Validation of Pervasive Cloud Task Migration with Colored Petri Net

Lianzhang Zhu, Shouchao Tan, Weishan Zhang*, Yong Wang, and Xiwei Xu

Abstract: Mobile devices are resource-limited, and task migration has become an important and attractive feature of mobile clouds. To validate task migration, we propose a novel approach to the simulation of task migration in a pervasive cloud environment. Our approach is based on Colored Petri Net (CPN). In this research, we expanded the semantics of a CPN and created two task migration models with different task migration policies: one that took account of context information and one that did not. We evaluated the two models using CPN-based simulation and analyzed their task migration accessibility, integrity during the migration process, reliability, and the stability of the pervasive cloud system after task migration. The energy consumption and costs of the two models were also investigated. Our results suggest that CPN with context sensing task migration can minimize energy consumption while preserving good overall performance.

Key words: colored Petri net; task migration; pervasive cloud; context information; validation

1 Introduction

In mobile cloud computing, the resource limitations of mobile devices sometimes mean that their computing and storage capabilities are insufficient for task execution[1]. Task migration has therefore become an extremely important challenge. Many researchers have investigated mobile cloud computing or cloud computing task migration[2, 3].

As early as 1988, Schwederski et al.[2] conducted research on a task migration model in a parallel processing system. They focused on task migration costs, which consist of directed and undirected overheads and migration penalties. The directed overhead encompasses the cost of moving a task when no other tasks affect the migration, while the undirected overhead is the additional time needed for migration due to the influence of other tasks. A migration penalty may be incurred when the migration affects other tasks. The authors proposed a model for parameterizing migration costs in a parallel processing system. A prototype that enabled adaptive application task migration in a Grid environment was presented[4]. The prototype automatically reconfigures distributed applications in response to network performance failures and Denial of Service (DoS) attacks. The use of performance monitoring software enables network connection failover and automatic application task migration within a heterogeneous distributed computing environment. Ma and Wang[3] proposed a Java bytecode transformation technique for realizing task migration without imposing significant overheads on normal execution. An asynchronous migration technique allows migrations to take place virtually anywhere in the user codes, and the proposed Twin Method Hierarchy minimizes the overhead resulting from state-restoration codes in normal execution.

Buchbinder et al.[5] proposed a novel Efficient Online Algorithm to solve the migration problem of batch jobs between datacenters. The algorithm has two distinctive
features: it considers future variability and uncertainty of energy sources, and it can handle the fundamental tradeoff between energy and the bandwidth costs of migrating application data and states.

Most existing work on task migration focuses on identifying ways of achieving task migration by constructing models or infrastructure. Other works have focused on migration policies and deriving algorithms that can solve certain task migration challenges. To the best of our knowledge, no analysis has been made of the following topics: when to start task migration, the integrity of task migration, or the reliability and stability of a mobile cloud platform after task migration. These issues are key to task migration in mobile cloud computing, as they help determine the appropriateness, feasibility, and performance of task migration. In this paper, we propose a new approach to validation of task migration in a mobile cloud. We expanded the semantics of the Colored Petri Net (CPN) to include description of contexts and the relationships between the CPU usage, memory usage rate, and network bandwidth of the system. For task migration, we created two task migration models with different task migration rules based on the semantic expanded CPN and drawing on the concept and composition of the OSGi-based pervasive cloud platform. We then simulated and verified the two models of task migration. Based on the data obtained from simulation, we undertook a comprehensive evaluation, including: task migration accessibility, integrity of the task migration process, and the reliability and stability of the mobile cloud system after task migration. We also compared the energy consumption of the two models.

The rest of paper is organized as follows. Section 2 presents some preliminaries, including a CPN description and definition, one CPN ML tool, an overview of task migration, and the OSGi-PC pervasive cloud infrastructure. Section 3 extends the semantic of the original CPN and gives a detailed description of our CPN-based task migration models. In Section 4, we introduce the simulation of the proposed migration models with CPN Tools. Evaluation and analysis of our proposed task migration models are given in Section 5. The conclusions and plans for future work in Section 6 end the paper.

2 Preliminaries

2.1 CPN

CPN is a language developed by Jensen\[6]. It is an extension of the basic Petri Net (PN), which is used to model and analyze parallel systems. CPN uses the same model elements as PN, but adds properties to distinguish between different individuals of the same type of a system factor\[7]. The properties of the model elements are referred to as “colors”. A formal definition of CPN is as follows:

\[
\text{CPN} = (\Sigma, P, T, A, N, C, G, E, I),
\]

where

(1) \( \Sigma \) is a finite set of data types that is called a color set;
(2) \( P \) is a finite set of places;
(3) \( T \) is a finite set of transitions;
(4) \( A \) is a finite set of arcs, and \( P \cap T = P \cap A = T \cap A = \emptyset \);
(5) \( N \) is a node set function, \( N : A \rightarrow P \times T \cup T \times P \);
(6) \( C \) is a color set function that assigns a color set to one place, \( C : P \rightarrow \Sigma \);
(7) \( G \) is a guard function or transition expression function that is defined as \( T \rightarrow \text{Expression} \);
(8) \( E \) is an arc expression function that assigns an arc expression to each arc. The definition is \( E : A \rightarrow \text{Expression} \); and
(9) \( I \) is an initialization function that assigns an initialization expression to each place.

CPN differs from PN in two ways: all places and transitions have color attributes, and the input and output are matrices, whereas in PN, they are scalars. In CPN, a token may have a complex data type, as in a programming language, and transitions can process the values of the received tokens and create new ones, which can be of different data types. CPN has better modeling than PN.

2.2 CPN tools

CPN is supported by a standard ML\[8] with a CPN ML tool, which is an extension of the ML language\[9]. This tool can be used to create a CPN model and allocate color sets, variables, and functions. Most importantly, functions can be used at arc expression and guards in the CPN model.

CPN Tools can help to construct places, transitions, and functions and automatically check grammar. It includes a simulation tool, a state space tool, and a constructing tool. Users can run simulations of the system and analyze the state space using CPN Tools. After the simulation, we obtain the step number, time, and statement of the system model.

Figure 1 shows samples of the ML syntax used in
CPN Tools. A color set is declared using the key word “colset”, whereas a variable is declared using the key word “var”. At the end of the figure, there are two functions that are declared using the keyword “fun”.

The CPN Tools version 4.0 interface is shown in Fig. 2.

### 2.2.1 Simulation tool of CPN Tools 4.0

When a model is constructed, CPN Tools 4.0 will automatically check grammar. If no errors are found, the simulation can be executed. The simulation tool is shown in Fig. 3. We can use it to rewind, stop, and debug the simulation and to produce a simulation report with the correct setting to index options.

### 2.2.2 State space tool of CPN Tools 4.0

A state space tool can help analyze the states of the system model. A label can be created for each accessible node. All binding elements from the start to end nodes form a state space graph. The state space tool can calculate the state space of the model, produce a connected graph of the state space, and output an attribute report to help analyze the accessibility, boundedness, activity, and fairness of the model. This allows the performance of the model to be evaluated.

### 2.3 Task migration and OSGi-based pervasive cloud

Resource-limited devices or systems often need to offload computational or resource-intensive tasks to cloud nodes\cite{10}. This is called task migration. In this research, we used an OSGi-based pervasive cloud (OSGi-PC) infrastructure as the mobile cloud platform because OSGi-PC\cite{11} can exploit both the cloud computing capabilities and component flexibility of OSGi\cite{12}. It is easy to model task migrations based on the concepts and structure of OSGi-PC, and OSGi-based frameworks are widely used for both small devices and powerful cloud nodes. All these services follow the same OSGi standard, which offers inter-operability, support for dynamic deployment and replacement of OSGi services, and scalability when new services are added. An overview of OSGi-PC is presented in Fig. 4. Using the coordination of the discovery component, both remote cloud service frameworks and local client service frameworks can be accessed.
3 CPN-Based Task Migration Model

In an actual pervasive cloud, the task migration system is very complicated because it involves mobile components, servers, virtual machines, networks, and others. Thus, before constructing the task migration system model, we made assumptions as follows.

1. All entity information of task data packages delivered among virtual machines through the network is related to task type.

2. When assessing system performance, only virtual machines in servers and mobile devices need to be considered. It is unnecessary to pay attention to the implementation of the bottom protocol.

As shown in Fig. 5, the overall working flow of our validation approach is as follows: obtain context information from OSGi-PC, expand the semantics of CPN, construct the task migration model by applying two migration policies, and finally, simulate the two models and compare their performance.

3.1 CCPN: CPN with context sensing

Task migration needs to integrate context information, but CPN has weak semantics that cannot support this. Therefore, we introduced a semantically expanded CPN called CPN with context sensing (CCPN). We used CCPN to integrate context with our task migration model.

A formal definition of CCPN is as follows:

\[
\text{CCPN} = (\Sigma, P, T, F, F_i, M_0),
\]

where

\( \Sigma \) is a set of data types, also called color sets;
\( P = P_c \cup P_t \) when \( P_c \) is a finite set of context places and \( P_t \) is a finite set of temporary places;
\( T = T_i \cup T_e \) when \( T_i \) is a finite set of internal transitions and \( T_e \) is a finite set of external transitions;
\( F \) is a finite set of arcs, and \( P \cap T = P \cap F = T \cap F = \emptyset \);
\( F_i \) is a finite set of restraining arcs; and
\( M_0 \) is the initial state.

For clarity, Fig. 6 presents context places, temporary places, internal transitions, internal clean transitions, and external transitions in CCPN.

3.2 CCPN_TM: CCPN-based task migration model

Using semantically expanded CCPN, we can construct a task migration model called CCPN_TM to model task migration in pervasive cloud computing, taking OSGi-PC as the platform.

\[
\text{CCPN}_{\text{TM}} = (P_{\text{host}}, P_{\text{vm}}, P_{\text{mobile}}, T, \Sigma_{\text{cloud}}, R),
\]

where

1. \( P_{\text{host}} \) represents the servers;
2. \( P_{\text{vm}} \) represents virtual machines in the servers;
3. \( P_{\text{mobile}} \) represents mobiles in pervasive computing;
4. \( T \) represents all actions in a pervasive computing environment;
5. \( \Sigma_{\text{cloud}} \) represents all color sets in this model; and
6. \( R \) represents two task migration policies: one with no context, and one with context information.

Figure 7 shows the CCPN-based task migration model.
model, where the solid-line arrows indicate pervasive cloud servers that are accessible and the dashed-line arrows indicate task migration policies among servers.

### 3.2.1 Task migration policies

Figure 8 shows the first model without context information. In this approach, task migration is determined by state of virtual machines where the tasks are located. There are two state values: “task\textsubscript{in}” and “task\textsubscript{out}”. Only a virtual machine whose “task\textsubscript{out}” value is “true” can migrate its tasks out to other virtual machines, and only a virtual machine whose “task\textsubscript{in}” is “true” can accept tasks from other virtual machines. The two state values are declared in the VirtualMachine color set. The functions “TaskMigrationOut” and “TaskMigrationIn” are used to change the state of the virtual machines.

The second task migration approach integrates the context information, as shown in Fig. 9. In contrast to the approach without context, this policy makes migration decisions based on not only the two state values of the virtual machines but also the context information, which includes the CPU, RAM, and bandwidth of the machines. A function named “fusioncontext” determines whether the context of a virtual machine is suitable to task migration.

The CCPN\_TM model uses many data types, the most important of which are “Host”, “Mobile”, “VirtualMachine”, and “Task”. Our color sets were defined as shown in Fig. 10. “Host” covers many attributes such as CPU, memory, and energy consumption. Many virtual machines may run on a single host, and each virtual machine may perform multiple tasks. The key attributes of virtual machines include CPU, memory, bandwidth, energy consumption, costs, and task migration policies. “Mobile” covers similar attributes to “Host” while whereas “Task” covers attributes such as size, state, and decision of whether to migrate.

### 3.2.2 Task execution and migration target selection

We used the function “process” to deal with task execution in the original virtual machine and after migration to another virtual machine, as shown in Fig. 11.

When a migration policy is selected, another important aspect of task migration is the selection of a new virtual machine for tasks that need to be migrated. We selected the migration-in virtual machine using the match degree, as shown in Fig. 12. We describe this selection algorithm in Section 4.

### 4 Simulation to CCPN\_TM with CPN Tools 4.0

To evaluate and analyze the task migration model described in Section 3, we implemented and simulated the models using our semantic expanded CCPN, as defined in Section 3.1.

#### 4.1 Top level module of task migration

The top module of task migration model was built with CPN Tools 4.0, and is shown in Fig. 13. This module included a Mobile Center module, a Net Transmit module, and a Data Center Broker module, described with substitution transition. The Mobile Center module made the preparations for task migration, and then transferred the task to the Net Transmit module, which provided the net transmit functions. Tasks to
be migrated were passed to the Data Center Broker for execution. When the task execution completed, it was transmitted to place “Result”. If further migration was needed, it was migrated to place “ReMigrate”, and then transmitted to the Data Center Broker through the “ProceessTask” transition.

In the task migration model, substitution transitions and places were connected with directed arcs, representing the data transmission channel. When the task migration system was initiated, virtual machines prepared the tasks that needed to be migrated. The tasks were then transferred to the “UserTask” list in the output place. Thereafter, CPN Tools sent the tasks to the Net Transmit module, which then selected suitable virtual machines, executed the tasks, and determined whether the current tasks needed secondary migration.

4.2 Mobile Center module

The Mobile Center module is shown in Fig. 14. It was mainly used to trigger the execution of the model, do pre-processing of the tasks in the mobiles, and transfer the tasks awaiting migration to the “UserTasks” place.

To illustrate the preparation needed before task migration, an example in which each cluster contains only two or three mobiles is shown in Fig. 15. Tasks need to be migrated when the utilization ratio of
Fig. 14 Mobile Center module of task migration with CPN Tools 4.0.

Fig. 15 Pre-process of tasks to be migrated in mobile device.

CPU and RAM in the mobiles is high. The status of unfinished tasks is changed from “Running” to “Suspend”. All the tasks with “Suspend” status are transferred to the “UserTask” place to await migration.

4.3 Net Transmit module

The Net Transmit module (Fig. 16) had two input variables “n” and “transmit”, which were used to control the transfer order of the task data packets. Output place “A” was used to migrate the tasks to the Data Center Broker module, while output place “C” was used to determine whether the Data Center Broker would take the next packet.

4.4 Data Center Broker module

After the tasks to be migrated were transferred from the
Net Transmit module to the Data Center Broker module, the Broker sent the tasks to the data centers. The centers then selected the most suitable virtual machine for the task based on the machine’s CPU and RAM. If the task execution was completed in the Process module, it was sent to the “Result” place, whereas tasks needing secondary migration were sent to the Data Center Broker again through the “ReMigrate” place. This process is as shown in Fig. 17. We can see that this module had three important functions. The first was target selection, which we discuss in Section 3.2.2. The second was the task execution. The third function applied context information. We now discuss these three functions.

In a pervasive cloud environment, there are many virtual machines in each datacenter. A key challenge is the selection of the most suitable virtual machine to complete the tasks with minimum cost and maximum resource release. If more than one task needs to be migrated, and the best virtual machines for these tasks are selected simultaneously, a clustering effect may be incurred. To avoid this, whilst balancing the workload of the virtual machines, we selected the target node based on probabilities, rather than by migrating tasks to the virtual machine with the best performance. CPU and RAM were considered in target selection, as follows:

1. Select $n$ targets from the pervasive cloud using the matching degree $UR_{available}$ of the target virtual machines and the $UR_{cost}$ of the task.
2. Construct a probability model based on the $R_{available}$ value of the $n$ targets.

The probability that the virtual machine would accept the task was then defined by Formula (1).

$$p_i = UR_i / (\sum_i UR_i)$$  \hspace{1cm} (1)

where $UR_i = C_i \times R_i$, while $C_i$ represents the utilization rate of CPU and $R_i$ represents the utilization rate of RAM in the virtual machines.

3. Randomly generate a number $[0, 1]$ and compare it with the probability value in the second step to determine the task migration target.

A task migration model without context determines whether to migrate based only on the state of the virtual machines. In contrast, our context fusion model applied statistical methods to the CPU utilization, RAM utilization, and bandwidth of the virtual machine in runtime, improving the accuracy of task migration. The context information was saved in the “CPURAMBW” place, as shown in Fig. 18. The function “judge” determined if the virtual machine could provide sufficient CPU, RAM, and bandwidth to support the execution of tasks, as shown in Fig. 19.

5 Evaluation and Analysis

In the previous sections, we have described the task migration models and established a multi-level task migration model using CPN Tools. We then wished to confirm the correctness of the model by showing that the model could describe the task migration system correctly. In this section, we present an analysis of task migration by our two different migration models in a pervasive cloud, using a combination of state space and monitor in CPN Tools 4.0.

5.1 Model evaluation

The task migration model built with CPN Tools 4.0 and based on CCPN provided the basis for demonstrating the logic and effectiveness of task migration. The simulation result report and state space report from the simulation were used to construct a state space graph. This allowed analysis of the features of the task migration model.
A partial simulation report is as shown in Fig. 20.
From the whole simulation report, we could check that the number of tokens in each place did not exceed the initial task number, proving that the model was bounded. We conducted simulations of the task migration model CCPN\_TM, and demonstrated that the model could complete its execution in finite steps, which confirmed the activity of our model. As can be seen from Fig. 21, there were 12 tokens in the place “Result”. This is the correct output, which indicates that our model is performed correctly.

5.2 State space analysis—Accessibility and activity
The state space report was used to demonstrate the accessibility and activity of the CCPN\_TM model. A partial state space report is shown in Fig. 22.
Figure 23 shows part of the state accessibility graph of task migration, with a detailed description of each node. Each node in the state accessibility graph represents an arrival identification, and each directed arc represents a binding element from source node to target node. From the figure we can check that the virtual machines are able to access each other in a task migration system. The task migration routes also confirmed the accessibility of our task migration model.

5.3 Energy consumption analysis
We declared the “Host” color set in the CCPN\_TM task migration model, and in this color set the energy consumption was defined. Using simulation,
we could obtain the energy consumption of servers in the pervasive cloud. Before analyzing the energy consumption of task migration, we first made the following assumptions, following earlier studies [13, 14].

(1) Task migration from one virtual machine to another will generate costs that reflect the size of the task and the bandwidth. Migration time is the main computing parameter in the task migration cost, and can be calculated using Formulas (2) and (3).

\[
\text{Migration Time} = \frac{\text{Task size}}{\text{Bandwidth available}} \quad (2)
\]

(2) Estimation of the energy consumption of servers defined in the color set “Host” can be calculated by the energy consumption model defined with the following Formula (3):

\[
E_i = \min E_i + \text{CPU}_{\text{Sage}} \times \frac{(\max E_i - \min E_i) + (\Sigma \text{Migration Time}) \times \text{Power}}{} \quad (3)
\]

where \(E_i\) represents consumption (per hour in Watts) for the servers, \(\text{CPU}_{\text{Sage}}\) is the average value of the utilization rate of the virtual machines running in the servers, \(\max E_i\) represents the maximum energy consumption when the server is at maximum load, and \(\min E_i\) represents the energy consumption when the server is in the idle state. Power represents the energy consumption value of the unit time to transfer tasks. \(\Sigma \text{Migration Time}\) represents the total time consumption of all task migration in the server.

(3) Task migration is without delay. Task execution cost was represented by \(E_{\text{Cost}}\), calculated using Formula (4), and the cost of task migration between virtual machines was represented by \(T_{\text{Cost}}\), calculated using Formula (5).

\[
E_{\text{Cost}} = \left(\Sigma_{\text{vm}} \text{EC}\right) \times \text{Price}(P) \quad (4)
\]

\[
T_{\text{Cost}} = \Sigma_{\text{vm}} \text{Migration Time} \times \text{Price(SLA}_{\text{min}}) \quad (5)
\]

In Formula (4), \(\text{Price}(P)\) represents the equipment price in the cloud environment, and \(\text{EC}\) represents the virtual machine energy consumption. In Formula (5), \(\text{Price(SLA}_{\text{min}})\) represents the lowest price of the Service Level agreement and is usually fixed in a pervasive cloud. \(\Sigma_{\text{vm}} \text{Migration Time}\) means the total migration time for migrating tasks from one virtual machine to others in the server.

These formulas gave the energy consumption of task migration. To improve accuracy, we averaged energy consumption, execution cost, and transmission cost from multiple simulations.

We selected five hosts in our pervasive cloud environment. The average power consumption of the two task migration models is shown in Fig. 24. Comparison of the two lines showed that the energy consumption of the migration model without context information was greater than that of the context sensing.
task migration model. This is because the latter model factored in the state and current context of the virtual machine, allowing the virtual machine with the most appropriate CPU and RAM to process the task to be selected, while the former considered only the state of the virtual machine.

The average execution costs of the two task migration models across the five hosts are shown in Fig. 25. In most cases the context sensing task migration model achieved a lower cost, because it considered the performance of the virtual machines when selecting the migration target.

Although the task migration model with context sensing reduced energy consumption, it could have higher transmission costs (Fig. 26).

### 6 Conclusions and Future Work

In this paper, we proposed a context enabled CPN approach to validating task migration in a pervasive cloud environment. Our research evaluated task migration accessibility, integrity during the task migration process, and the energy consumption of the mobile cloud system during task migration. We analyzed the CPU, RAM, and bandwidth of both hosts and virtual machines, in terms of task size, task
state, and other contexts. We built two CCPN-based task migration models with different task migration policies. CPN Tools 4.0 was then used to evaluate the activity and accuracy of the systems, and to analyze the energy consumption and costs of the two models. The simulation results suggest that CCPN-based task migration can combine energy consumption minimization with good overall performance.

In future research we will also consider the time parameter of task execution, to determine whether task migration should be done or not. We currently select the target virtual machine for task migration by matching virtual machines and tasks, but in real applications, the location of the virtual machines is also an important selection factor. Adding this to the selection function could further improve the accuracy of the task migration model, while reducing energy consumption and migration costs.

References

[1] M. D. Kristensen, Empowering mobile devices through cyber foraging, PhD dissertation, Department of Computer Science, Aarhus University, 2010.

[2] T. Schwederski, H. J. Siegel, and T. L. Casavant, A model of task migration in partitionable parallel processing systems, in Frontiers of Massively Parallel Computation, 1988, Proceedings, 2nd Symposium on the Frontiers of,

Fig. 26 Average transmission costs of two task migration models.

Lianzhang Zhu is a professor in China University of Petroleum. He graduated from East China Petroleum Institute, and received a bachelor degree in science. In 1988, he graduated from Shenyang Institute of computing technology, Chinese Academy of Sciences with a master’s degree in engineering. He got his PhD degree in mineral and general survey from China University of Petroleum in 2003. He has visited Science and Technology University of Missouri. His main research interests are software performance evaluation, software testing, oil field data model, etc.

[3] R. K. K. Ma and C. L. Wang, Lightweight application-level task migration for mobile cloud computing, in Advanced Information Networking and Applications (AINA), 2012 IEEE 26th International Conference on, Fukuoka, Japan, 2012, pp. 550–557.

[4] S. Frechette and D. R. Avresky, Method for task migration in grid environments, in Network Computing and Applications, Fourth IEEE International Symposium on, Cambridge, MA, USA, 2005, pp. 49–58.

[5] N. Buchbinder, N. Jain, and I. Menache, Online job-migration for reducing the electricity bill in the cloud, in NETWORKING 2011 Springer. Springer, 2011, pp. 172–185.

[6] K. Jensen, Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Springer Science & Business Media, 2013.

[7] Z. Hu, T. Lu, and Z. Zhao, Context-aware service system modeling using timed CPN, in Service Systems and Service Management (ICSSSM), 2013 10th International Conference on, Hong Kong, China, 2013. pp. 164–169.

[8] A. W. Appel and D. B. MacQueen, Standard ML of new jersey, in Programming Language Implementation and Logic Programming. Springer Berlin Heidelberg, 1991, pp. 1–13.

[9] A. V. Ratzer, L. Wells, H. M. Lassen, M. Laursen, J. F. Ovortrup, M. S. Stissing, M. Westergaard, S. Christensen, and K. Jensen, CPN tools for editing, simulating, and analysing coloured Petri nets, in Applications and Theory of Petri Nets 2003. Springer Berlin Heidelberg, 2003, pp. 450–462.

[10] M. Shiraz and A. Gani, A lightweight active service migration framework for computational offloading in mobile cloud computing, The Journal of Supercomputing, vol. 68, no. 2, pp. 978–995, 2014.

[11] W. S. Zhang, L. C. Chen, X. Liu, Q. H. Lu, P. Y. Zhang, and Y. Su, An OSGi based adaptive and flexible pervasive cloud infrastructure, Science China Information Science, vol. 53, no. 3, pp. 1–11, 2014.

[12] O. S. G. Alliance, OSGi—The dynamic module system for Java, http://www.osgi.org/, 2009.

[13] M. M. Alansari and B. Bordbar, An architectural framework for enforcing energy management policies in cloud, in 2013 IEEE Sixth International Conference on Cloud Computing, Santa Clara, CA, USA, 2013, pp. 717–724.

[14] P. Aziz Abdulla and R. Mayr, Petri nets with time and cost, arXiv preprint arXiv: 1302.3291, 2013.
Shouchao Tan is a master student working on big data processing and software architecture. She graduated from China University of Petroleum with a bachelor degree in 2014. She has published 3 papers on pervasive cloud computing and decision making for task migration in pervasive cloud.

Weishan Zhang is a full professor, deputy head for research of Department of Software Engineering, School of Computer and Communication Engineering, China University of Petroleum. He got his PhD degree from Northwestern Polytechnical University in 2001. His research interests include big data processing, Internet of things, and cloud computing. He is the director of “Big Data Intelligent Processing Innovation Team of Huangdao District”, director of “Smart City Research Center of Fuwode Electronic Corp.”, and founding director of the Autonomous Service Systems Lab. He was a research associate professor/senior researcher at Computer Science Department, University of Aarhus, where he had been working on the EU FP6 Hydra pervasive middleware project, for which he was a technical manager (Aug. 2008–Dec. 2009). He was a visiting scholar of Department of Systems and Computer Engineering, Carleton University, Canada (Jan. 2006–Jan. 2007). He was an associate professor at School of Software Engineering, Tongji University, Shanghai, China (Aug. 2003–June 2007). He was a NSTB post-doctoral research fellow at Department of Computer Science, National University of Singapore (Sept. 2001 to Aug. 2003). His current h-index according to google scholar is 13 and the number of total citations is around 530 in Sept. 2015.

Yong Wang is a master student working on modeling and validation for cloud environment using colored Petri net. He graduated from Nanjing University of Chinese Medicine with his bachelor degree in 2012.

Xiwei Xu is a researcher in the Software Systems Research Group at NICTA. Her research interests include software architecture, dependable distributed systems, service/cloud engineering especially RESTful services, and business process management and she is currently involved in the Dependable Cloud Computing Project and the Security Architecture Project. She got her bachelor degree from Nankai University in 2007 and PhD degree from The University of New South Wales in 2012. She has been in an 7-month internship experience in Aragon Consulting Group (Nov. 2006–May 2007) focusing on web 2.0 development/AJAX.