Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 512476, 9 pages
http://dx.doi.org/10.1155/2014/512476

Research Article

Modeling, Design, and Implementation of a Cloud Workflow Engine Based on Aneka

Jiantao Zhou,[1] Chaoxin Sun,[1] Weina Fu,[1] Jing Liu,[1] Lei Jia,[1] and Hongyan Tan[2]

[1] *College of Computer Science, Inner Mongolia University, Hohhot 010021, China*
[2] *High Performance Network Lab Institute of Acoustics, Chinese Academy of Sciences, Beijing 100190, China*

Correspondence should be addressed to Jiantao Zhou; zhoujiantao@tsinghua.org.cn

Received 28 January 2014; Accepted 11 March 2014; Published 29 April 2014

Academic Editor: X. Song

This paper presents a Petri net-based model for cloud workflow which plays a key role in industry. Three kinds of parallelisms in cloud workflow are characterized and modeled. Based on the analysis of the modeling, a cloud workflow engine is designed and implemented in Aneka cloud environment. The experimental results validate the effectiveness of our approach of modeling, design, and implementation of cloud workflow.

1. Introduction

With the successful cases of the world's leading companies, for example, Amazon and Google, cloud computing has become a hot topic in both industrial and academic areas. It embraces WEB 2.0, middleware, virtualization, and other technologies and also develops upon grid computing, distributed computing, parallel computing, and utility computing, and so forth [1]. Comparing with classic computing paradigms, cloud computing provides "a pool of abstracted, virtualized, dynamically scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet" [2]. It can provide scalable resources conveniently, on demand to different system requirements. According to features of services mainly delivered by the established cloud infrastructures, researchers separated the services into three levels, which are infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS). IaaS offers hardware resources and computing power, such as Amazon S3 for storage and EC2 for computing power. PaaS targets providing entire facilities including hardware and the application development environment, such as Microsoft Azure Services platform and Google App Engine. SaaS refers to those software applications offered as services in cloud environments.

However, along with the development of cloud computing, corresponding issues are also arising in both theoretical and technical aspects. One of the most prominent problems is how to minimize running costs and maximize revenues on the premise of maintaining or even improving the quality of service (QoS) [3]. Workflow technology can be regarded as one of the solutions [2–4]. A workflow is defined as "the automation of a business process, in whole or part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules" by the workflow management coalition (WFMC) [5]. Workflow management system (WFMS) is a system for defining, implementing, and managing the workflows, in which workflow engine is the most significant component for task scheduling, data movement, and exception handling.

Cloud workflow, also called cloud-based workflow [1] or cloud computing-oriented workflow [3], is a new application mode of workflow management system in the cloud environment, whose goal is to optimize the system performance, guarantee the QoS, and reduce running cost. It integrates workflow with cloud computing, which combined the advantages of both sides. Cloud workflow technology can be used in two levels [3]; from the cloud users' view, it supports process definitions of cloud applications and enables flexible configuration and automated operation of the processes. This kind of cloud workflow is regarded as "above-the-cloud," such

as Microsoft Biztalk workflow service and IBM Lotuslive. From the cloud service providers' view, cloud workflow offers automatic task scheduling and resource management of cloud computing environment. This category is referred to as "in-the-cloud," for example, Cordys Process Factory (for Google APPs).

There is still much timely and worthwhile work to do in the field of cloud workflow, for example, scalability and load balance of above-the-cloud workflow, optimization and integration of in-the-cloud workflow, and so on. The goal of our work in this paper is to design an above-the-cloud workflow engine based on Aneka cloud platform [6], considering scalability and load balance. The remainder of the paper is organized as follows. Related work about workflow, grid workflow, and cloud workflow is given in Section 2. The core part is given in Section 3, including formal modeling of workflow processes and design of the Aneka-based cloud workflow engine. Then, the implementation and experiments are presented in Section 4. Effectiveness and correctness of the novel workflow engine are shown by analysis results. Finally, in Section 5, the main results of the paper are summarized.

2. Related Work

From the mid-1990s to around 2000, business process management and workflow technology were developed rapidly, and many valuable results and products were gained. The detailed surveys can be found in [7–9]. Besides, flexibility, adaptability, dynamic changes, and exception handling of workflow have also been focused on and taken progress in the next few years [10–13]. Thus, from the modeling and verification of workflow process, to the design and implementation of workflow engine, to the building and practice of workflow management system, a series of results have made workflow technology play a more and more important role in social life.

In the next ten years or so, with the emergence of new computer technologies and computing paradigms, for example, web services, P2P, and grid computing, workflow can be developed in two aspects. On the one hand, the existing workflows can be transferred to new computing paradigms. On the other hand, adapting to new features of technologies and paradigms, workflow technology should make a progress. Grid workflow can be classed into two categories, above-the-grid and in-the-grid, as the discussion of cloud workflow in the above section. The goal of in-the-grid workflow is integration, composition, and orchestration of grid services in the grid environment, considering peer-to-peer service interaction and complicated lifecycle management of grid services [14]. In regard to above-the-grid workflow, transformation of traditional business workflow systems is the part of the work, and many researches and practices have been done concerning distributed and parallel features of grid [15, 16], covering modeling, verification, scheduling problems, and so on [17–21].

Comparing cloud computing emerging in 2007 with grid computing [2], it can be seen that their visions are the same; meanwhile, there are both similarities and differences between them, from architecture, security model, business model, and computing model to provenance and applications. Based on the analysis of workflows running in these two kinds of infrastructure, similarities and differences are also found [3, 22]. There are many workflow studies on different levels of cloud services. Research of workflows building on IaaS focuses on dynamical deployment and monitoring in cloud nodes, which is used in large-scale data intensive computing [23, 24]. Some researchers use cloud workflows in community team working among multiple processes [25]. Otherwise, many workflow studies in PaaS pay attention to the integration of cloud and workflow. It is concerned with recognition and execution of workflow tasks in cloud environment [26, 27]. Today, there are also many studies on scientific and commercial cloud workflows. Yuan et al. studied data dependency and storage on scientific cloud workflows [28, 29]. Wu et al. carried out researches on hierarchical scheduling strategy in commercial cloud workflows [30].

According to our analysis, three kinds of differences or improvements can be concluded. Firstly, cloud workflow technology research is always carried out joint with multiple technologies and computing paradigms, such as web services, P2P, and grid computing. Secondly, cloud workflow concentrates more on data and resources and not just the control flow. Thirdly, performance of cloud workflow is paid more attention than functionalities [31–33].

3. Modeling and Design of Aneka-Based Cloud Workflow Engine

Our Aneka-based cloud workflow engine will be given in this section. To improve scalability, the cloud workflow engine will be designed to support different parallelism levels of workflow processes. And to clarify these parallelisms, the formal modeling technique for processes is given firstly.

3.1. Preliminaries. Workflow management system completely or partly supports automation of workflow schema. And workflow schema models business processes; it is characterized by the decomposition into subflows and atomic activities, the control flow between activities (subflows), data flow and data, and the assignment of resources (including human resources and equipment resources) to each activity [5]. That is, workflow schema is a combination of three essential dimensions: control flow, data flow, and resource flow.

Petri net is a simple, graphical, yet rigorous, mathematical formalism, which has been used to model workflow processes [34]. However, the usage of Petri net is always limited to model control flow, which is not enough for describing the above three dimensions of workflow. An advanced model founded on the basic Petri net has been developed in our previous paper [35], called 3DWFN. Its definition is given as follows, which is suitable for describing cloud workflow.

Definition 1. Three-Dimension WorkFlow Net, 3DWFN.

Let Σ, Γ, and Ψ be finite alphabet sets of activity names, data names, and resource names, respectively.

A three-dimension workflow net over them is a system $3DWFN = \langle S, T, F, C; Lab, Exp, M_0\rangle$, where we have the following:

(i) finite sets S of places and T of *transitions*: $S \cap T = \emptyset$, $S \cup T \neq \emptyset$, and $T = Ta \cup Tp \cup T\tau$, where Ta is a set of *atomic transitions*, Tp is a set of *subnet transitions*, and $T\tau$ is a set of *internal transitions*;

(ii) $F \subseteq (S\times T)\cup(T\times S)$ is a set of arcs, especially, including the set of inhibitor arcs: $F^\circ \subseteq F$;

(iii) C is a finite and nonempty *color set* including types of tokens and the default is denoted by a black dot, "•;"

(iv) $Lab: T \to \Sigma, C \to \Gamma\cup\Psi\cup\Gamma\cup\Psi$ is a *labeling function*;

(v) $Exp: F \to N \times C \times Con$ is an *arc expression function*, where N denotes the set of natural numbers and Con is a set of expressions of numeric computation or logic computation;

(vi) $M_0 : \mathbf{S} \to \{\emptyset, \mu\mathbf{C}\}$ is the initial marking of the net, where $\mu\mathbf{C}$ is the multiset over C.

In view of defining transition rule of 3DWFN, some notations are introduced beforehand.

Definition 2. (i) *Projection*. Suppose $Fx : Q \to P_1 \times P_2 \times P_3 \times \cdots, \exists q \in Q, p_i \in P_i\ (i = 1, 2, 3, \ldots) : Fx(q) = (p_1, p_2, p_3, \ldots), Fx(q) \uparrow P_1 \times P_2$ represents $Fx(q)$'s projection on $P_1 \times P_2$, and $Fx(q) \uparrow P_1 \times P_2 = (p_1, p_2)$ or $\{p_1, p_2\}$.

(ii) *Variables Replacement*. Let "$x : A$" represent x is a variable of set A. Variable x may be replaced by any element in A.

Then the transition rule is given.

Definition 3. Transition Rule of 3DWFN.

(i) *Preconditions* about a transition t are denoted as for all $s \in {}^\bullet t, (s,t) \notin F^\circ : \mathrm{Pre}(s,t) = \mathrm{Exp}(s,t)$, $\mathrm{Pre}(t) = \bigcup_{s\in} {}_{\bullet t} \mathrm{Exp}(s,t)$, and for all $s \in {}^\bullet t, (s,t) \in F^\circ : \mathrm{Prev}(s,t) = \mathrm{Exp}(s,t)$.

(ii) *Postcondition* about a transition t is denoted as for all $s \in t^\bullet$, : $\mathrm{Post}(t,s) = \mathrm{Exp}(t,s), \mathrm{Post}(t) = \bigcup_{s\in t^\bullet} \mathrm{Exp}(t,s)$.

(iii) A transition $t \in T$ is *enabled* under a marking M if and only if $(\text{for all } s \in {}^\bullet t, (s,t) \notin F^\circ : M(s) \geq (\mathrm{Pre}(s,t) \uparrow \mathbf{N}\times\mathbf{C})) \wedge (\text{for all } s \in {}^\bullet t, (s,t) \in F^\circ : M(s) \prec (\mathrm{Prev}(s,t) \uparrow \mathbf{N} \times \mathbf{C}))$.

(iv) A new *marking* M' is produced after an enabled transition fires: $M' = M - (\mathrm{Pre}(t) \uparrow \mathbf{N}\times\mathbf{C}) + (\mathrm{Post}(t) \uparrow \mathbf{N} \times \mathbf{C})$ denoting as $M[t > M'$ or $M \xrightarrow{t} tM'$.

3.2. Modeling and Analysis of Workflow Process. With detailed analysis of generalized cloud workflow systems, it is really indispensable to discriminate different parallelisms of workflow processes for adopting cloud technologies to promote its execution efficiency. Therefore, we divide the execution of multiple tasks in a cloud workflow system into three levels according to different parallelisms, that is, process level execution, task level execution, and application level execution.

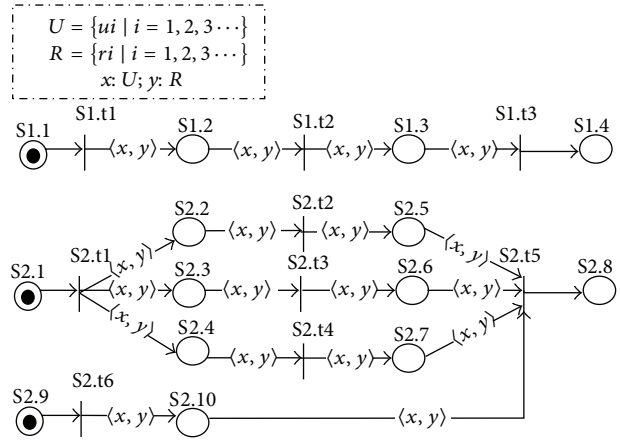

FIGURE 1: Parallel processes modeled by 3DWFN.

As shown in Figure 1 by 3DWFN, there are two parallel processes, named S1 and S2. S1 shows a sequential process and S2 shows a process including parallel tasks. U and R are colored sets. U represents the set of users, while R represents the set of resources. As a result, execution of a 3DWFN looks like a user acting, as some role carries some kind of data or uses some kind of resources to "walk" through a certain path of the net.

These two processes could be performed in parallel at different computing node in cloud environment, which illustrates the parallelism of process level execution in the cloud workflow system.

When the workflow system enter state S2.2, S2.3, and S2.4 after transition S2.t1 fired, three tasks that are represented by S2.t2, S2.t3, and S2.t4 are enabled to be carried out in parallel at different computing node in cloud environment controlled by a single user or multiple users. Then, the joint task S2.t5 could be triggered to be executed only if resources in S2.5, S2.6, S2.7, and S2.10 are all available. This scenario illustrates the parallelism of task level execution in the cloud workflow system.

Finally, if a single task is intensive computing, such as the task execution between S1.2 and S1.3, it could be divided into more fine-grained subtasks and carried out in parallel at different computing node in cloud environment controlled by a single user. This scenario shows the parallelism in application level.

3.3. Architecture of Cloud Workflow. According to the above-mentioned analysis, the integrated framework of Aneka-based cloud workflow engine is presented. There are three parts, environment, applications, and control parts, which can solve the analyzed parallelisms problems in the three levels and achieve extensibility and reusability of workflow. Details are presented in Figure 2.

3.3.1. Cloud Workflow Environment. There are three executing models in the Aneka cloud environment, Task Model, MapReduce Model, and Thread Model. In the remainder of this paper, all experiments are run with Task Model.

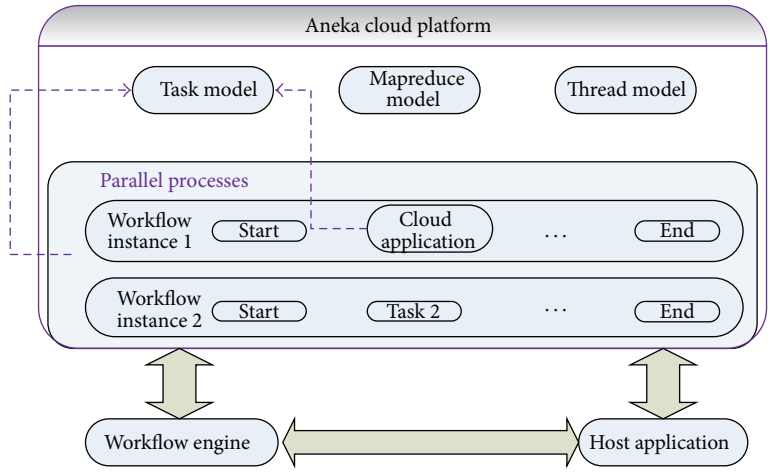

Figure 2: Integrated framework of Aneka-based cloud workflow engine.

3.3.2. Cloud Workflow Applications. The cloud workflow applications contain all instances of workflow processes.

3.3.3. Cloud Workflow Control. The cloud workflow control part contains the workflow engine and host application. The cloud workflow engine is in charge of running workflow instances, and the host application is in charge of defining workflow processes and monitoring workflow process executions.

3.4. Design of Cloud Workflow

3.4.1. Design of Workflow Runtime Environment. A lightweight cloud workflow engine is designed, whose functions include starting process execution, scheduling processes, and tasks based on preestablished rules. The workflow runtime environment is shown in Figure 3, which is composed of three parts, host application, workflow instances, and runtime engine.

Then, Figure 4 shows all service classes design of the workflow engine. The class "WorkflowRuntimeService" is the base class and the others are the derived classes.

3.4.2. Design of Cloud Workflow Execution Process. After a process is started, each task will be executed following the control flow when requirements of data flow and resource flow are met. When a task is enabled, the task executor will send its execution request to workflow engine. Then, the workflow engine will instantiate the ready task. If the task is not intensive computing, then it will be executed locally. Otherwise, if the task is intensive computing, it will be submitted to the cloud. The execution process of cloud workflow is depicted as in Figure 5.

3.4.3. Design of Task Model Submission Process. Next, the submission process of task is designed. As mentioned above, Task Model is chosen. In Aneka cloud environment, Task Model is used not only to solve the distributed applications which are composed of single tasks, but also to execute

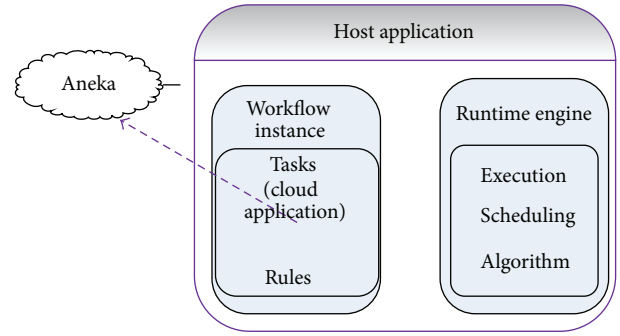

Figure 3: Workflow runtime environment.

the correlating tasks of these applications. Once users submit their sequence of tasks with rules, the results will be returned by Aneka after a while.

Aneka Task Model is composed of the class AnekaTask, the interface ITask, and the class AnekaApplication. The task submission process in Aneka Task Model is as follows: firstly, to define a class "UserTask," which inherits class "AnekaTask" in Aneka Task Model; secondly, to create an instance of UserTask for the application program; and thirdly, to package class "UserTask" instance to class "AnekaTask" and submit it to Aneka cloud by class "AnekaApplication." The sequence diagram of the above process is shown in Figure 6.

Then, the implementation of workflow tasks is presented in Figure 6. Firstly, the implementation manager submits tasks to the workflow runtime engine. Then, the workflow runtime engine selects a custom made operation flow and creates its workflow instance and then runs it at the same time.

4. Implementation and Experiment

4.1. Building Aneka Cloud Environment. Our experimental cloud environment includes a server as a master node, some common PCs as the worker nodes, and a manager

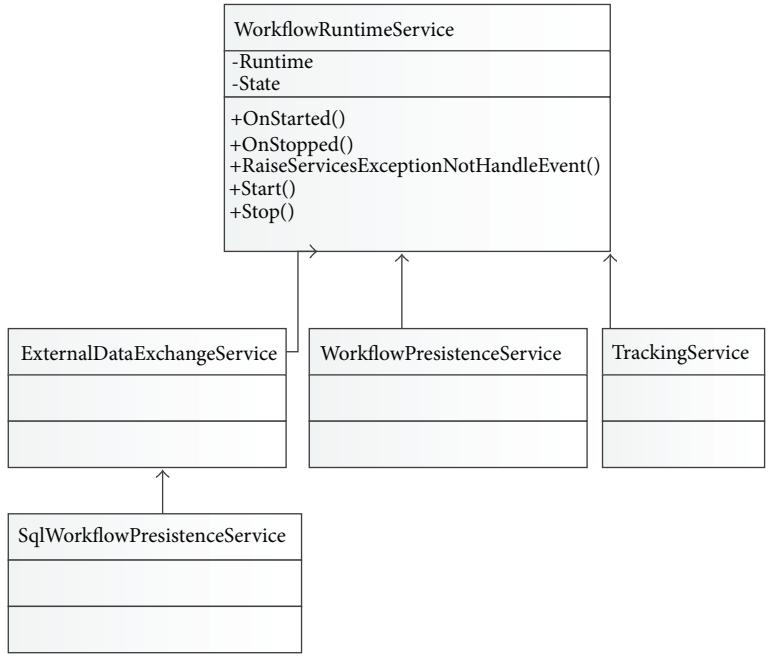

Figure 4: Class diagram WF service.

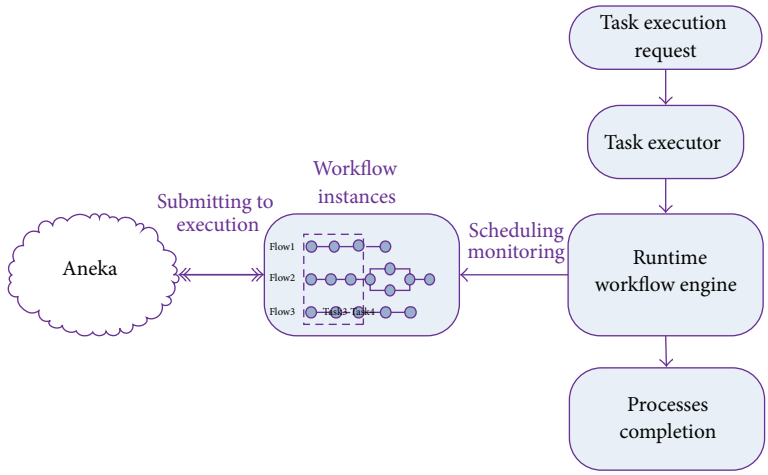

Figure 5: Process of cloud workflow execution.

node. In this paper, the detailed hardware configuration is presented in Table 1.

Then, the cloud nodes are pretreated according to the following steps before installing Aneka.

(a) Install FTP server on the master node and worker nodes.

(b) Offer access authority for the manager node to the master node and worker nodes.

After pretreatment, Aneka cloud management platform is installed in the manager node. Then, we use remote access to install and configure the master node and worker nodes.

4.2. Workflow Process of an Example. Our example is to compute definite integral by a probabilistic method. The workflow process is composed of four steps: "generations of random number," "computing X-axis," "computing Y-axis," and "computation of final result."

In this processing, there are three correlating tasks, which are "generation of random numbers", "computation of X-axis x" and "computation of Y-axis y." We use the task "generation of random numbers" to compute real and

Table 1: Aneka cloud environment configuration.

Type of node	Operating system	Quantity of computers
Manager	Windows 7	1
Master	Windows Server 2008	1
Worker	Windows 7	3

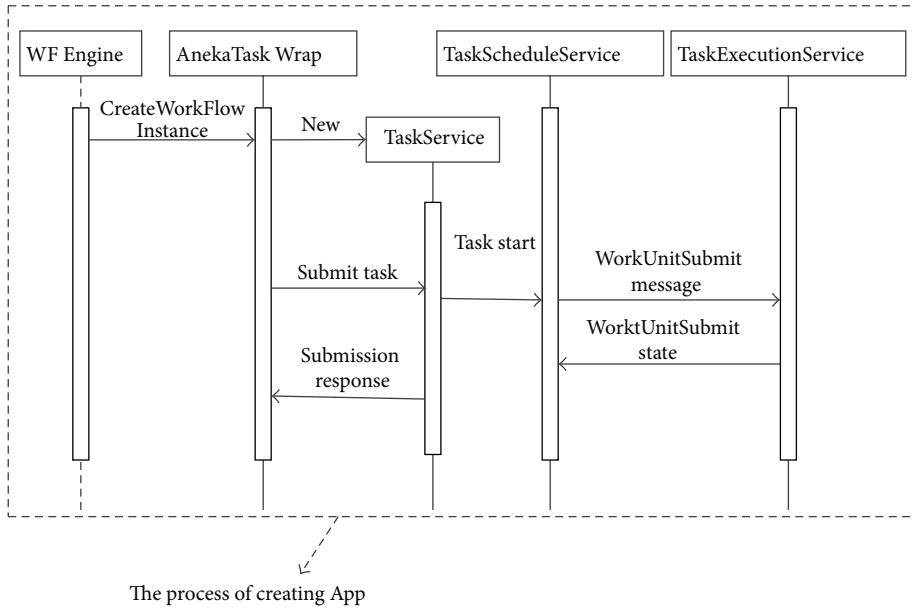

Figure 6: Process of submitting tasks.

imaginary axes. Then, we match and combine values of x, y to a $point(x, y)$. Admittedly, the combinational step is next to the computational step of $f(x, y)$, and the steps are in one task. In this case, we divide the position relations of all points into two cases. When a $f(x, y)$ is between $x = a$ and $x = b$, we let the count increase by 1. In contrary, count is not changed. Then, letting the number of all points be equal to number n, we use $p = R/n$ to estimate $\int_a^b f(x)$. The operation flow is present in Figure 7.

4.3. Implementation Methods. The task of computing axes is intensive computing and will be submitted to the Aneka cloud in Task Model. The execution function under the interface ITask is presented in Algorithm 1. Then, the task is packaged and submitted to the cloud by the above mentioned class AnekaApplication. The core implementation is configured in Algorithm 2.

4.4. Results Analysis. Based on the above experiments, running results and workflow logs are analyzed. Three analysis conclusions are given as follows. Firstly, as shown in Table 2, the results of definite integral workflow are presented with different number of points. It shows that the degree of accuracy increases along with the increase of task's number. It accords with the mathematical regular rule. It is indicated that the functionality of our cloud workflow engine is normal.

Secondly, the intensive computing tasks submitted to the Aneka cloud are executed in parallel by different workers. As shown in Figure 8, the screenshot of workflow log, tasks A and B represent, respectively, the two intensive computing tasks, "computing X-axis" and "computing Y-axis." It is indicated that our cloud workflow engine is effective and efficient.

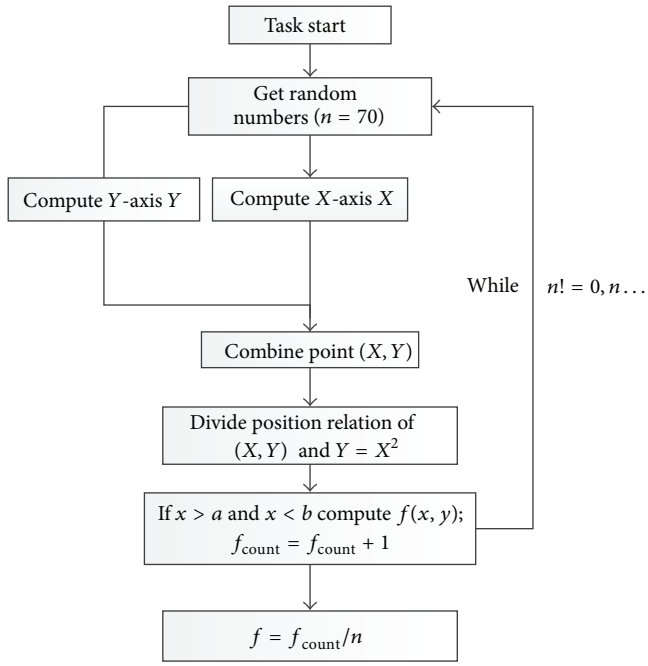

Figure 7: Process diagram of task.

Table 2: Result of different number of the spots.

Number of the spots	70	100	150	1000
Computing result	0.286	0.310	0.393	0.342

Thirdly, the running time of the whole workflow completion is not simple linear growth with the number of points used in definite integral. The screenshot of workflow log shown in Table 3 presents the time table of 10 tasks for their

```
public void Execute ()
{
this.result = (this.y * this.y) * this.rd;.
}
public void Execute ()
{
        this.result = this.x + (this.y – this.x) * this.rd:
}
```

Algorithm 1: Code of execute function.

```
private static AnekaApplication < AnekaTask, TaskManager > Setup (string [] args)
    {
        Configuration conf = Configuration.GetConfiguration ("configuration file");
        // ensure that SingleSubmission is set to false
        // and that ResubmitMode to MANUAL.
        conf.SingleSubmission = false;
        conf.ResubmitMode = ResubmitMode.MANUAL;
        conf.UserCredential = new Aneka.Security.UserCredentials("administrator", "");
        AnekaApplication < AnekaTask, TaskManager > app =
            new AnekaApplication < AnekaTask, TaskManager > ("Workflow1", conf);
        // ensure that SingleSubmission is set to false
        if (args.Length = = 1)
        {
            bLogOnly = (args [0] = = "LogOnly"? true: false);
        }
        return app;
      }
|
```

Algorithm 2: Aneka application configuration.

```
Setting Up Aneka Application..
Executing Cloud WorkFlowTest 1
The workunit 1 of task A finished | The result is :  0.226948272076877
The workunit 2 of task A finished | The result is :  0.401436967962159
The workunit 1 of task B finished | The result is :  0.577157754719796
The workunit 2 of task B finished | The result is :  0.882427516804276
The workunit 3 of task A finished | The result is :  0.71877162331658
The workunit 3 of task B finished | The result is :  0.719695053398467
The workunit 4 of task A finished | The result is :  0.507233585932866
The workunit 5 of task A finished | The result is :  0.464722440794447
The workunit 4 of task B finished | The result is :  0.908792505463954
The workunit 5 of task B finished | The result is :  0.42897149195381
The workunit 6 of task B finished | The result is :  0.327410095523768
The workunit 6 of task A finished | The result is :  0.932547775065781
The workunit 7 of task A finished | The result is :  0.0863404269732257
The workunit 7 of task B finished | The result is :  0.841972474400873
The workunit 8 of task A finished | The result is :  0.58476772419399
The workunit 8 of task B finished | The result is :  0.891292396882219
The workunit 9 of task B finished | The result is :  0.824472365819138
The workunit 9 of task A finished | The result is :  0.816016230646528
The workunit 10 of task A finished | The result is :  0.163874565234349
The workunit 10 of task B finished | The result is :  0.112339088745573
```

Figure 8: Process of task execution.

execution time, waiting time, and total time. Mean execution time is about 2.6 seconds, and mean waiting time is about 1 second except the last task. The waiting time of the last task is so long that it is nearly equal to the sum time of the other 9 tasks. In this case, if the number of used points is increased in the next experiment, the total time might not be increased but decreased, because the cloud might assign more workers (resources) to execute them to save time. It is indicated that the cloud workflow engine is scalable, which will not spend an impossible large amount of time when a lot of tasks and processes run in parallel.

5. Conclusion

The work in this paper can be concluded as follows. A Petri net-based model called 3DWFN is given firstly, which can describe three dimensions of a workflow, that is, control flow, data flow, and resource flow. According to the analysis of the existing workflow systems, it is found that cloud workflow pays more attention to data, resources, and performance than control flow and functionality researched commonly in traditional workflow. Thus, 3DWFN is suitable for modeling of cloud workflow processes. Through analysis of the features of workflow processes executed potentially in cloud environment, three kinds of parallelisms are recognized as process level, task level, and application level and then modeled specifically by 3DWFN. The goal of the following design of cloud workflow engine is to support these parallelisms to improve scalability by using resources as far as in parallel.

Then, the architecture of the Aneka cloud based workflow engine is designed. The workflow runtime environment and execution process are stated, and the process of packaging and submitting an intensive computing task to Aneka is

Table 3: Timetable of task.

Total execution time	Total waiting time	Total time spent
00:00:02	00:00:00.5730000	00:00:02.5730000
00:00:06	00:00:01.2300000	00:00:07.2300000
00:00:02	00:00:00.6670000	00:00:02.6670000
00:00:02	00:00:01.3100000	00:00:03.3100000
00:00:03	00:00:00.8700000	00:00:03.8700000
00:00:04	00:00:00.0470000	00:00:04.0470000
00:00:03	00:00:00.4500000	00:00:03.4500000
00:00:01	00:00:02.9370000	00:00:03.9370000
00:00:01	00:00:00.7000000	00:00:01.7000000
00:00:02	00:06:019.5058631	00:06:15.9415449

explained. After the engine is implemented, a definite integral process is used as an example. By different numbers of points used to definite integral, the functionality and effectivity of the cloud workflow engine are shown. Based on the analysis of running results and workflow logs, the scalability and efficiency of the cloud workflow engine are given.

In the future, the research can be improved in following directions. First, more practical workflow processes will be designed using powerful expression of 3DWFN. Second, novel cloud workflow engine will be built, which has dynamic scheduling and handling functions with complicated process.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by Grants of the National Natural Science Foundation of China [nos. 61262082 and 61261019], Key Project of Chinese Ministry of Education [no. 212025], and Inner Mongolia Science Foundation for Distinguished Young Scholars [2012JQ03]. The authors wish to thank the anonymous reviewers for their helpful comments in reviewing this paper.

References

[1] M. A. Vouk, "Cloud computing—issues, research and implementations," *Journal of Computing and Information Technology*, vol. 16, no. 4, pp. 235–246, 2008.

[2] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Proceedings of the Grid Computing Environments Workshop (GCE '08)*, pp. 1–10, November 2008.

[3] X. Z. Chai and J. Cao, "Cloud computing oriented workflow technology," *Journal of Chinese Computer Systems*, vol. 33, no. 1, pp. 90–95, 2012.

[4] R. Buyya, J. Broberg, and A. M. Goscinski, Eds., *Cloud Computing: Principles and Paradigms*, vol. 87, John Wiley & Sons, 2010.

[5] D. Hollingsworth, "Workflow management coalition the workflow reference model," Tech. Rep. 68, Workflow Management Coalition, Hampshire, UK, 1993.

[6] R. N. Calheiros, C. Vecchiola, D. Karunamoorthy, and R. Buyya, "The Aneka platform and QoS-driven resource provisioning for elastic applications on hybrid clouds," *Future Generation Computer Systems*, vol. 28, no. 6, pp. 861–870, 2012.

[7] W. M. P. Van Der Aalst, A. H. M. Ter Hofstede, and M. Weske, "Business process management: a survey," in *Business Process Management*, vol. 2678 of *Lecture Notes in Computer Science*, pp. 1–12, Springer, Berlin, Germany, 2003.

[8] R. Lu and S. Sadiq, "A survey of comparative business process modeling approaches," in *Business Information Systems*, pp. 82–94, Springer, Berlin, Germany, 2007.

[9] E. M. Bahsi, E. Ceyhan, and T. Kosar, "Conditional workflow management: a survey and analysis," *Scientific Programming*, vol. 15, no. 4, pp. 283–297, 2007.

[10] H. Schonenberg, R. Mans, N. Russell, N. Mulyar, and W. Van Der Aalst, "Process flexibility: a survey of contemporary approaches," in *Advances in Enterprise Engineering I*, Lecture Notes in Business Information Processing, pp. 16–30, Springer, Berlin, Germany, 2008.

[11] S. Smanchat, S. Ling, and M. Indrawan, "A survey on context-aware workflow adaptations," in *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM '08)*, pp. 414–417, ACM, November 2008.

[12] S. Rinderle, M. Reichert, and P. Dadam, "Correctness criteria for dynamic changes in workflow systems—a survey," *Data and Knowledge Engineering*, vol. 50, no. 1, pp. 9–34, 2004.

[13] F. Casati, S. Ceri, S. Paraboschi, and G. Pozzi, "Specification and implementation of exceptions in workflow management systems," *ACM Transactions on Database Systems*, vol. 24, no. 3, pp. 405–451, 1999.

[14] S. Krishnan, P. Wagstrom, and G. Von Laszewski, "GSFL: a workflow framework for grid services," Tech. Rep. ANL/MCS-P980-0802, Argonne National Laboratory, 2002.

[15] J. Yu and R. Buyya, "A taxonomy of workflow management systems for Grid computing," *Journal of Grid Computing*, vol. 3, no. 3-4, pp. 171–200, 2005.

[16] G. C. Fox and D. Gannon, "Special issue: workflow in grid systems," *Concurrency Computation Practice and Experience*, vol. 18, no. 10, pp. 1009–1019, 2006.

[17] C. Pautasso and G. Alonso, "Parallel computing patterns for grid workflows," in *Proceedings of the Workshop on Workflows in Support of Large-Scale Science (WORKS '06)*, pp. 1–10, IEEE, June 2006.

[18] F. Lautenbacher and B. Bauer, "A survey on workflow annotation & composition approaches," in *Proceedings of the Workshop*

on Semantic Business Process and Product Lifecycle Management (SemBPM '07), 2007.

[19] M. Wieczorek, A. Hoheisel, and R. Prodan, "Taxonomies of the multi-criteria grid workflow scheduling problem," in *Grid Middleware and Services*, pp. 237–264, Springer, New York, NY, USA, 2008.

[20] J. Yu, R. Buyya, and K. Ramamohanarao, "Workflow scheduling algorithms for grid computing," *Studies in Computational Intelligence*, vol. 146, pp. 173–214, 2008.

[21] J. Chen and Y. Yang, "A taxonomy of grid workflow verification and validation," *Concurrency Computation Practice and Experience*, vol. 20, no. 4, pp. 347–360, 2008.

[22] E. Deelman, "Grids and clouds: making workflow applications work in heterogeneous distributed environments," *International Journal of High Performance Computing Applications*, vol. 24, no. 3, pp. 284–298, 2010.

[23] S. Pandey, D. Karunamoorthy, K. K. Gupta, and R. Buyya, "Megha workflow management system for application workflows," in *Proceedings of the IEEE Science & Engineering Graduate Research Expo*, 2009.

[24] Q. Chen, L. Wang, and Z. Shang, "MRGIS: a MapReduce-enabled high performance workflow system for GIS," in *Proceedings of the 4th IEEE International Conference on eScience (eScience '08)*, pp. 646–651, December 2008.

[25] W. Li, "A Community Cloud oriented workflow system framework and its Scheduling Strategy," in *Proceedings of the 2nd IEEE Symposium on Web Society (SWS '10)*, pp. 316–325, August 2010.

[26] X. Liu, J. Chen, and Y. Yang, *Temporal QOS Management in Scientific Cloud Workflow Systems*, Elsevier, 2012.

[27] S. Pandey, D. Karunamoorthy, and R. Buyya, "Workflow engine for clouds," in *Cloud Computing: Principles and Paradigms*, R. Buyya, J. Broberg, and A. Goscinski, Eds., pp. 321–344, John Wiley & Sons, New York, NY, USA, 2011.

[28] D. Yuan, Y. Yang, X. Liu, G. Zhang, and J. Chen, "A data dependency based strategy for intermediate data storage in scientific cloud workflow systems," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 9, pp. 956–976, 2012.

[29] D. Yuan, Y. Yang, X. Liu, and J. Chen, "On-demand minimum cost benchmarking for intermediate dataset storage in scientific cloud workflow systems," *Journal of Parallel and Distributed Computing*, vol. 71, no. 2, pp. 316–332, 2011.

[30] Z. Wu, X. Liu, Z. Ni, D. Yuan, and Y. Yang, "A market-oriented hierarchical scheduling strategy in cloud workflow systems," *Journal of Supercomputing*, vol. 63, no. 1, pp. 256–293, 2013.

[31] G. Juve, E. Deelman, G. B. Berriman, B. P. Berman, and P. Maechling, "An evaluation of the cost and performance of scientific workflows on amazon EC_2," *Journal of Grid Computing*, vol. 10, no. 1, pp. 5–21, 2012.

[32] A. Verma and S. Kaushal, "Deadline and budget distribution based cost-time optimization workflow scheduling algorithm for cloud," in *Proceedings of the IJCA on International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT '12)*, pp. 1–4, 2012.

[33] J. Behl, T. Distler, F. Heisig, R. Kapitza, and M. Schunter, "Providing fault-tolerant execution of web-service-based workflows within clouds," in *Proceedings of the 2nd International Workshop on Cloud Computing Platforms (CloudCP '12)*, article 7, April 2012.

[34] W. M. P. Van Der Aalst, "The application of Petri nets to workflow management," *Journal of Circuits, Systems and Computers*, vol. 8, no. 1, pp. 21–66, 1998.

[35] J. Zhou and X. Ye, "A flexible control strategy on workflow modeling and enacting," in *Proceedings of the 8th International Conference Advanced Communication Technology (ICACT '06)*, pp. 1712–1716, Republic of Korea, February 2006.