International Journal of u- and e- Service, Science and Technology
Vol.8, No. 6 (2015), pp.257-266
http://dx.doi.org/10.14257/ijunesst.2015.8.6.25

Research of the Combination of Distributed Business Processes Based on Dynamic Planning

Yuan Gang, Sun Rui-zhi and Shi Yin-xue

Key laboratory of Agricultural information acquisition technology (Beijing), Ministry of Agriculture P.R.China China Agricultural University, Beijing 100083
yuan7_7@163.com

Abstract

In order to achieve the collaborative completion of entire business process by composition between different workflow management systems, a process composition method based on dynamic programming has been put forward. By analyzing differences and relations between process composition and Web services composition, many aspects have been analyzed with regard to service interface of the workflow system, the interaction control between the workflow systems, parameters and message's delivery and the planning indexes according to the characteristics of process interaction among the distributed workflow systems. When the process is running, according to the indicators and expectations of users, the optimal workflow services and sub-processes would be selected through dynamic planning method, to provide support to the dynamic combination of workflow processes in a distributed environment.

Keywords: *Process Combination, Dynamic Planning, Distributed, Web Service*

1. Introduction

At present, workflow management system had been widely used among enterprises' information systems as a business process management middleware. Due to the universal application of workflow management system, there have been all kinds of workflow products which are different in workflow model, process description language and system's functions. On the other hand, the modern enterprises' business processes are presented cross-regional, cross-sectoral and even cross different companies. Therefore, the implementation of a business process will interact and collaborate between different workflow management systems of different companies or departments, and the processes should be executed collaboratively in a heterogeneous environment.

2. Related Work

The combination of cross-system workflow business processes has been fully considered when the workflow management system was proposed. The Workflow Management Coalition (WFMC) defined the workflow engine could communicate with others collaboratively, and the interface 4 of its reference model was used for different workflow systems or engines' interaction [2]. Fernando [4] established interconnection between heterogeneous workflow systems by translating all processes to a formalized intermediary language IWRL. He designated a specific service while modeling, but this approach can't be well changed for the situation of a service when it becomes abnormal in the runtime. With the proposal of Web Service and SOA's concept and the realization of its techniques, the framework of business processes' integration and workflow systems' interconnection become a major technical means. Yang [3] proposed to package all the sub-workflow systems of the distributed workflow system into web services, which can provide services to others. Pavlin [5] established communication

ISSN: 2005-4246 IJUNESST

between the encapsulated process services according to the domain knowledge of the various heterogeneous workflow systems. His work was based on Dynamic Process Integration Framework (DPIF). Zhang [1] proposed service-oriented collaboration protocols and tool which include a collaboration ontology associated with a set of composable collaboration primitives and patterns, as well as concurrent control mechanisms to support collaborative workflow composition. Terstyanszky [6] presented the Coarse-Grained Interoperability concept which used meta-workflow concept that was described in the workflow language of the host native workflow system to enable combination and execution of workflows of different workflow systems. And the CGI approach wrapped non-native workflows as legacy applications to enable their execution using a formal description. Alqaoud [7] used Scientific Workflow Integration Framework (PS-SWIF) to establish interoperability between different workflow systems based web services' notification messaging system when the processes were executed.

In order to facilitate distributed implementation of the service, the models and methods for the combination of various services have been proposed. Xiao [8] expanded the functions, probability and cost information of the services into the process algebra to drive the web services' composition by Qos. Jing [11] presented a cloud manufacturing service composition method considering execution reliability based on discrete particle swarm optimization algorithm. Xie [14] proposed an algorithm to turn the service optimization and composition into service workflow templates which modeled with Petri net. The services' composition can also be seen as a planning problem, which means that it can generate Web service's composition programs and link the implementation of Web services dynamically. Especially based on AI planning, the web services can be combined semi-automatically or automatically according to a certain method. Fang[9] combined the functional and non-functional requirements of services automatically by HTN planning method and ontology, but the semantic information of each Web service was not well enough equipped to be solved before the services' combination. Song [12] studied the automation of services' combination through ontology and rule-based approach, but how to accomplish an overall goal collaboratively between the services was not well represented. To adapt to achieve the objectives of planning between each service in a heterogeneous environment, Falou [10] used service planning agent for each service. If the number and state of services change, it will bring more cumbersome. Under the premise of a clear overall objective, the various processes could be executed collaboratively in a distributed workflow systems.

This paper was also based on services' combination to achieve distributed execution of business processes and packaged the process as service, while taking advantage of the dynamic programming method to select the optimal execution of a process between different workflow engines.

3. Combination of Distributed Business Processes Based On Dynamic Planning

The processes or sub-processes are packaged as services to achieve a full implementation of the process by methods of combination. The implementation is similar to the service combination, but this manner of service's invoking and a combination is somewhat different from general service, mainly in: (1) Service is a corresponding sub-process or process fragment which is an instance that should be performed by its own workflow engine. So we should consider that off-site workflow engine can start and run a process fragment. (2) Since there is a close relationship between the data before and after the events in the workflow process, when we call a process fragment, some parameters need to be passed to these process fragments which are actually passed between the

different workflow systems. (3) The sub-processes or activities of a workflow process should always be executed in parallel, if the fragments on different engines, at the end of the treatment process, the results and the service requester waiting for a response need to be considered by the synchronous or asynchronous control strategy. (4) In order to achieve the optimum combination of real-time process fragments, it also need to consider the discovery and management of the processes in different workflow systems, so we can catch the processes which are available in the runtime.

3.1. Distributed Workflow Combination Framework

SOA architecture provides a description, discovery and invocation mechanism for the web services in different locations. When the workflow system function is provided as a service to offer or be invoked, its structure is similar to the SOA, but the workflow service is provided by a workflow system which offers just a process fragment of the whole process. If there is more than one workflow system service to execute a total process, we need to consider the processes' publishing, services' discovery, processes' implementation and the interaction with the users and so on. The design of the framework which supports the heterogeneous combination of process planning is shown in Figure 1 as follows:

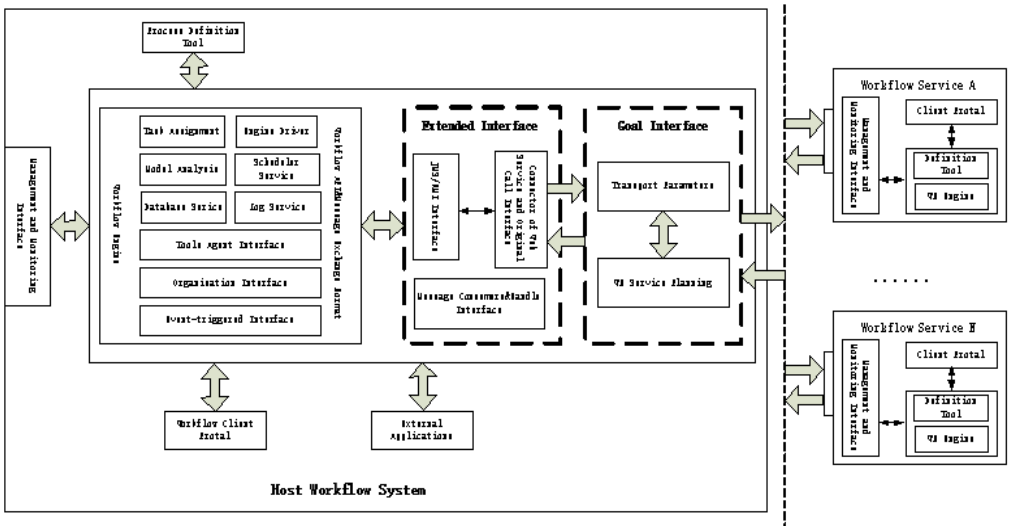


Figure 1. Heterogeneous Processes Combination Based On Service Planning

In the framework shown in Figure 1, the left side is the host process workflow system, the extended interfaces which are shown in black dashed box can communicate and pass parameters with other workflow systems. Moreover, it can plan the workflow services. The right sides of Figure 1 are the extended workflow systems which are packaged as services.

Host workflow system has the following features: (1) it provides the normal service and shields the details of the performed process to the users. The user does not know the specific processes' implementation of the engine. (2) Its workflow engine is responsible for the host process' starting and execution, and the interaction with users. (3) It can schedule and control distributed process' execution when it receives the user's task request. (4) It can obtain the planning processes' non-functional factors that characterize the status information. (5) It has planning functions. The planning engine can match a concrete process fragment according to the needs of the primary process' execution. (6) It can manage process fragments, keep abreast of the changes in various processes or

process fragments in distributed workflow systems and the function of each process fragment.

For each distribution workflow system, while maintaining the functionality of the original system, the following features need to be added: (1) their original functions should be packaged as a service. (2) Increase the interactive control over other processes to facilitate the reception and transmission parameters of the core system. (3) Increase the function of flow registration, when the flows are increased or deleted, it should promptly notify the core system, and the process should be changed correspondingly.

3.2. Interfaces of the Workflow Service For Processes Combination

When the workflow systems require multiple distributed processes to be executed collaboratively, the process of each workflow system is encapsulated to be one of the activities of the host process. So the host process' activities may run in different workflow engines, and the original simple data flow between a process' activities will become more complex, such as data exchange between the different workflow systems.

The interaction data between the workflow systems include the processes' names, process-related data, the mode of invocation, the result and the completion of execution state of the process and so on. Processes' names inform the workflow engine which process should be executed; Process-related data are the data which would be passed to the process executed when needed after the implementation of the former activity; the mode of invocation means that the host process requires synchronous or asynchronous execution. In the synchronous control mode, after acquiring the result of called service, the engine will stop waiting for the process to continue the next activity; the returned results of the process' implementation means the processes' results need to be returned to the host flow; process execution completion status refers to the end of the process execution, which is abnormal or terminated.

Web services can easily realize the data exchanged and communication between different systems, therefore, this paper provides the functions of the workflow systems for clients to invoke by encapsulating the workflow systems' interfaces. The functions of the workflow service include: (1) it provides the necessary interfaces for the external applications, such as processes' modeling and deployment, process instance's creation and starting, and the historical data's query. (2) It provides data and services during the execution of the workflow engine when it needs. Therefore, the packaged workflow service's interfaces are designed as follows:

Definition 1: Functions and Interfaces Definition of Workflow Service

Definition 1.1: Workflow Service's Input

<workflow-service-input> ::=<process-function><invoke-strategy><business-data>
<process-function> ::={processDef-id, processDef-name, operator, description}
<invoke-strategy> ::="synchronous"|"asynchronous"

Among them, "process-function" indicates the functional factors of the process, including the process's name, descriptions, definition identifiers and operators; "invoke-strategy" is the way of service invoked, including two ways - synchronous and asynchronous; "business-data" are the relevant business data which are transferred from the host flow.

Definition 1.2: Workflow Service's Output

<workflow-service-output> ::=<invoke-strategy><processIns-result><ending-status>
<invoke-strategy> ::="synchronous"|"asynchronous"
<processIns-result> ::={process-data, business-data}
<processIns-data> ::=<processIns-id, processIns-name, operator>

<ending-status> ::={ending, suspending, exception}

Among them, "invoke-strategy" is the way of service invoked; "processIns-result" is the result of an executed process's completion, including process and business data; "ending-status" is the status of the implementation of a completed process, such as the ending, the suspension pending or abnormal termination.

3.3. Extension of the Workflow System Interface

The expansion of original workflow engine interfaces' functions include the following aspects:

(1) It can create a process instance: The workflow engine generates the process instance's information(such as process's ID, name, operator and global variables) based the process's definition, and the process instance's state is initialized, and then the state of the instance would turn to running after loading its information into database and memory. Before the start of the instance, the engine would pass the above parameters, and then start the first activity. At the beginning of the activities, if the mode of invoking is synchronous, it will wake a thread after the process is finished.

(2) It can start and run a process instance: The engine would start and run the appropriate process according to the process instance's ID, parse the needed parameters for the running activity instances from the global variables and deal with the work items.

(3) It will complete the process instance: When the process runs to the end activity, firstly the engine will determine whether all the activities of the process are completed, and if it is a completion of the process, then it determines whether the father of the current process is exit; if there is none, it will return the corresponding parameters to the goal interface at the end of the end activity.

Definition 2: Extended Definition of Workflow Engine Interface

Definition 2.1: Create a Process Instance and Return Process Instance's Related Attribute Information

<interface-creatprocess> ::=<data-type><createProcess><formal-parameter-list>
<formal-parameter-list> ::=<data-type><parameter>
<data-type> ::="string"|"int"|"double"|"boolean"
<parameter> ::=<process-function, globalPara>

Definition 2.2: Start Running Process Instance

<interface-startprocess> ::=<startProcess><formal-parameter-list>
<formal-parameter-list> ::= <data-type><parameter>
<data-type> ::="string"|"int"|"double"|"boolean"
<parameter> ::=<processIns-data, globalPara>

Definition 2.3: Complete a Process Instance and Return the Instance's Result

<interface-completeprocess> ::=<data-type><completeProcess><formal-parameter-list>
<formal-parameter-list> ::= <data-type><parameter>
<data-type> ::="string"|"int"|"double"|"boolean"
<parameter> ::=<processIns-data, globalPara>

External workflow system will call and start a workflow process via "createProcess" and "startProcess" interface. The process instance is executed by the invoked workflow system's engine which will determine whether the process instance is performed completely and then return the result via "completeProcess" interface. Process data

between different workflow systems pass through the interface parameter "globalPara", and "data-type" represents the type of parameters and the function's return values.

In order to pass all kinds of external parameters in the workflow system, make the host workflow system to plan all the available process services by the obtained functional parameters, and facilitate different users to have access to the results they care, in addition to the expansion of original interfaces' function, a goal interface is also added between workflow engine and external applications. The main function of the goal interface is to receive and parse the client's parameters, to choose and create a new link to the workflow engine service, and also to receive and interpret the response of workflow engine service and then return the result back to the client. The specific approach is to establish parameter mapped between the calling and called process, to unify workflow systems' requirements for different data formats, and to assemble the required external data and the parameters for planning into XML file which acts as the goal interface's global variable. It will parse the XML before executing a process. Firstly, it will plan the process services based the indicators which are cared by the users, and the global variables will be mapped to the process which is located in the best workflow service. Secondly, it will analyze and acquire the process data needed by the follow-up activities. Finally, at the end of the process, the operating results and other parameters will be assembled according to a uniform format for the caller to parse and get the appropriate data.

In order to match the input and output parameters of the service interface effectively, referring to the previous section, the goal interface is defined as follows:

Definition 3: Goal Interface Definition

<interface-goal> ::=<Goal><function-list>
<function-list> ::=<function-generateXML><function-parseXML>
<function-generateXML> ::=<data-type><generateXML><formal-parameter-list1>
<function-parseXML> ::=<data-type><parseXML><formal-parameter-list2>
<data-type> ::="string"|"int"|"double"|"boolean"
<formal-parameter-list1>=<data-type><goal-parameters>
<formal-parameter-list2>=<data-type><parameter>
<goal-parameters> ::=<goal-parameter-name, goal-parameter-value>
<parameter> ::=<xml-file, node-name>

Among them, "generateXML" is used for assembling the goal's parameters into XML file in a uniform format; "parseXml" is used for parsing the XML file which is assembled by the goal's parameters; "goal-parameters" is the goal's parameters which include the global variables of the process, the parameters for process services' planning and the external data; "node-name" is the parameter required to parse, such as process data, planning parameters, processes' results and so on.

3.4. Process Services Combination Based On Planning

Selecting the available process service is better than specifying a specific service's link while modeling [4], as it can reflect the dynamic nature of the process's combination. Dynamic programming method is used to match the best optimal flow from the existing processes based on the customer's needs. The factors that affect the planning will change when the processes may be deployed in a distributed environment, and the results of the process planning from the original fragment [13] into process services' interfaces and process' names. The evaluative rules will change, if the current service is available and online, mainly in the cost of the processes' execution time in different engines that can realize the same purpose, the level of the services' reliability, the cost of the services and so on. Therefore, a process service planning algorithm adapted for distributed processes' execution is as follows:

Algorithm: Service planning matching algorithm **FlowServiceMatch(PF, ContextTerm[])**

Input: "PF" is a process service's functional requirements, such as the process' name; "ContextTerm[]" is the evaluation of parameters for planning, such as service cost, time spent and the level of service's credibility.
Output: "WFServiceLink" is a workflow service link which is matched for the host process' needs; "subProcess" is the optimal flow which will be executed on this service's engine.

```
// parse the process' current node and judge if there has a sub-process need to be planned
ParserXML(Process.curNode)
if  node.type=SUBFLOW_ACT  AND  getAct.byName=PF
        then
                // match all the workflow services which included the process
                WFService[i] ⟵ ProcessMatch(ProcessList)
                // selecte the workflow services which are available and online
                  WFServiceList.add(WFService.isContain(WFService[i]).isOnLine)
                 PlanFactorsTerm[] ⟵ getContextTerm[].byName
                // extract indicators and their weights from the parameters for planning
                  Factor_Weight[] ⟵ getContextTerm[].byValue
              // get all the available processes' parameters from the state list
              // such as service cost, running time, the level of credibility
            for k ⟵ 1 to WFServiceList.size
          termName[] ⟵
getFactorLabel.byName(Ws_Cost,Ws_ConsumeTime,Ws_CredibilityLevel)
                scenarioTerm[] ⟵ gettermName[].byValue
          // plan and match the optimal process according to the calculation between the
          expected
          // weight of users with the state parameters' value
                WFServiceLink ⟵
          minValue.Compare(Factor_Weight[],scenarioTerm[],operator)
                                (OrMaxValue.Compare(Factor_Weight[],scenarioTerm[],op
                            erator))return WFServiceLink+subProcess
```

The algorithm uses a sub-process's information to plan and find out the workflow system that has the sub-process's definition and has been registered in a distributed environment. It also matches the process service's link which could meet the best need of clients by some planning parameters.

4. Case Study

In the following part, it will take three kinds workflow systems as an example for illustration. "SynchroFlow" is a commercial workflow system based on XPDL workflow language, the open source ODE is a workflow engine that could execute BPEL processes, and the scientific workflow Kepler. SynchroFLow is the host workflow system among them. These three workflow systems are packaged and deployed by the open source Apache Ant tool.

A process can be packaged into service as one of the host process's activities, and its implementation is on a different engine. As shown in Figure 2, the activity named "fill in

the questionnaire and statistics" is a sub-process activity of the host process. Both the Kepler and BPEL processes shown in Figure 2 have the same function of the Questionnaire's process. When the host process is executed to the sub-process activity, it will plan the best service from this two based on the service function's description and the need of the user.

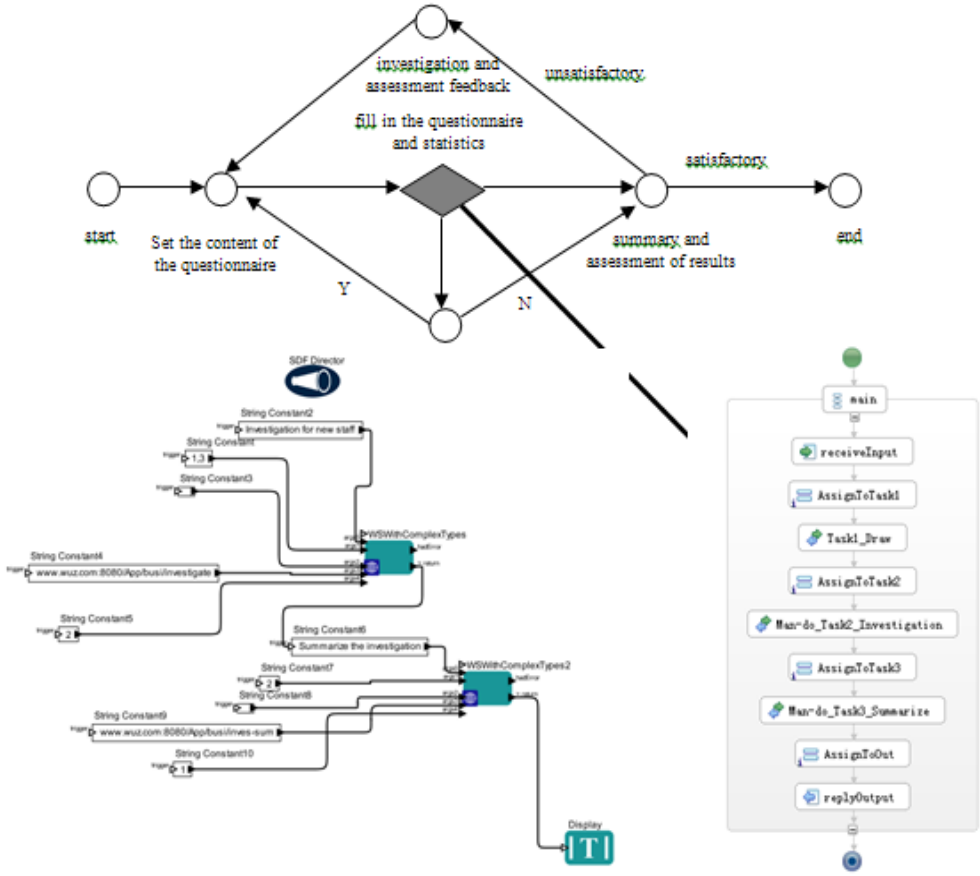


Figure 2. SynchroFlow, Kepler and BPEL Processes Composition Case

The value of an alternative process's planning parameters could be extracted and parsed from the process goal's parameters, historical data or vendor's setting. For example, the value of Kepler and BPEL Questionnaire processes' planning parameters are shown in Table 1, the indicators' weight according to the user's need are assigned in parenthesis.

Table 1. Alternative Processes' Planning Parameters and Evaluation Result

	Service Charge(0.5)	Time Spent(0.3)	credibility Level(0.2)	Evaluation Result
Kepler	5	10	1	5.7
BPEL	2	12	1	4.8

By calculating, we can find the BPEL process has the smaller evaluation result which is closer to the need of user. So the host process will match the "InvokeBpelService" from the workflow services provided in the current network, the ODE engine would be started and the sub-process would be executed in ODE engine. After the end of this BPEL process's execution, workflow server will return the control right back to SynchroFLOW engine and continue to run the next activity of the host flow.

5. Conclusions and Future Work

In this paper, aiming at the workflow process combination based on web service, we made an analysis on service encapsulation, parameter passing, interfaces and planning indicators of different workflow systems. In addition, dynamic planning is used for selecting the available and optimal service in the runtime. The dynamics and flexibility of process composition is greatly improved. However, there are still many issues that should be considered on the evaluation of planning results, and the exception handling of process composition under the distributed environment. Our future work will continue to focus on the methods of the planning results' evaluation and exceptions' handling.

Acknowledgements

The presented work is partially supported by the National Science and Technology Support Program (2012BAK17B09).

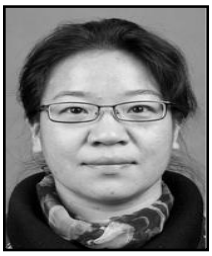
References

[1] J. Zhang, D. Kuc and S. Y. Lu, "Confucius: A tool supporting collaborative scientific workflow composition", IEEE Transactions on Services Computing, vol. 1, no. 7, **(2014)**.
[2] D. Hollingsworth, "The workflow reference model", Workflow Management Coalition, **(1995)**.
[3] B. Yang, K. Yan, J. S. Jiang and G. Y. Hu, "Web Services architecture oriented cooperative workflow model", Computer Engineering and Design, vol. 3, no. 32, **(2011)**.
[4] S. D. I. Fernando and A. C. Simpson, "Towards a formal framework for workflow interoperability", Lecture Notes in Computer Science, vol. 5387, **(2009)**.
[5] G. Pavlin, M. Kamermans and M. Scafes, "Dynamic process integration framework: Toward efficient information processing in complex distributed systems. Informatica, vol. 34, **(2010)**.
[6] G. Terstyanszky, T. Kukla, T. Kiss, P. Kacsuk, A. Balasko and Z. Farkas, "Enabling scientific workflow sharing through coarse-grained interoperability", Future Generation Computer Systems., vol. 37, **(2014)**.
[7] A. Alqaoud, I. Taylor and A. Jones, "Scientific workflow interoperability framework", International Journal of Business Process Integration and Management, vol. 1, no. 5, **(2010)**.
[8] F. X. Xiao, Z. Q. Huang, Z. N. Cao, L. Z. Tu and Y. Zhu, "Unified formal modeling and analyzing both functionality and Qos of web services composition", Journal of Software, vol. 11, no. 22, **(2011)**.
[9] Q. Q. Fang, X. M. Peng, Q. H. Liu and Y. H. Hu, "Study of web service composition on combining AI planning with workflow", Computer Science, vol. 9, no. 36, **(2009)**.
[10] M. Falou, M. Bouzid, A. Mouaddib and T. Vidal, "A distributed Planning Approach for Web Services Composition", IEEE International Conference on Web Services, **(2010)** July 5-10, Miami, USA.
[11] S. K. Jing, H. Jiang, W. T. Xu and J. T. Zhou, "Cloud manufacturing service composition considering execution reliability", Journal of Computer-Aided Design & Computer Graphics, vol. 3, no. 26, **(2014)**.
[12] T. X. Song and C. M. Wei, "Automatic composition of Web Services based on rules mapping", 2nd International Asia Conference on Informatics in Control, Automation and Robotics, **(2010)** March 6-7, Wuhan, China.
[13] D. W. Chen, R. Z. Sun, Y. Xiang and Y. X. Shi, "Static approach for process pattern-based workflow planning", Computer Engineering and Design, vol. 1, no. 32, **(2011)**.
[14] B. Xie, Y. Yang, J. Song and L. Wang, "Service workflow optimization and composition method based on templates", Journal of University of Science and Technology Beijing, vol. 10, no. 34, **(2012)**.

Authors

Yuan Gang, Working toward the PhD degree in the College of Information and Electrical Engineering, China Agricultural University. Her current research interests include workflow technologies and applications, business process management and web services.

Sun Rui-Zhi, Received his PhD degree (2003) in Tsinghua University. He is a Full Professor with the College of Information and Electrical Engineering, China Agricultural University. His research interests include computer network and applications, workflow management and cloud computing.



Shi Yin-Xue, Working toward the PhD degree in the College of Information and Electrical Engineering, China Agricultural University. Her current research interests include business process management, workflow technologies and applications.