User-defined Business Process Based On Web Modeler

Yuanchun Xu[1, a], Lin Mei[1, b] and Zhengying Li[1, c]

[1]School of Wuhan University of Technology, Wuhan 430070, China;

[a]xuyuanchun@whut.edu.cn, [b]meilin@whut.edu.cn, [c]zhyli@whut.edu.cn

Keywords: User-defined business process, Encapsulation, Web modeler.

Abstract. A novel approach to achieving user-defined business process based on web modeler is proposed and experimentally demonstrated. The modeler which is based on AngularJS providing MVC design pattern and mainly composed by stencil set, canvas, gives us an opportunity to encapsulate the common elements of bpmn in process, or even sub processes. We can save the activities after modeling in the modeler and use them later with esay drag-drop. To achieve the encapsulation, we also need to sort the similar activities into same tasks, and then make up the differentiation by parameters while abstracting the process in reality, finally realize it in both backend and frontend. The web modeler responsible for encapsulation on frontend is the core module in this paper.

1. Introduction

Business process management (BPM), also named workflow, is based on the observation that each product or service that a company provides to the market is the outcome of a number of activities performed. Business processes are the key instrument to organize these activities and to improve the understanding of their interrelationships [1-2]. In the last decades, business process management has been a "hot topic" because it is highly relevant to a practical point of view. It has been wisely applied in numerous fields, such as industrial manufacturing and office automation.

With the development of workflow technology, a workflow reference model (fig.1) has been proposed by Workflow Management Coalition (WFMC) [3], in order to get a common design principle. From the model we know that we should establish the process definition first and then transform workflow engines to be workflow service. In fact, however, business requirement will not be changeless. Instead, we must react to the change of requirement quickly. So there are a lot of researches done to improve the flexibility and dynamics. Most researches focus on the modelling of business process, especially the modelling language [4-5], and some attention paid to the structure of system [6-7]. In general, the process definition is done by software developers, not the end users. So if there need some changes later, things will be difficult because of the lack of professional technology of end user and the expensive cost of redefinition by developer. So there is a serious problem about how to define and modify the business process easily.

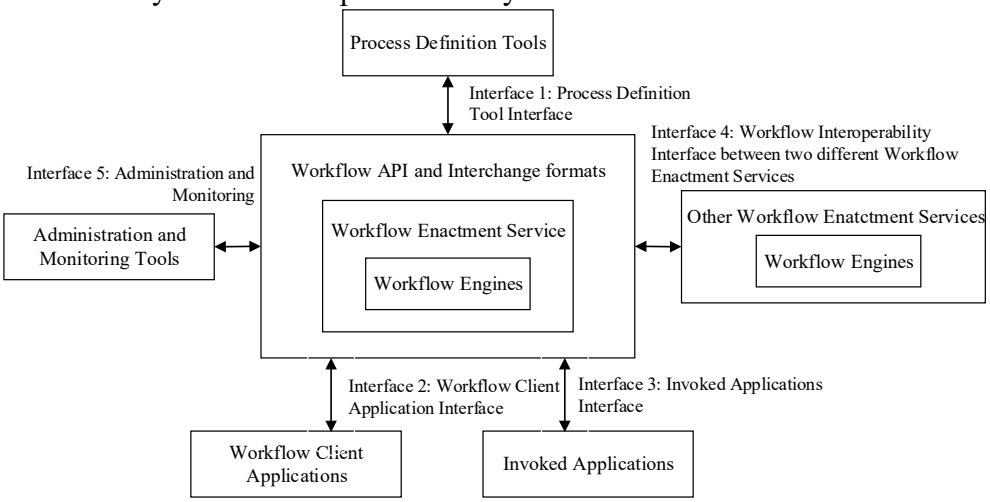

Fig.1 workflow reference model

In this paper, a user-defined business process management system based on web was investigated to solve this problem. The B-S infrastructure is essential for end user to define process without development environment. Many researches of flexible workflow are based on web technology [8]. The main component of the system is a modeler with which we can model the process easily. This modeler gives us an opportunity to encapsulate the common elements of bpmn (Business Process Model and Notation) in process, such as tasks or events, and we can use these elements to redefine process quickly without setting of numerous parameters. A practice of the system shows that the modeler works well and experiences good.

2. User-defined business process activities

2.1 Principle of user-defined business process activities

In office automation system, we often meet some business process like what fig.2 shows, which is a funds application process. Financial officers need to handle the application when someone report it, if the application is approved, then the applicant can receive the funds. But if the application is disapproved, the applicant must modify the application and reapply or just give up the application to finish the process. In fig.2, rounded rectangle represents for user task which is done by one user and diamond represents for exclusive gateway which owns two branches conditioned by the expression on the arrowed line. And the interfaces on the dotted line are invoked when someone want to complete related user task.

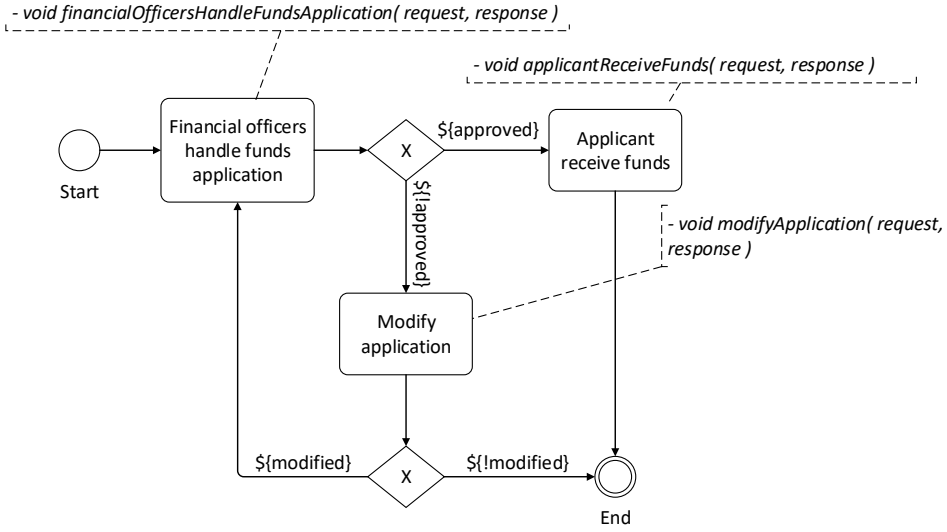

Fig.2 Funds application process

It's easy to accomplish this business process with most of workflow engines and relative technologies. Usually we just follow steps below:

1) Model the business process
2) Convert it to bpmn, a standard BPM vendors accepted
3) Realize the services related
4) Compile source code
5) Deploy application

But we almost need to accomplish these steps again in order to modify this model. Unfortunately, it's happened frequently. For example, department leader wants to handle the funds application first, so we need to change the process to what fig.3 shows. In fact, the two handling user tasks are similar just with difference in parameters. The first user task's operator should be a department leader and the second should be a financial officer. So we can encapsulate the user task making use of the similarity. According to the thought of object-oriented, we can treat the user task as an object of a class, with the different parameters as properties. This thought can also be applied to the design of service. The two user tasks can invoke the same service interface with different parameters. So we can simplify the first step and skip the third step because of the encapsulation when modify the model. Then the new

process model should look like fig.4. And the services invoked by the two different models are listed in table 1.

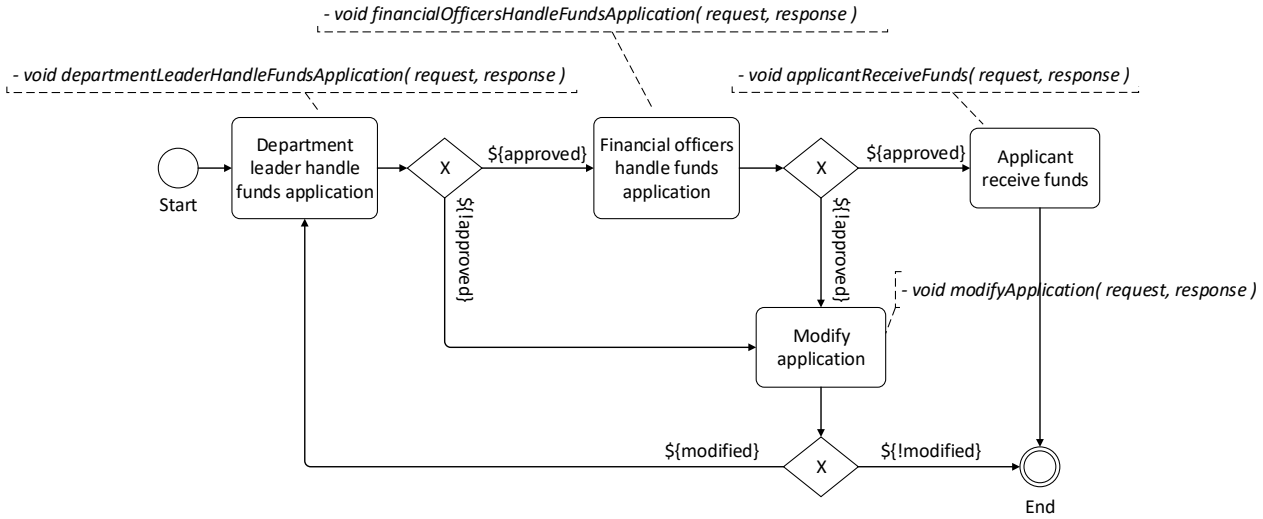

Fig.3 Modified funds application process

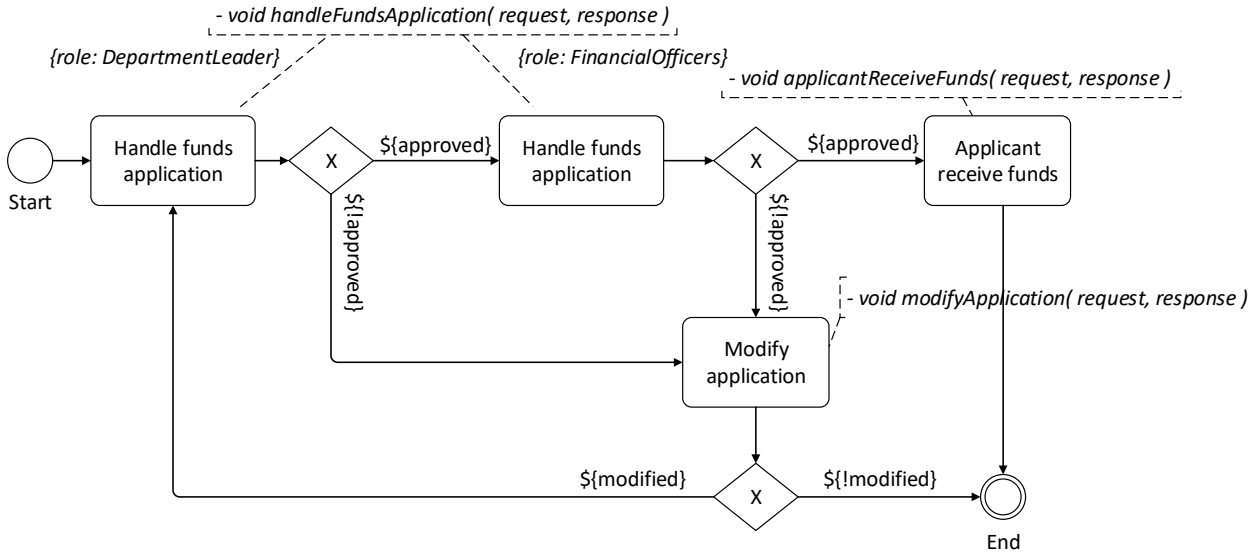

Fig.4 Modified funds application process with encapsulation

Table 1 Service invoked without encapsulation vs with encapsulation

Services	Invoked without encapsulation	Invoked with encapsulation
- void financialOfficersHandleFundsApplication(request, response)	*YES*	*NO*
- void departmentLeaderHandleFundsApplication(request, response)	*YES*	*NO*
- void handleFundsApplication(request, response)	*NO*	*YES*
- void applicantReceiveFunds(request, response)	*YES*	*YES*
- void modifyApplication(request, response)	*YES*	*YES*

In order to reduce the steps mostly to minimize cost, we can make the most of the web technology to automate the second step and skip the fourth step. We have designed a modeler in the web showed in chapter 2.2. Due to the web modeler, end user need not setup the operational environment, only a browser needed.

2.2 Realization of the user-defined business process

We build the modeler taking advantage of the excellent AngularJS framework and with the popular Bootstrap framework as the UI framework. AngularJS provides the MVC pattern for solving the complexity and Bootstrap contributes to a good user experience. Fig.5 shows that the modeler is the core component of user-defined business process system.

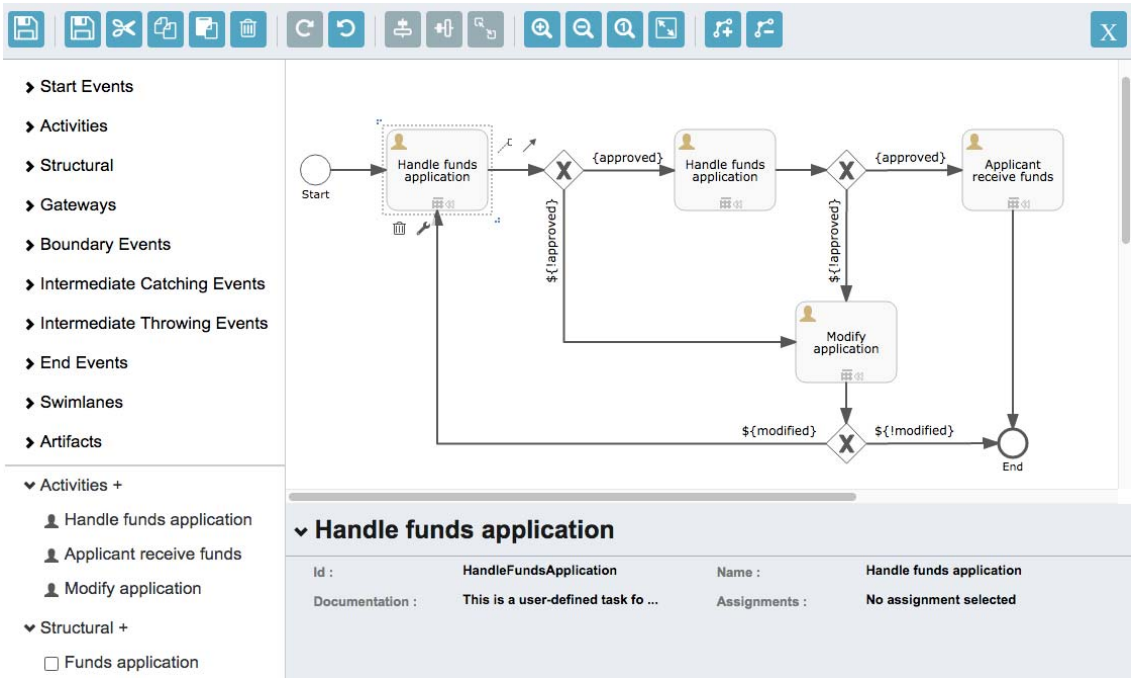

Fig.5 the modeler we realized

There is an example in the modeler we introduced before in chapter 2.1. We encapsulate the user tasks we used in the example, so we can pick them up easily to make up the model or modify it. With the modeler, we can encapsulate new bpmn's elements by clicking the second save button when we select the element on the canvas after setting properties. It's clear that the properties setting area owns only several necessary properties of the user task named "Handle funds application" because of encapsulation. When we accomplish a model, we can save it to reuse later or deploy immediately.

The core of user-defined process is the encapsulation, both backend and frontend. The backend provides services and saves encapsulated object, while the frontend provides the modeler. We could use any backend technology to realize the service in principle, but we choose java web here because of a large number of mature framework including "activiti" which is the workflow engine we used. As for front end, we mainly use AngularJS as the MVC framework. The structure of the core system showed in fig.6. The backend contains several layers we often used with base data stored in database. The frontend mainly consists of three classes, in which model is built with stencil as data and canvas as view. We get stencils and current model from backend with data interaction in JSON format, and then canvas will render it with rule checking. If we want to save the model, backend will receive data too and store in the database.

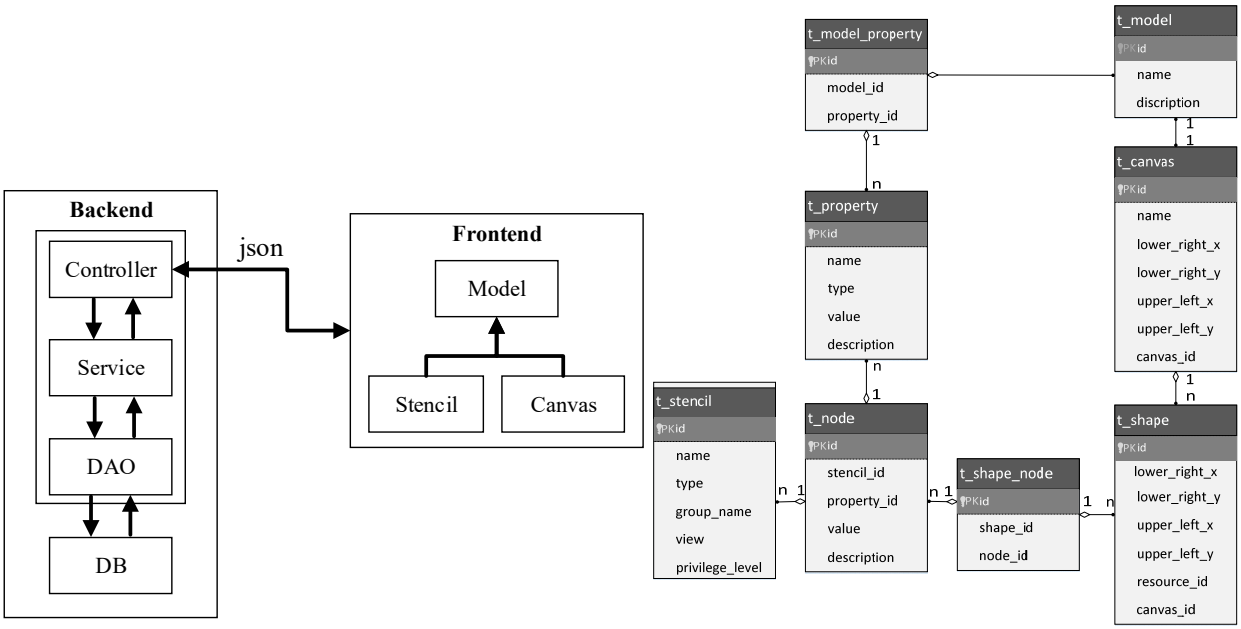

Fig.6 Core system structure (left), E-R model of frontend's objects (right)

According to the E-R model which contains main properties of objects, we can figure out some relationships of the front objects. The table named t_node combines t_property and t_stencil to instance stencil as the base data, and the canvas consists of a lot of shapes recursively with plenty of position information. Relationship between t_shape and t_node is related by the table named t_shape_node. The type property of stencil distinguished the standard stencils and user-defined stencils, Besides, the group property is for grouping and the view property is for SVG display. We add privilege_level to stencil considering to safety that end users are only available to user-defined and basic stencils. All the basic information made up the final model.

3. The User-defined Workflow system and result

3.1 The User-defined Workflow system

According to the workflow reference model as fig.1 shows, we need monitoring tools to compose the basic system. So our system structure should look like what fig.7 shows. There are three main parts of the system: tasks, processes and management. Tasks module deals with user tasks in which current user involved. The module contains three sub modules too. Inbox submodule sorts out the user tasks we should handle. The user tasks we involved in but unfinished are sorted into submodule named involved while the finished sorted into history. In processes module, we can start a process and then monitor the instance with view displayed. Otherwise, the module will be the entry of modeler. As same as most systems, there is a management module. Besides managing users and groups, we can also active process or suspend it.

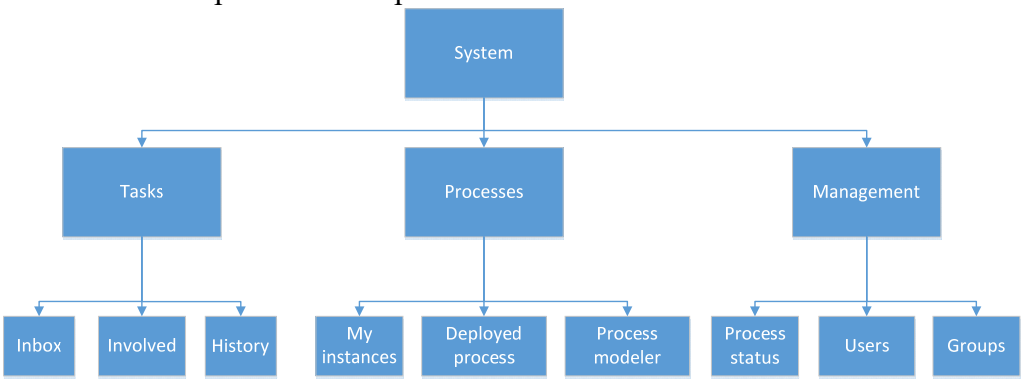

Fig.7 System structure

3.2 Test process and result

We test our system with two business processes. The funds application process we introduced before is the first process and which serves as the sub process of the second one which is a business travel process. In the business travel process, we start the process by reporting a business travel application, like the funds application process, then the department leader and manager handle the application one by one. If it's approved, applicant can call the funds application process to finish process. What is described before is showed in fig.8 where funds application serves as the sub process.

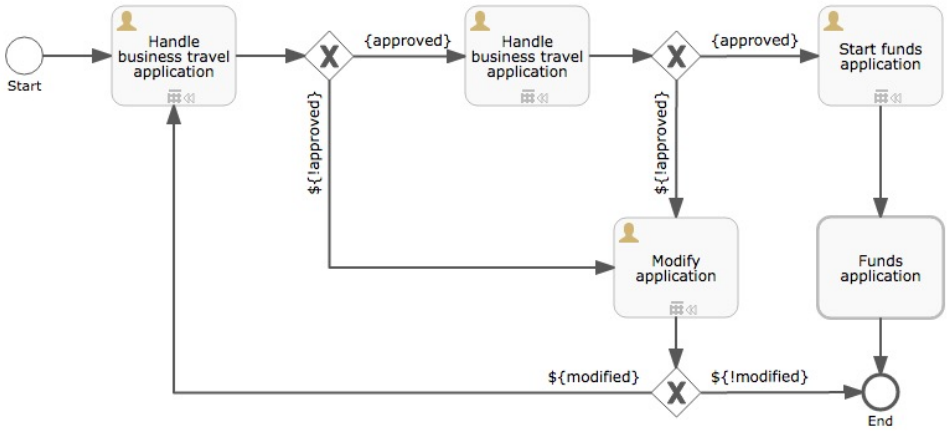

Fig.8 really business travel application process

We have designed four users with different roles in order to complete the test and complete task step by step. We invoked the business travel process first and then called the sub process until we finished the whole process. Each user completes a task with parameters, so the system runs differently with the same tasks of different users according to the parameters. Every step runs well in our system during the testing.

4. Summary

In this paper, a novel approach to achieving user-defined business process based on web modeler is proposed and experimentally demonstrated. The modeler which is based on AngularJS providing MVC design pattern and mainly composed by stencil set, canvas, gives us an opportunity to encapsulate the common elements of bpmn in process, or even sub processes. We can save the activities after modeling in the modeler and use them later with esay drag-drop. To achieve the encapsulation, we also need to sort the similar activities into same tasks, and then make up the differentiation by parameters while abstracting the process in reality, finally realize it in both backend and frontend. Moreover, we realized the system with managing and monitoring tools in order to achieve user-defined business process. Finally, a business travel process containing a funds application process demonstrated shows that our system works well.

Acknowledgments

The authors would like to acknowledge funding support from the National Science and Technology Pillar Program (grant no. 61575149).

References

[1]. Mathias Weske. Business Process Management Concepts Languages Architectures [M]. Springer-Verlag Berlin Heidelberg. 2007. p.4-5.

[2]. Rosemann M, vom Brocke J. The six core elements of business process management [M]. Handbook on Business Process Management 1. Springer Berlin Heidelberg, 2015. p. 105-122.

[3]. Information on: https://en.wikipedia.org/wiki/Workflow_Reference_Model

[4]. M. Pesic and W.M.P. van der Aalst. A declarative approach for flexible business processes management [C]. International Conference on Business Process Management. Springer Berlin Heidelberg, 2006. p. 169-180.

[5]. Ralf Mueller. Enterprise applications of semantic technologies for business process management [J]. Journal of Zhejiang University Science C. 2012. p. 308-310

[6]. Wu Shao-fei. Service-based flexible workflow system for virtual enterprise [J]. Journal of Chongqing University (English Edition). 2008, 1.

[7]. Wang S, Zhang Y, Peng Y. Flexible Mechanism Research of Workflow System Based on SOA[C]. International Conference on Trustworthy Computing and Services. Springer Berlin Heidelberg, 2013. p. 86-92.

[8]. Wang, Y. J. Research and application in emergency system of flexible BPMN workflow engine based on WEB [D]. Hefei, South China University of Technology, 2011.