Application-level Web Component Framework for Distributed Workflow Management

Yinzhou Zhu[1] and Baolin Yin[2]

[1,2] National Laboratory Software Development Environment,
Beihang University, Beijing 100191, China
{zhuyinzhou, yinbaolin}@nlsde.buaa.edu.cn

Abstract

Distributed workflow management system has been widely used for the requirement of rapid development of process oriented software systems. In the past few years, more and more modern enterprises prefer the web-based workflow solutions. However, developing a web-based distributed workflow is a time-consuming task, since the architecture is complex and the programmers need to master the skills of both front-end and back-end development. In this paper, we present an application-level web component framework to develop the web-based workflow efficiently in the distributed environment. In this framework, the functions of the components are restricted to human-computer interaction and data computation, and all other functions are integrated into the workflow management system. Most web applications can be easily created through the component configuration instead of coding. Real-world scenarios which are built and implemented based on this architecture are shown to prove the effectiveness and usefulness of the framework.

Keywords: *Distributed Workflow Management, Business process, Component-based development, Web application*

1. Introduction

Workflow management has received considerable attention in recent years due to its potential for significantly increasing productivity and saving costs. More and more large enterprises consolidate their project implementation into a workflow management system [1]. With the development and popularization of Internet, large enterprises often require the workflow management system provide a web-based workflow solution in the distributed environment, since it could cut the costs of upgrades, maintenance and software deployment. Distributed workflow management systems (DWFMS) are often adopted under such conditions. However, traditional method of developing workflow web applications is a time-consuming task. Not only because of its complex architecture, but because the business logic is coupled with the data model too closely so that the applications are not able to be widely reused in many different business fields. Therefore it is necessary to find out a set of web application development patterns which are compatible with the distributed workflow.

In the past decade, workflow management technology has played an important role in the fields of business process management and scientific computation [2, 3]. Distributed workflow management is focused mainly in scientific computation environments [3]. However, with the advent of rapid evolution of the business process in large enterprises, the DWFMS are increasingly used in business process management field these days. Muthusamy et al developed a flexible and distributed platform to develop, execute, and monitor business process [4, 5]. The distributed architecture of this platform focus on the optimization of the computational resources which are based on web services and a distributed content-based publish/subscribe overlay which could work effectively in heterogeneous environments. Khalaf and Leymann present a BPEL fragmentation covering data and explicit control dependencies, and an approach to handle fragmenting loops and scopes [6]. The emphasis of this distributed model is on the optimization of the control flow. Hamann *et al.*, present a migration data meta-model for business processes with the ability of runtime migration [7]. This distributed model is oriented to improve the flexibility of the ad-hoc workflow, and the prototype is applied to WS-BPEL and XPDL processes.

In above recent research, the applications of the workflow are developed mainly based on web service, and the process of the development needs close co-operation between business process analysts and programmers [8]. On the other hand, most of common business process management systems, such as SAP, Oracle, Staffware, WebSphere, FLOWer, COSA [1, 2, 9, 10], provide the module library to composite the workflow. However, most of these components are designed for the tasks related to the specific business area, and users are required to have technical expertise, which make these components hard to adapt in general conditions.

Along the lines presented in this paper, we propose a web component framework for distributed workflow management, and implement it on the EasyWork system. Our work distinguishes itself from above approaches by using the application-level components [11, 12] and restricting the functions of the components to human-computer interaction and data computation, which means all other application functions are integrated into the workflow management system, such as the web infrastructure of the front-end and the back-end, business logic controlling, network communication, database accessing, and so on. If the functions of the components could be more concentrated, the process of application development would be more efficient and easier. To achieve this target, the general functions of the workflow web application should be abstracted, and relevant mechanism should be provided by DWFMS. Besides, the process of the workflow web application development should be improved to minimize the need of the additional coding.

The EasyWork system, on which the web component framework is implemented, is a DWFMS developed by our group. It is based on the persistent messaging mechanism, decentralized distributed workflow model, and the universal data bus mechanism. Compared to other DWFMSs, EasyWork system is more flexible in controlling data stream between the workflow engine and applications, which is helpful to restrict the functions of the components.

This paper focuses on the framework model and the runtime mechanism of the workflow web applications. And a lot of real-world workflow scenarios developed based on EasyWork system is discussed to prove the effectiveness and usefulness of the framework.

2. EasyWork System

The architecture of the EasyWork system, which is shown in Figure 1, aims to define, execute and monitor the workflows for the cross-regional enterprises which often have several highly autonomous subsidiaries. The network architecture of the EasyWork system is a hybrid structure, which is based on the peer to peer network and the client/server framework. There are two kinds of nodes in the EasyWrok network: EasyWork Server node and the EasyWork Client node. The EasyWork Servers are the basic nodes which are used to store and dispatch the distributed workflow instances. The relationships between EasyWork Servers are symmetrical, while the EasyWork Clients are client nodes of the EasyWork Server. The EasyWork Platform is installed on every EasyWork Server, and offer the access interface that allows users to get and do their job in remote EasyWork Clients though a standard web browser.

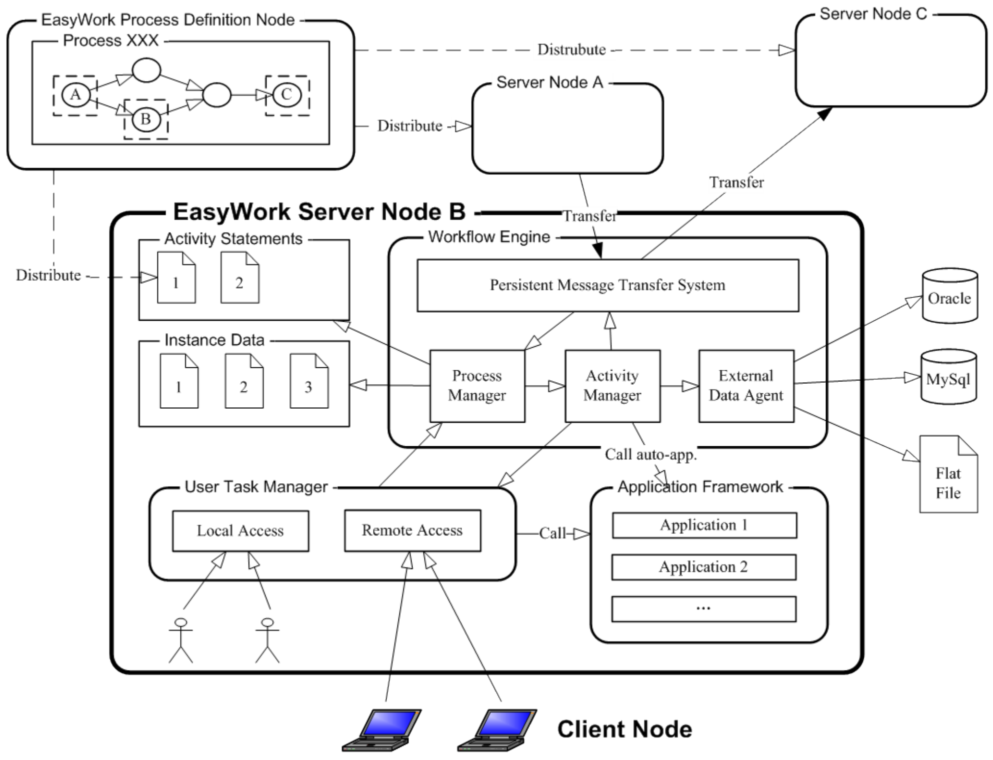


Figure 1. The Architecture Diagram of the EasyWork System

The EasyWork Platform on EasyWork Server is composed of three main parts: the workflow engine, the user task manager and the application framework. The workflow engine is used to receive, store, and dispatch the workflow instances, and invoke the application to process the tasks of the activities. The user task manager offers the access interface of the workflow system to users, shows the task-lists, and communicates with the workflow engine to process user commands. The application framework is a set of applications which are created based on the components with application-level granularity. These applications, which are invoked by the workflow engine to process the tasks of the activities, are executable programs, such as the executable binary files of the operational systems of Microsoft Windows or UNIX, or the web pages that could be interpreted by http server or client browser.

EasyWork Process Definition Server is a kind of EasyWork Server on which the workflow definition tool is installed. The workflow definition tool is used by workflow administrators to define, compile and deploy the distributed workflow. After a workflow model has been designed in the Process Definition Server, it will be split into several segments by activities, and be compiled to configuration files. These configuration files will then be distributed to EasyWork Servers which are defined as the computing resources of the activities.

3. Application-level Web Component

The applications in EasyWork system are developed based on the application-level components. Compared to the traditional function-level components, the application-level components are with larger granularity, and could be directly executed as applications [11, 12]. Compared to the function modules in professional business management system, such as the module library in SAP R3, the application-level components are with smaller granularity, and could be used in more general conditions. Application-level components do not involve complex business logic. But through the component configuration, the functionality of the application-level component could be widely extended. Further more, the application-level components are even able to combine together in the "Work-item List" in the EasyWork system to implement the workflow with complex business logic.

In the EasyWork system, applications are not allowed to access any data sources external to the workflow directly. Instead, the workflow engine is in charge of providing the input data for the components and maintaining the output data from the components. The benefits of this approach is that the designer of the components need not to concern the low-level data operation any more, such as database access and network transmissions, and the functions of the component are more concentrated than before. Based on the past experience of workflow development, the components in EasyWork system are grouped into two classes: the UI Components, which concentrate on the functions of user interface and human-computer interactions; and the Computational Components, which are used to compute based on the input data.

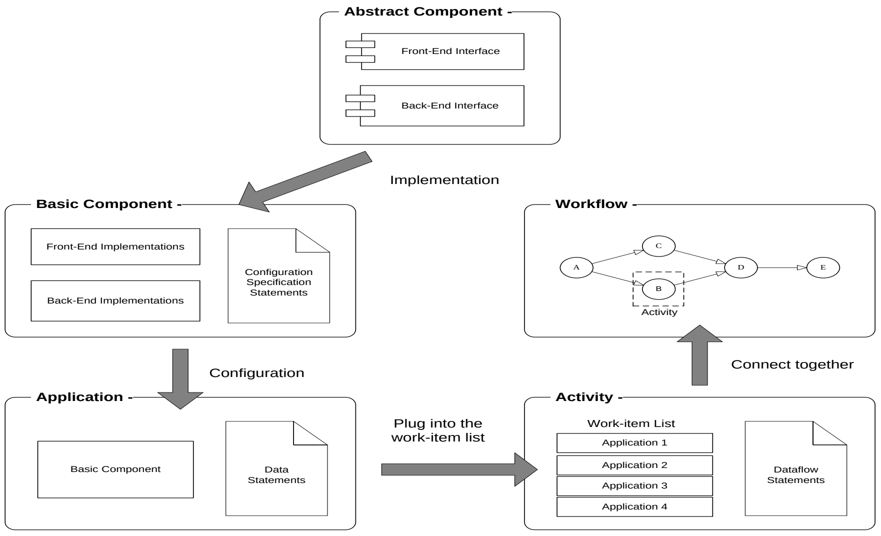


Figure 2. The Workflow Application Development Process in the EasyWork System

The application-level web components are used to construct the web applications in the EasyWork system. The process of the web application development can be seen in Figure 2. Each web component consists of three parts: the front-end, the back-end, and the Configuration Specification Statements file. The front-end and the back-end of the web component are the programs, which are implemented based on the predetermined web component interface, and can be executed on the server side and web browsers respectively. The Configuration Specification Statements file specifies the configurable variables of the component, including the variable name, usage, I/O type, and data model constraints. Through this file, the component configuration tool can guide the developers to configure the component, and create the Application Data Statements file, which specifies the fixed data models and values of the component variables, the list of input and output variables, and the usage of the component. With the Application Data Statements file, the web component has already become a web application that can be plugged into the Work-item List of workflow activities. When plugging a configured component into a Work-item List, the developer should set the execution order of the application and configure the mapping relation between workflow data and component data. The Work-item List supports the basic control flow management, including sequence flow, condition flow and loop flow.

4. Architecture

The EasyWork network is composed of EasyWork Servers on which the EasyWork system is installed. The EasyWork Servers are the basic nodes which are used to store and dispatch the distributed workflow instances. The relationships between all EasyWork Servers are symmetrical. When a server finishes the tasks of an activity of a workflow instance, the workflow instance will be transferred to another server where the next activity resides.

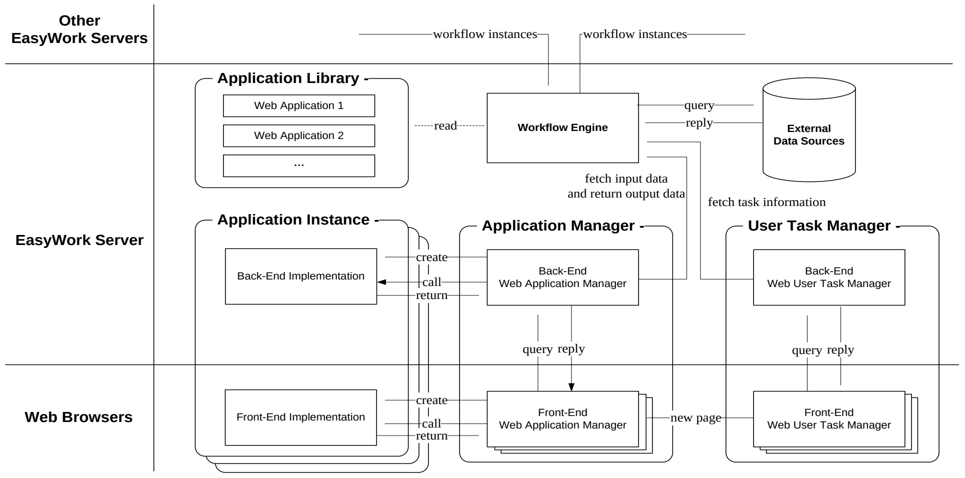


Figure 3. The Runtime Architecture of the Web Applications in the EasyWork System

As shown in Figure 3, EasyWork system helps the users manage their tasks through the User Task Manager and Application Manager which are installed on every EasyWork Server. The detailed steps of executing a web application in EasyWork system are as follows:

(a) First, the user accesses the User Task Manager through a web browser, get the task list, and select the task which the user hopes to do.

(b) When the user starts the task in the front-end of the User Task Manager, a new web page will be opened and will automatically access the Application Manager to process the task.

(c) The Application Manager then sends a query that contains the task ID to workflow engine to find where is the first application in this task and its input data.

(d) The workflow engine gets the address of the application from the Application Library, and prepares the input data of the application based on the configuration of the Work-item List of this task. If some of the input data is stored in the external data sources, the workflow engine will fetch the data though the relevant interface.

(e) The Application Manager loads the code of the application into the back-end and front-end containers respectively to create the application instance. And set the input data of the application instance.

(f) For the applications developed based on UI Component, most of the operations are often executed on the front-end. But for some specific functions, such as sending emails or doing a secret computation, the operations can only be executed on the back-end. In this situation, the front-end application can call the functions contained in back-end application though the interface provided by the Application Manager.

(g) For the applications developed based on Computational Component, since there is no front-end in the application, the back-end of the application will be executed automatically without human intervention.

(h) When the user finishes the current application, the Application Manager gets the output data, close the application instance and then sends the output data to the workflow engine.

(i) The workflow engine maps the output data to the flow data or external data source based on the configuration of the Work-item List of this task. If not reach the end of the Work-Item List, the workflow engine will prepare the input data of the next application in the Work-item List, then return the address of the next application, and back to the step (e).

(j) The Application Manager informs the user and refreshes the front-end of the User Task Manger.

5. Development Framework

The EasyWork system provides a web application development framework to help developers create, configure, and combine the application-level web components. The structure of the framework is shown in Figure 4.

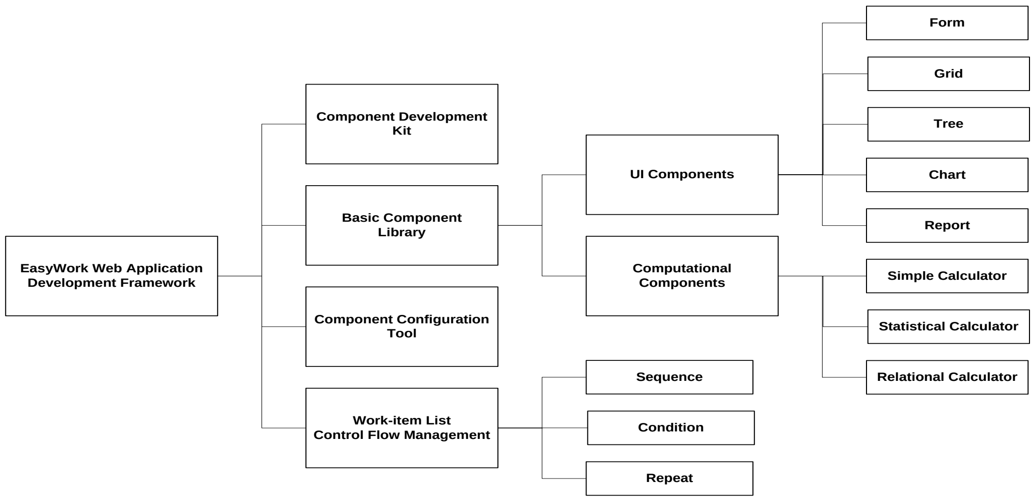


Figure 4. The EasyWork Web Application Development Framework

The framework mainly consists of four parts:

(1) Component Development Kit (CDK), which is used to develop low-level basic components. When the existing components can't meet the business requirement or the Work-item List is too complex, the developers could use CDK to develop the specific components to enhance the workflow functionality. The CDK provides the abstract interfaces

of the component, the development templates, the extending specification and the guidelines for the component development and test.

(2) Basic Component Library, which consists of the standard basic components of the EasyWork system. These components are developed according to the classic web application patterns which are summarized from the past experience of workflow development to meet the general requirements of the business application. These components can be grouped into two classes. The UI Component provides various options to customize a web application, such as the layout of the page, the data model of the view, the operation mode, the button functions, and so on. The Computational Components offers the computational functions based on various data model, such as the simple data, the list, or even the relational table which is very useful in the conversion of the data schema.

(3) Component Configuration Tool, which is a visual tool to configure the application-level component. This tool can parse the configuration specification statements file of the component and produce the graphical configuration options to the developers in order to decrease the difficulty in the generation of the application data statements file.

(4) Work-item List Control Flow Management (WLCFM), which is a mechanism used to combine the multiple applications in order to implement more complex business logic. The control flow supported by this mechanism is mainly based on the sequence construct, and support simple conditional branch and loop. From our past experience of workflow development, most of the application in business process can be presented in a work-item sequence with a little simple control flow management, because these applications often focus on the logic of the functions instead of the user-friendly but complex operations. With the help of this mechanism, most of the complex business logic could be implemented based on the basic components, although the mechanism may decrease the convenience of the operation at the same time.

6. Scenarios

To illustrate the effectiveness of the application-level web component framework, the real world scenarios which implemented based on EasyWork system are discussed in this section. As shown in Table 1, the scenarios are grouped into three classes:

(1) Office automation (OA) applications, which are the collaboration systems based on the workflow. The OA system is a type of classic workflow system, and plays an important role in paperless office. The OA systems are widely used almost in all fields without the limit of the industry or geographic region. The key element of the OA is the form, which often is easy to be customized based on the web components in the EasyWork system.

(2) Enterprise resource planning (ERP) system, which is a more integrated platform, compared to OA systems, used to manage various kinds of information in the enterprise. From the model layer, such as the data model and business logic, to the view layer, such as the view of the decision support, all kinds of the information in the enterprise are integrated together into a strict architecture. The design of the data model and process in ERP system are more professional than other information management systems, and are more difficult to be implemented.

(3) Report generation systems, which are the process-oriented systems used to collect and summarize the information which is distributed in different areas, and generate the Summary Report finally. This kind of application is very common in different areas, and is not easy to implement for the complex computation and distributed architecture. The real-

world report generation scenario implemented based on EasyWork system is the Freshwater Quality Monitoring (FQM) which is a process required by the National Oceanic Administration of China to investigate the freshwater quality in the coastal areas and provide a summary report every year.

Table 1. Function List of the Real World Scenarios Implemented by EasyWork System

Class	Subclass	Description	Number of Function Points
OA	Document Circular	The workflows for the documents authorization, incoming documents transaction, documents issuance	3
	Administrative Approval	The applications for the network, bus, conference room, and so on	6
	Personnel Management	The workflows for recruitment, resign, day off, over-time work, and so on	9
	Financial Management	The workflows for reimbursement, requisition, resource redeployment, and so on	3
	Product Management	The workflows for price fixing, quality control, and so on	6
	Customer Service	The workflows for price inquiry, maintenance services, customer complaints and product return	5
ERP	Data Maintenance	Maintain the data model of GL, materials, warehouses, factories, customers and suppliers	36
	Business Processes	The process for sales, purchases, banks payments, transportation, and production	26
	Report Query	Various types of reports for decision support	36
Report	Summary Report	The process of investigation form issuance, collection, summary and report generation	10

The utilization rate of the web components when implementing the above scenarios can be seen in Table 1. It shows that the Form component and Grid component are the most commonly used components that can cover over 60% function points of an application. And the Report component, which supports the rich text edit, is also a frequently used component that contributes near 10% function points. There are less than 3% UI function points, which are some professional configuration user interface, and 18% computational function points, which are some special computation methods, such as data format conversion and progressive rates computation, can not be implemented based on existing basic web components. These functions are implemented by extending the existing components with the help of the CDK.

The control flow management mechanism provided by EasyWork system could support the most of the applications in above scenarios. And the applications that need the complex control flow, such as the pages with the hyperlinks, are also easily converted into the control flow that our mechanism supported.

Table 2. The Function Points Coverage Rate of the Web Components in the Scenarios

		OA	ERP – Data Maintenance	ERP - Business Processes	ERP - Report Query	Summary Report	Average
Basic UI Components	Form	40.00%	16.46%	11.85%	25.42%	35.71%	25.89%
	Grid	17.50%	69.62%	32.59%	30.51%	35.71%	37.19%
	Tree	/	3.80%	/	/	/	0.76%
	Chart	2.50%	1.27%	/	13.56%	7.14%	4.89%
	Report	/	/	5.19%	30.51%	7.14%	8.57%
Basic Computational Components	Simple Calculator	21.25%	5.06%	14.07%	/	/	8.08%
	Statistical Calculator	13.75%	1.27%	13.33%	/	/	5.67%
	Relational Calculator	3.75%	1.27%	13.33%	/	/	3.67%
Other UI		/	/	/	/	10.71%	2.14%
Other Computation		1.25%	1.27%	9.63%	/	3.57%	3.14%

In the process of the web application development in above scenarios, about 95% amount of the work are assigned to the people who just master the basic knowledge about the database management. And the average workflow development cycles shrink by up to 70%, compared to the original development process. These facts empirically prove that the web component framework reduces the requirements of the technical expertise, and significantly increases the efficiency of the distributed workflow development.

7. Conclusions

In this research, we propose an application-level web component framework to improve efficiency of the web application development in distributed workflow management. Compared to traditional DWFMSs, our components are designed in application level and the functions of our components are more concentrated. The main benefits of this approach are that it simplifies the process of the web application development in the DWFMS, and the components in the framework are able to adapt in general conditions. The idea of this architecture is tested in a lot of real world scenarios, and can be applied to the development of other DWFMSs. In the future, the framework will provide the components with more specific functions and focus on the communication patterns between the application-level components.

References

[1] J. Sinur and J. B. Hill, "Magic Quadrant for business process management suites", Gartner RAS Core research note, **(2010)**, pp. 1-24.

[2] W. M. van der Aalst, "Business Process Management: A Comprehensive Survey", ISRN Software Engineering, **(2013)**.

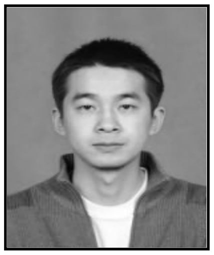
[3] E. Deelman, D. Gannon, M. Shields and I. Taylor, "Workflows and e-Science: An overview of workflow system features and capabilities", Future Generation Computer Systems, vol. 25, no. 5, pp. 528-540, **(2009)**.

[4] V. Muthusamy, "Flexible Distributed Business Process Management", University of Toronto, **(2011)**.

[5] S. Ganesan, Y. Yoon and H.-A. Jacobsen, "NIÑOS take five: the management infrastructure for distributed event-driven workflows", Proceedings of the 5th ACM international conference on Distributed event-based system, DEBS, **(2011)**.

[6] R. Khalaf and F. Leymann, "Coordination for fragmented loops and scopes in a distributed business process", Information Systems, vol. 37, no. 6, **(2012)**, pp. 593-610.
[7] K. Hamann, S. Zaplata and W. Lamersdorf, "Process instance migration: Flexible execution of distributed business processes", Software Services and Systems Research-Results and Challenges (S-Cube), Workshop on European, IEEE, **(2012)**.
[8] N. Russell, A. ter Hofstede, D. Edmond and W. van der Aalst, "Workflow data patterns: Identification, representation and tool support", Conceptual Modeling–ER 2005, **(2005)**, pp. 353-368.
[9] N. Russell, W. M. van der Aalst, A. H. ter Hofstede and D. Edmond, "Workflow resource patterns: Identification, representation and tool support", Advanced Information Systems Engineering, Springer, **(2005)**.
[10] J. Guojie, Y. Baolin and Z. Qiyang, "Enhancing Software Reuse through Application-level Component Approach", Journal of Software, vol. 6, no. 3, **(2011)**, pp. 374-385.
[11] G.-J. Jin, B.-L. Yin and Q.-Y. Zhao, "Reusability enhancing method for integratable software components with application-level granularity", Computer Integrated Manufacturing Systems, vol. 17, no. 2, **(2011)**, pp. 425-432.

Authors



Yinzhou Zhu received his M.S. in 2006 and currently is a Ph.D. candidate, both in Computer Science and Engineering Department at Beihang University (Beijing). His research interests include workflow management, enterprise application integration and distributed systems.

Baolin Yin is a Professor in Computer Science and Engineering Department at Beihang University (Beijing), and is currently head of the State Key Laboratory of Advanced Software Development. He received his Ph.D. in Artificial Intelligence in the University of Edinburgh (1984).His research activity is concerned with distributed systems, workflow management, image identification, and computer security.