform hardware structure is introduced and the main sub-system named visual flow design system is described in detail. The function structure of visual flow design system is presented. Two main modules in the visual flow design system including flow modeling and flow monitoring are introduced. For flow modeling module, basic widgets and task widgets are designed to realize visual flow design and self-definition of computing tasks based on the workflow theory and XML technology. Flow monitoring module can realize real-time monitoring of flow state, flow execution time and IP of workstation assigned to each task. Flex-based visual flow design makes AUV collaborative design platform easy to use and helps the designers improve design efficiency and shorten development period.

Keywords*: collaborative design, visual flow design, AUV, workflow, flex*

1. Introduction

The design of autonomous underwater vehicle is a complicated systematic project. Different subsystems relate to different disciplines. The multiple parts of the whole system effect and restrict each other. In the design process, it is required to coordinate the constraint relationship among different disciplines, optimize the flows and variables, and ensure harmonious and unified of each subsystem. AUV collaborative design platform based on B/S mode is proposed in this paper. This is a novel platform to realize the collaborative design for AUV specially. It can realize distributed collaborative design of multiple AUV designers and parallel computing of multiple AUV flows.

Because the operations of a flow design including definition, modification, submission and monitoring are all user-oriented, this paper mainly focuses on the visual flow design problems in AUV collaborative design platform. The system is required to be used easily, which means the designers can freely drag and drop widgets to design a flow, modify a flow through the graphical design interface, view and monitor the flow state and so on. Therefore, the core issue is to solve the visual flow design problems.Current implementations of flow design tools based on B / S mode can be divided into three categories:

(1) Tools based on JavaScript. This method brings low compatibility because of the different supporting degrees of JavaScript on different browsers. This kind of design tool makes the whole browser very slow when complex operations are adopted.

(2)Tools based on adding plug-ins (Applet, SilverStream, etc.) to browsers. The advantage of this approach is that designers can design a more complex flow and the efficiency is high, while the disadvantage is that users must install plug-ins which makes supporting degree and compatibility very low.

(3)Tools based on Flex [1]. Flex is an efficient free framework for building rich client applications based-on B/S. It has good compatibility with browsers, excellent interface, sensitive reaction and higher efficiency [2-3].

According to the advantages and disadvantages of the three methods mentioned above, this paper adopts Flex technology to realize visual flow design for AUV. Meanwhile, workflow theory is referenced and XML language is used during the implementation of this system. Thus, AUV flows can be displayed graphically on the browser, which makes modeling and monitoring much easier. The visualization level and work efficiency of AUV collaborative design platform can be improved eventually.

2. System Function Structure

2.1 Hardware Structure of the Platform

The hardware structure of AUV collaborative design platform is shown in Figure 1. The platform is a distributed system which includes client, server and workstation. These three parts are all connected to a local area network. The client corresponds to the designer's own office PC and the workstation has been installed multiple design software. The server is responsible for scheduling multiple AUV flows submitted by different designers and assigning computing tasks in the flows to the appropriate workstations to realize parallel execution. At last, the calculation results of a flow will be returned to the server to save.

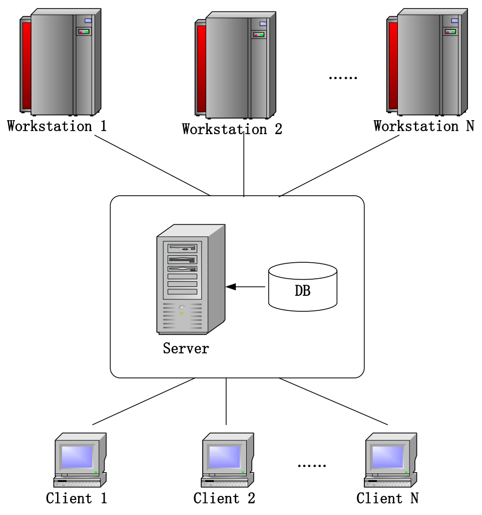


Figure 1. Hardware Structure of the Collaborative Design Platform

2.2 Function Structure of Visual Flow Design System

Designers design flows through the client browsers, and then submit them to the server. The visual flow design system works on the client and it is an important subsystem of AUV collaborative design platform. It includes two main functional modules: flow modeling and flow monitoring. Figure 2 shows function structure of Flex-based visual flow design system.

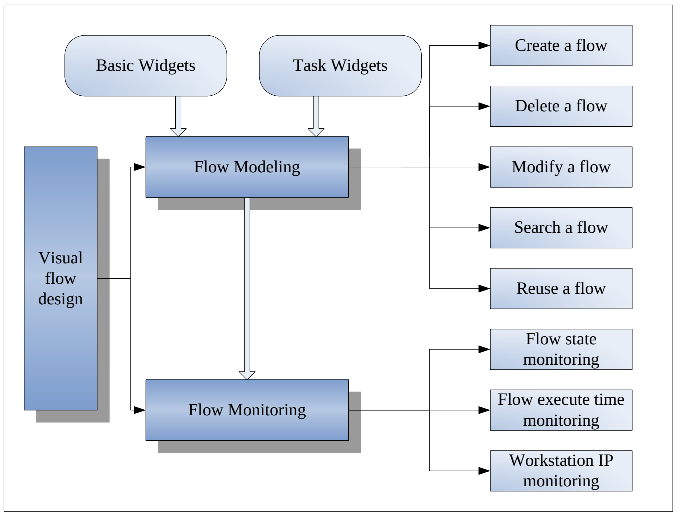
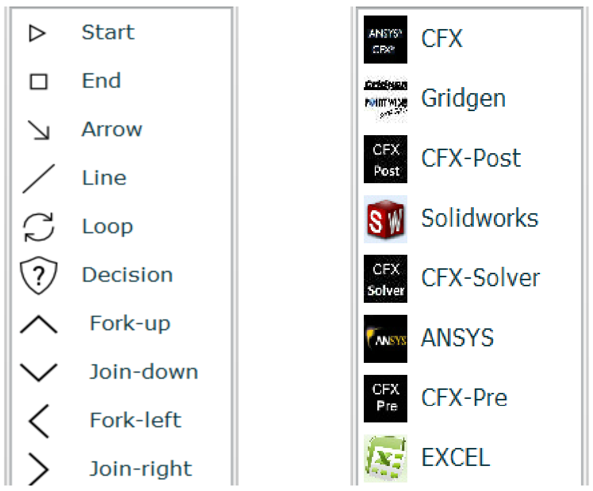


Figure 2. Function Structure of Visual Flow Design System

3 Flow Modeling

AUV flow consists of multiple AUV computing tasks. Flow structures may contain sequence, selection, loop, branch, nestification and other complex structure. Flow modeling function includes creating a flow, deleting a flow, modifying a flow, retrieving a flow and reusing a flow. For flow modeling function, the most important thing is to design widgets including basic widgets and tasks widgets which can compose a flow. All widgets are shown in Figure 3.



(a) basic widgets (b) task widgets

Figure 3. List of Widgets

3.1 Basic Widgets

Workflow theory is widely applied in enterprise management and product development [4-5]. The reference model given by WfMC (workflow management coalition) provided interchange format and read-write operation of flow modeling [6-8]. To enhance the versatility and scalability of flow modeling in AUV collaborative design platform, we defined a set of basic widgets which obey the WfMC standards, including Start Widget, End Widget, Fork Widget, Join Widget, Loop Widget, Decision Widget, and TransLine Widget, as shown in Figure 3 (a). Designers can create different nodes of AUV flow by dragging corresponding widgets to the flow modeling canvas. The function of each node is described as follows.

(1) StartNode: Each flow has a start node as a node index for parsing flow. Each flow has only one StartNode.
(2) EndNode: Each StartNode corresponds to an EndNode, which marks the end of a flow.

Both of the two nodes execute automatically without any tasks or actions.

(3) ForkNode: ForkNode is used in parallel flows marking the beginning of the parallel execution.
(4) JoinNode: Each ForkNode corresponds to a JoinNode which indicates the end of parallel execution. If there is a ForkNodes in a flow, there must also be a JoinNode.
(5) DecisionNode: It is a kind of decision-making node which includes the attribute of boundary value. The computer will automatically determine which branch to go according to the variables value and the boundary value.
(6) LoopNode: In AUV flows, some of the task nodes need looping execution. A LoopNode sets the loop termination condition and the loop variables.
(7) TransLine: One TransLine is responsible for connecting two nodes. There are two kinds of TransLine including Straight Line and Fold Line.

First of all, we define a class AbstractElement that inherits from the UIcomponent which is a visual component in Flex. The class AbstractElement includes basic information such as Name, Id, Vertical coordinates, Horizontal coordinates, Length and Width and so on. Then, we define a class Node that inherits AbstractElement which implements the basic function of each node, including Draw (), reDraw (), FigureToXM() and XMLToFigure() and so on. Each of the basic node like ForkNode is a subclass of class Node, implementing the specific function of itself. The relationships between these widgets classes are shown in Figure 4.

The MXML codes [9-10] in Flex to display basic widgets are shown as follows:

```
<mx:ArrayCollection id="FigureButton">

<mx:Object id="b001" type="basic" name="StartNode" label=" start"
icon="@Embed('/assets/figure/start32.png')"/>

<mx:Object id="b002" type="basic" name="StopNode" label=" end"
icon="@Embed('/assets/figure/stop32.png')"/>

</mx:ArrayCollection>
```

Picture “start32.png” is banded to a Flex object through the code “ icon=“@Embed(‘/assets/figure/start32.png’)”, and then the picture will be shown in the view layer.

3.2 Task Widgets

In order to support the specific computational tasks of AUV flows, we defined a set of task widgets, containing all the common computing tasks in AUV designing, as shown in Figure 3 (b), This set also follows the workflow modeling standards, which means flows designed in this platform have the same interfaces as standard workflow, thereby facilitating the expansion of the platform.

Software corresponding to each task widget is installed in workstations. To improve the scalability, the system supports customization of task widget. For example, administrators can easily create a new task widget and import a picture as its icon if a kind of new software is installed in a workstation.

Dragging the task widgets to the flow modeling canvas can generate the corresponding task nodes. Task nodes are critical nodes in visual modeling and responsible for composing various AUV flows together with basic widgets.

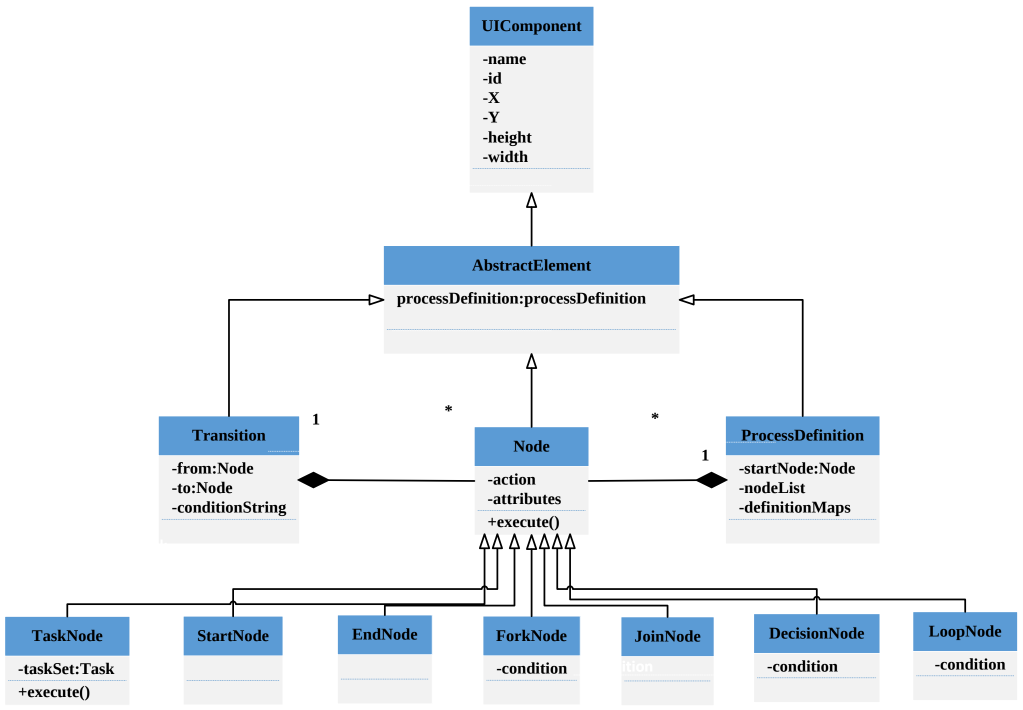


Figure 4. Relationships Between Widgets Classes

A task node is also designed as a subclass of class Node as shown in Figure 4. All computing tasks are completed under the organization of task nodes. Designers need to configure the appropriate task attributes of each task node in a flow. The attributes include input and output files, startup commands, computing variables and so on. Input and output files are required when tasks are actually executed on workstations. They should be uploaded to the server and stored in the database in advance. Implementation of file upload function is shown as follows.

```
protected function selectHandler(event:Event):void{
//url address to upload the file
var url:String = "http://localhost:8080/Designer/upload?path="+path;
var request:URLRequest = new URLRequest();
request.method = URLRequestMethod.POST;
request.url = url;
//fileList saves the files selected, supporting multi-file selecting and uploading at one time
for each(var f:FileReference in fr.fileList){
var obj:Object = new Object();
obj.name = f.name;b
obj.type = "input";
obj.status = "";
this.fileCollection.addItem(obj);
try{ f.upload(request);//upload file }
catch(error:Error){
Alert.show("upload error！");}
}
}
```

The first task of AUV flow design is composing a flow with multiple task nodes in an appropriate structure according to the needed rules in a graphical way. And then the flow will be submitted to the server and each task node will be parsed and implemented at the server. As shown in Figure 5, the statechart diagram of task node describes the entire life cycle of a task node.

4. Flow Monitoring

4.1 Procedure of Flow Monitoring

Flow monitoring module is responsible for monitoring flow status, flow execute time and workstation IP. Different colors of task nodes are used to identify different task states, such as waiting to be executed, under executing and executing completed. Flow execute time monitoring means the interface will display the begin time, end time and time length of a computing task. And the interface also displays IP of workstation assigned to each task. In addition, if a task node has not responded for a long time, it means a failure occurred. Monitoring interface can also keep track of the failed node to facilitate the designers to restart a failed flow.

The procedure of flow monitoring is described as follows.

(1)Designers submit a flow to Web server.
(2)Web server saves the flow information to the database.
(3)Server assigns tasks in flows to appropriate workstation.
(4)A task is executed on a workstation, and the executing result will be reported to the server when the task is finished.
(5)Server calls the message processing components, which can notify the web server of message type through URL.
(6)Web server pushes message to the client, and the client update the corresponding view components according to the message type. Thus, real-time changes can be seen in the monitoring interface.

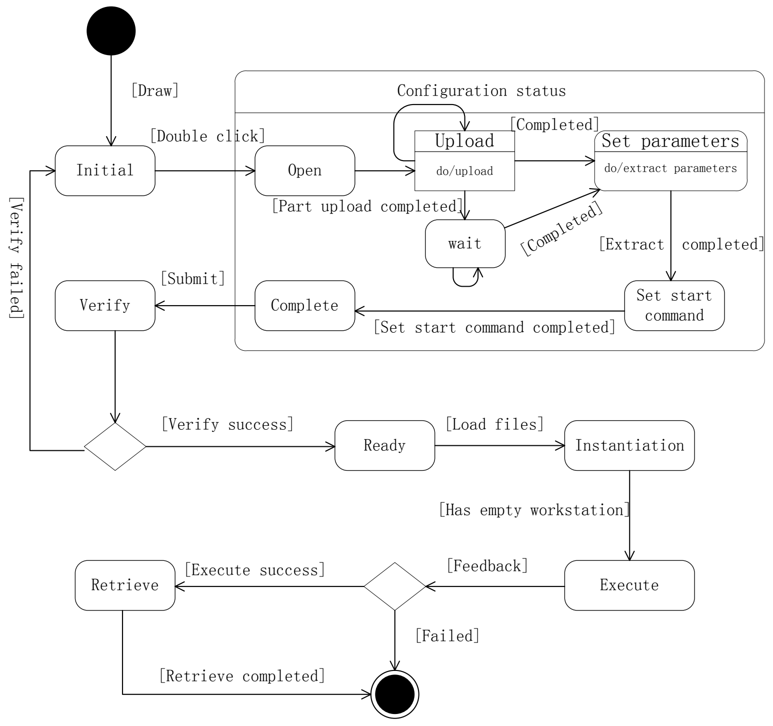


Figure 5. State Chart Diagram of Task Node

For example, request can be sent to URL address:

http://localhost:8080/Designer/message/push?type=WAITING&&FlowId=1&TaskId=3

It represents the task with task id 3 in the flow with flow id 1 is waiting to be executed. The web server will parses the URL and push real-time monitoring information back to the client through the message push mechanism. This process is a continuous cycle until all tasks in a flow are completed.

4.2 Implementation of Flow Monitoring

Flow monitoring relies on BlazeDS message push mechanism. BlazeDS is a java remoting and web messaging technology based on server, making the Java applications running background and the Flex applications on the browser can communicate with each other. There is a Message-proxy.xml file in the BlazeDS configuration files, which defines the endPoint message channel opening a route between the client and the server by Source and Destination.

Message Consumer component is binded in the client's monitoring interface. Publisher component in the Model layer of PureMVC [1] is used to listen to the messages pushed by web server. Consumer component will get real-time information and display different status of tasks as soon as Publish component catches the messages. Thus, the purpose of flow monitoring is achieved.

A global variable MESSAGE_LIST with the type of set is defined in memory to store the messages. When a message is pushed, it is removed from MESSAGE_LIST. The information of messages can be pushed to client is shown include:

(1) WAITING: task is waiting to be executed.

(2) EXECUTING: task is under executing.

(3) FINISHED: task executing has completed.
(4) COMPLETED: the whole flow executing has completed.
For example, when a "COMPLETED" message is submitted to the web server via a URL, the message will be added to MESSAGE_LIST and also written to the database, updating the definition file of the flow instance, to ensure consistent between database and memory.

5. An Example of Application

Take the AUV hydrodynamic calculation flow "Gridgen -> CFX-Pre -> CFX-Solver -> CFX-Post" as an example, Figure 6 shows the user interface of hydrodynamic calculation. In the middle of the interface is modeling canvas, designers can design a flow by dragging the basic widgets and task widgets on the left of the interface. This is a typical loop structure flow which includes four task nodes (Gridgen, CFX-Pre, CFX-Solver, CFX-Post), a LoopNode, a StartNode , an EndNode and several TransLines.

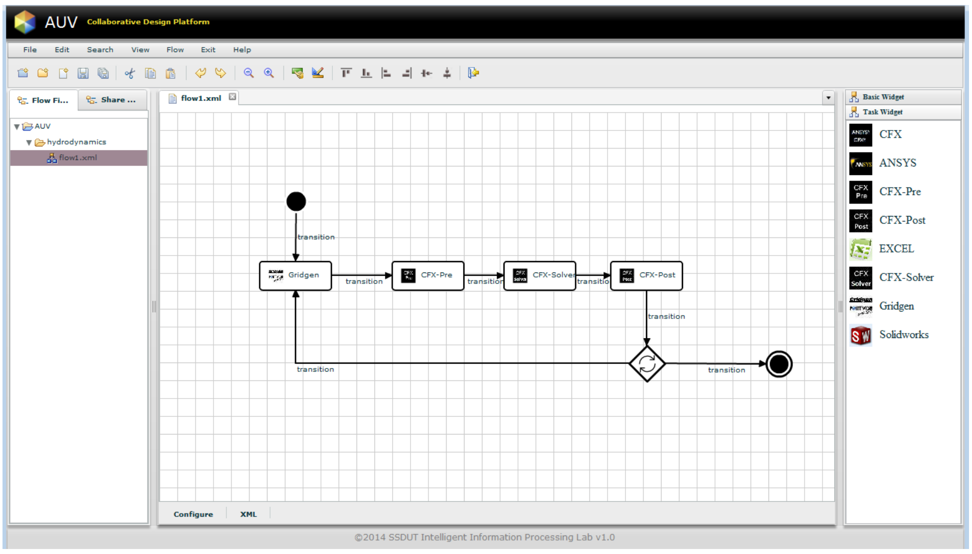
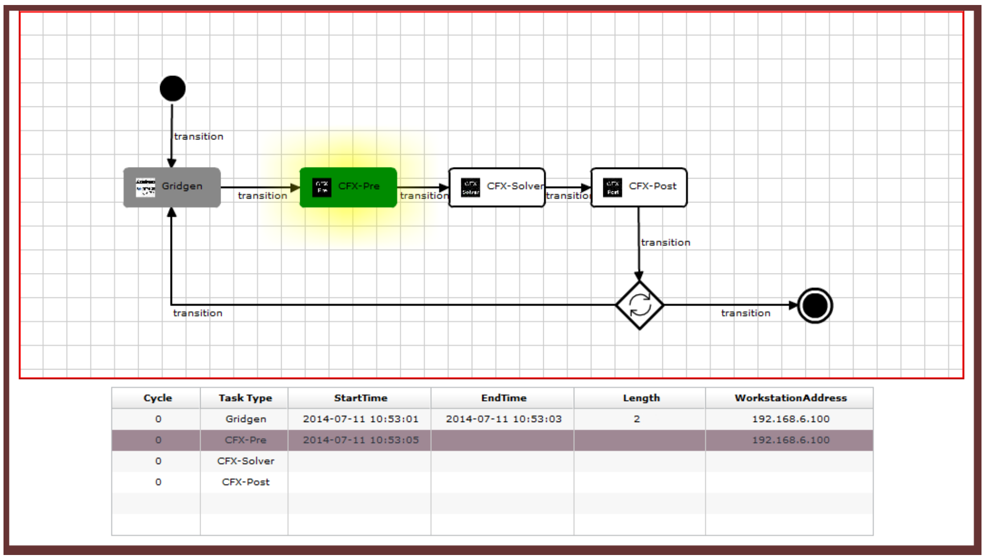


Figure 6. Interface of Flow Modeling

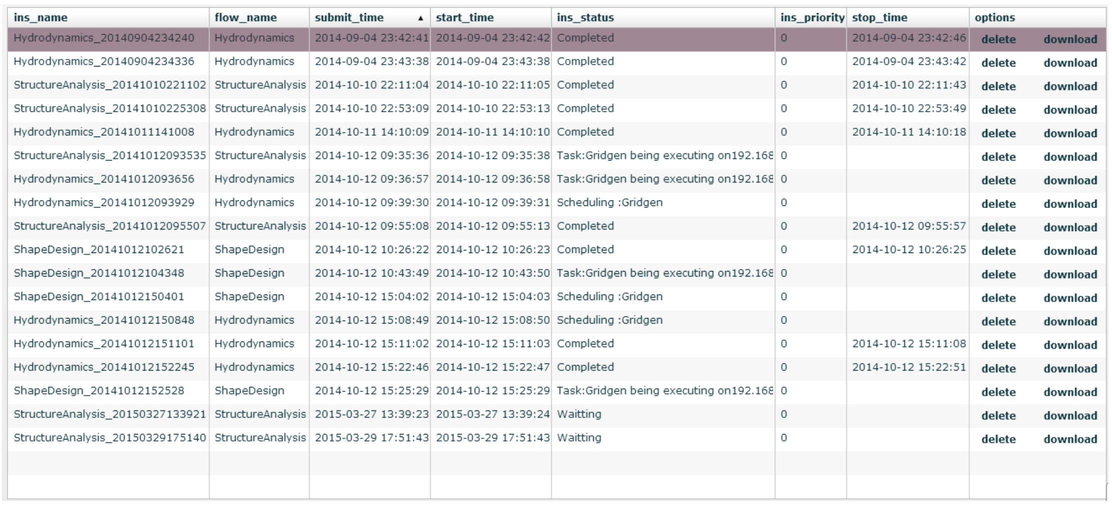
Figure 7 is the flow monitoring interface when the first task "Gridgen" is under executing. The color of task node "Gridgen" has changed to green. As displayed on the interface, this task has assigned to a workstation whose IP is "192.168.6.100".

Figure 8 shows the information of multiple flows that have been submitted to the server. It can be seen from the figure that the flow instance name, flow name, submission time, start time, flow instance status (waiting/scheduling/executing/completed), flow instance priority (default value is 0) and complete time. As we can see, a kind of flow can generate multiple flow instances and the name of flow instance includes flow name and the time when a flow instance has been created. Furthermore, options mean two operations about the files including "delete" to delete the flow instance and "download" to download the flow file and the configuration files which are compressed into a ZIP file with the same name of the flow instance.



Cycle	Task Type	StartTime	EndTime	Length	WorkstationAddress
0	Gridgen	2014-07-11 10:53:01	2014-07-11 10:53:03	2	192.168.6.100
0	CFX-Pre	2014-07-11 10:53:05			192.168.6.100
0	CFX-Solver				
0	CFX-Post				

Figure 7. Interface of Flow Monitoring



ins_name	flow_name	submit_time ▲	start_time	ins_status	ins_priority	stop_time	options	
Hydrodynamics_20140904234240	Hydrodynamics	2014-09-04 23:42:41	2014-09-04 23:42:42	Completed	0	2014-09-04 23:42:46	delete	download
Hydrodynamics_20140904234336	Hydrodynamics	2014-09-04 23:43:38	2014-09-04 23:43:38	Completed	0	2014-09-04 23:43:42	delete	download
StructureAnalysis_20141010221102	StructureAnalysis	2014-10-10 22:11:04	2014-10-10 22:11:05	Completed	0	2014-10-10 22:11:43	delete	download
StructureAnalysis_20141010225308	StructureAnalysis	2014-10-10 22:53:09	2014-10-10 22:53:13	Completed	0	2014-10-10 22:53:49	delete	download
Hydrodynamics_20141011141008	Hydrodynamics	2014-10-11 14:10:09	2014-10-11 14:10:10	Completed	0	2014-10-11 14:10:18	delete	download
StructureAnalysis_20141012093535	StructureAnalysis	2014-10-12 09:35:36	2014-10-12 09:35:38	Task:Gridgen being executing on192.168	0		delete	download
Hydrodynamics_20141012093656	Hydrodynamics	2014-10-12 09:36:57	2014-10-12 09:36:58	Task:Gridgen being executing on192.168	0		delete	download
Hydrodynamics_20141012093929	Hydrodynamics	2014-10-12 09:39:30	2014-10-12 09:39:31	Scheduling :Gridgen	0		delete	download
StructureAnalysis_20141012095507	StructureAnalysis	2014-10-12 09:55:08	2014-10-12 09:55:13	Completed	0	2014-10-12 09:55:57	delete	download
ShapeDesign_20141012102621	ShapeDesign	2014-10-12 10:26:22	2014-10-12 10:26:23	Completed	0	2014-10-12 10:26:25	delete	download
ShapeDesign_20141012104348	ShapeDesign	2014-10-12 10:43:49	2014-10-12 10:43:50	Task:Gridgen being executing on192.168	0		delete	download
ShapeDesign_20141012150401	ShapeDesign	2014-10-12 15:04:02	2014-10-12 15:04:03	Scheduling :Gridgen	0		delete	download
Hydrodynamics_20141012150848	Hydrodynamics	2014-10-12 15:08:49	2014-10-12 15:08:50	Scheduling :Gridgen	0		delete	download
Hydrodynamics_20141012151101	Hydrodynamics	2014-10-12 15:11:02	2014-10-12 15:11:03	Completed	0	2014-10-12 15:11:08	delete	download
Hydrodynamics_20141012152245	Hydrodynamics	2014-10-12 15:22:46	2014-10-12 15:22:47	Completed	0	2014-10-12 15:22:51	delete	download
ShapeDesign_20141012152528	ShapeDesign	2014-10-12 15:25:29	2014-10-12 15:25:29	Task:Gridgen being executing on192.168	0		delete	download
StructureAnalysis_20150327133921	StructureAnalysis	2015-03-27 13:39:23	2015-03-27 13:39:24	Waitting	0		delete	download
StructureAnalysis_20150329175140	StructureAnalysis	2015-03-29 17:51:43	2015-03-29 17:51:43	Waitting	0		delete	download

Figure 8. Multiple Flows Monitoring

6. Conclusion

This paper presents a Flex-based visual flow design method for AUV using in the AUV collaborative design platform which is based on B/S mode. This method realized visual AUV flow design embedded in browser. The function structure of visual flow design system is introduced and the design of basic widgets and task widgets for flow modeling is described in detail. The procedure and implementation of flow monitoring are presented and an application example is given. Flex-based visual flow design system has visual interface and it is easy to operate so that it can improve the AUV design efficiency.

Acknowledgment

This work is supported by Natural Science Foundation of China under Grant No. 51209036 and the Fundamental Research Funds for the Central Universities (No. DUT15QY40).

References

[1] Z. Pang, F. Wen, X. W. Pan and C. Lu, “Migration model for rich internet applications based on PureMVC framework”, 2010 International Conference on Computer Design and Applications (ICCDA), **(2010)** June 25-27; Qinhuangdao, China, pp. V5-340-V5-343

[2] P. Juszkiewicz, B. Sakowicz, P. Mazur and A. Napieralski, “The use of Adobe Flex in combination with Java EE technology on the example of ticket booking system”, 11th International Conference on the Experience of Designing and Application of CAD Systems in Microelectronic, **(2011)** February 23-25; Polyana-Svalyava, Ukraine, pp. 317-320.

[3] X.Y. Li and X.F. Wang, “Research on interactive CAPTCHA mechanism based on RIA”, 2011 International Conference on Multimedia Technology(ICMT), **(2011)** July 26-28, Hangzhou, China, pp. 679-682.

[4] F.Y. Song, B.C. Jiang, Z.H. Yan, C.H. Liang and S.X. Li, “The application of workflow in prefecture-county integrated dispatching system collaborative modeling⋆”, Journal of Computational Information Systems. vol. 10, no. 3, **(2014)** February 1, pp. 1141-1148.

[5] H.G. Zhou and Y.P. Jia, “Modeling of marine diesel engines collaborative development workflow based on HTCPN”, Information Technology Journal, vol. 12, no. 22, **(2013)**.

[6] http://www.wfmc.org/

[7] P. Wohed, N. Russell, A.H.M. ter Hofstede, B. Andersson and W.M.P. van der Aalst, “Patterns-based evaluation of open source jBPM systems: The cases of jBPM, OpenWFE, and Enhydra Shark”, Information and Software Technology. vol. 51, no. 8, **(2009)** August, pp. 1187-1216.

[8] Y.C. Song, B.D. Wu and J.L. Chen, “The design and implementation of code generation based on J2EE in the development of JBPM workflow system”, Applied Mechanics and Materials, vol. 263-266, **(2012)** December, pp. 1961-1968.

[9] E. Pardede, J.W. Rahayu and D. Taniar, “XML data update management in XML-enabled database”, Journal of Computer and System Sciences, vol. 74, no. 2, **(2008)** March, pp. 170-195.

[10] S.C. Haw and C.S. Lee, “Efficient preprocesses for fast storage and query retrieval in native xml database”, IETE Technical Review, vol. 26, no. 1, **(2009)**, pp. 28-40.

Authors

Zhenzhen Xu, She received the Ph.D. degree from Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China in 2009. She is currently an Assistant Professor at School of Software in Dalian University of Technology. Her research interests include modeling, scheduling, simulation and optimization of complex system.

Jie Dai, She obtained the Bachelor’s degree from Dalian University of Technology in 2013. And she is studying for her Master’s degree at School of Software in Dalian University of Technology. Her major is software engineering and her interest area includes workflow scheduling and visual flow design and monitoring.

Xiaowei Zhao, She received the M.S. degree from Queen Mary and Westfield College in University of London in 2005. She is pursuing a doctorate in the Faculty of Management and Economics in Dalian University of Technology. She is currently an Assistant Professor at School of Software in Dalian University of Technology. Her research interests involve complex system and gaming theory.

Xiujuan Xu (Corresponding Author, xjxu@dlut.edu.cn), She received the Ph.D. degree in College of Computer Science and Technology from Jilin University, Changchun, China in 2008. She is currently an Assistant Professor at School of Software in Dalian University of Technology. Her research interests include problems in complex system especially in social network.