International Journal of Computational Intelligence Systems, Vol. 7, No. 3 (June 2014), 493-505

A Component-based Intelligent Seamless Service Migration Mechanism and Flexible Communication Protocol in Pervasive Computing Systems*

Haibin Cai*
Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, 200062 Shanghai, China
State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, China
E-mail: hbcai@sei.ecnu.edu.cn

Chao Peng
Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, 200062 Shanghai, China

Yue Zhang
Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, 200062 Shanghai, China

Linhua Jiang*†
CCSR, Stanford University, Stanford, CA 94305, USA
E-mail: lhjiang@stanford.edu

Received 8 July 2012

Accepted 16 April 2013

Abstract

In pervasive computing systems, service selection and fault tolerance are two primary factors, which determines reliability and efficient of service selection system. In this paper, according on ANN-based (Artificial Neural Network) service selection model, we extend the definition and formal description of service, and propose the architecture of service migration system, evaluation method, judgment theorems of triggering and a service-oriented seamless migration (SOSM) algorithm in pervasive computing systems. We also propose the flexible communication protocol for service seamless migration activity based on mobile agent computing network. We have implemented the SOSM mechanism and flexible communication protocol in an intelligent service selection prototype (ISSPS) system, which provide the pervasive web services for users in pervasive computing environment. The results show that the proposed service migration mechanism is not only reliable but also efficient.

Keywords: Pervasive computing, Service failure, Service migration, Communication Protocol

1. Introduction

The dawn of the 21st century has seen explosive interest in Pervasive/Ubiquitous Computing. Coming roughly a decade after founding manifesto by Mark Weiser1, much of this research addresses issues that may seem exotic relative to mainstream computing. Promoters of this idea hope pervasive computing embedding the integration of computer, communication and digital media technology makes it possible to integrate the physical world we are living in and the virtual world in the information space together as the whole, which would enable people to move around and interact with computers more naturally than they currently do [2].

The service migration in pervasive computing environments is one kind of the process migrations. It is an operating system capability that allows a running service program or date to be paused, relocated to

*Corresponding author. Address: Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, 200062 Shanghai, China. Email: hbcai@sei.ecnu.edu.cn; CCSR, Stanford University, Stanford, CA 94305, USA, Email: lhjiang@stanford.edu

another node of network, and continued there. It represents seamless migration at the granularity of individual processes, and has been a research focus on many experimental operating systems. Examples include Charlotte [3], Sprite [4], Demos [5], V [6], Condor [7], and Mach [8]. These independent validation efforts have shown beyond reasonable doubt that process migration can indeed be implemented with acceptable efficiency. The service migration must implement seamless migration, namely, implementing seamless computing. One of the requirements of seamless migration is communication network and device independence which allows users to not only move freely between heterogeneous networks, but also change computing devices, if necessary, while maintaining application continuity. Because pervasive computing environment is heterogeneous network environment, the issue is particularly difficult for service selection system which has quite stringent requirements with QoS [20, 21], delay [22], jitter [23] and performance efficiency [24, 25]. In a dynamic and heterogeneous environment (e.g. due to disconnections or mobility etc.), the network service provided to users may change and may no longer meet the application requirements. This needs to research the problem of service seamless migration by adapting the networking environment, or the application, or its communication streams, or using a combination of these different types of adaptations. Several mechanisms of migration for distributed computing [22, 23] or grid computing [20, 21] environment have been suggested by some researchers. But these can not meet the seamlessness requirements of service-oriented migration of pervasive computing paradigm, because pervasive computing environment have some different traits from the traditional distributed computing environment and gird computing environment. Therefore, in this paper, we firstly discuss the extension definition and formal description of service, then and propose architecture of service selection system and architecture of service migration system. We adopt a novel ANN-based (Artificial Neural Network) service selection model (shortly called the ANNSS model) to select optimal service for user in pervasive computing environment using an improved artificial neural network algorithm [19]. We put forward the architecture of service migration system and a service-oriented intelligent seamless migration algorithm (shortly called SOSM Algorithm) in pervasive computing systems. The idea is that when the service failure happened, a service migration activity should be trigged. We implemented the service migration mechanism in an intelligent service selection prototype system (shortly called ISSPS), which provide the pervasive web services for users in a pervasive computing environment. Namely, the ANN-based service selection mechanism and service-oriented seamless migration algorithm are implemented in this system. We tested our approaches through extensive simulations and obtained promising experimental results. The results of simulation show that the proposed service migration scheme is not only reliable but also efficient.

This paper is organized as follows: In Section 2, we describe extended definition and formal description of service. In Section 3, we explain the architecture of architecture of service migration system. In Section 4, we show the service-oriented intelligent seamless migration algorithm and flexible communication protocol. We also describe the evaluation method and judgment theorems of triggering a service migration activity in detail. Section 5 describes the simulation experiment and discusses our simulation results. Finally, we present conclusions that can be drawn from this work and possible directions for further investigation.

2. Extended Definition and Formal Description Service

Within the service architecture for pervasive computing environments, service is regarded as service component which is the most fundamental and important conception. Services can be implemented as either hardware devices, software programs, or a combination of the two, and can be found by human and computational clients. The definition of service in existed distributed or gird system is appropriate for pervasive computing systems no longer because:

(a) The connotation of service is different from each other. In pervasive computing systems, a service is an entity that can be used by a person, a program or another service. A service may be a computation, storage, a communication channel to another user, a software filter, a hardware device or another user;

(b) In pervasive computing systems, all kind of attributes of service are varying according to the context information;

(c) The description approach for service is different from the description approach of traditional distributed systems.

So we expand the traditional definition of service [12, 13] to a new service definition by adding new items to service definition for pervasive computing environments.

DEFINITION 1: Service is defined as an entity which can provide certain functions and be composed of a set of atomic services AS_j $(0 \leq j \leq n)$ which is a description for certain subunit with self-governed function. A service can be used by a person, a program or another service. A service may be a computation, storage, a communication channel to another user, a software filter, a hardware device or another user.

Our formalized definition of service can be expressed as:

$$S_i = \{AS_j < ID, T, K, P, V, C, Z, A > | 0 \leq j \leq n\} \quad (1)$$

where S_i denotes one of available services in current service system, AS_j denotes atomic services and S_i is composed of AS_j, ID represents the global service identifiers in order to ensure uniqueness in the system, T represents service type, K represents keywords for matching search/lookup, P represents the set of service attributes which describe characteristics of service, V represents the value of respective attributes, C represents the set of context information including user context ,computing context, physical context and time context, Z represents the evaluation value of respective QoS of service providers by the neural network controller estimating and predicting, and A represents the address of service interfaces which include the methods that users and applications will invoke to execute the service along with any other descriptive attributes.

3. Architecture of Service Migration System

In this section, we give the architecture of service migration system based on our ANN-based service selection model (shortly called ANNSS), for details, please see our related work [16]. After the service selection system selected an appropriate service provider, the client or user invoke the selected service. Because context environments are changing, it results in service failure. Under the conditions we must use a service migration system to migration the process and data of the service from the source service provider to a new service provider with the same or similar function.

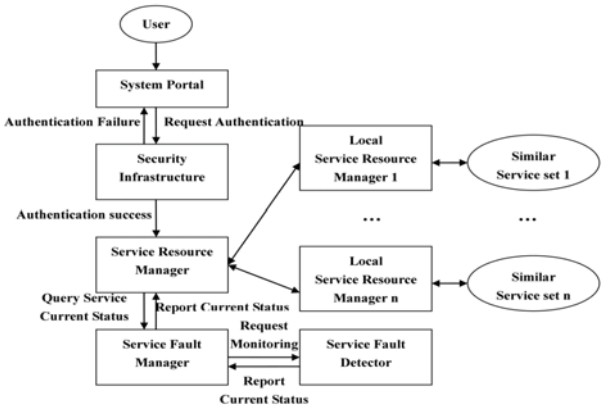

Fig. 1. Architecture of service migration system.

In Figure 1, the architecture of our service migration system consists of system portal, security infrastructure, global and local service resource manager, service fault manager, service fault detector and similar service set. Because our service selection system eliminates user participation and selects the optimal service from similar service set using an intelligent service selection algorithm based on artificial neural network, it overcomes the shortcomings of blindness and randomicity in traditional and improved trust-mechanism-based service selection models and improves the system performance. The service resource manager cooperates with a system portal, security infrastructure, local service resource manager and service fault manager. The system portal provides an interface for a user to launch a service or application that will utilize the service resources provided by the pervasive computing system. The security infrastructure is responsible for the dynamic identity authentication based on attributes and the dynamic access authorization control based on role. To execute the job with the service resource manager, a user's service request should describe a service resource type, a service resource condition, and the number of resources using above USDL. The USDL parser extracts the service resource type and condition and sends them to the service resource manager. The service resource manager is responsible for (a) the discovery of the service resource that satisfies the requirements, (b) the service migration when the service provider is under the conditions of service fault, (c) discovery the most appropriate destination of service migration and look for

the optimization route from source to destination node, (d) generation of migration agents. The local service resource manager is responsible for managing the service resource with similar function in respective similar service set. The fault detector is responsible for detecting a service resource failure. If the service resource failure or system performance degradation occurs, the fault detector sends an alert message to a fault manager. If the fault manager received an alert message, it should decide whether a service migration should occur or not. If it decides to do a service migration, it requests to allocate newly selected service resource and restarts execution using a checkpoint.

In pervasive computing, one of key ability of service migration system is to detect fault based on components of service fault detector and manage fault based on components of fault manager. Our service fault detector provides five kinds of services. A monitoring service monitors resource states of processes, processors, network, service delay and over-load. A communication service provides communication with each component. A fault detection service decides the failure occurrence for each service resource. For fault detection service, the service fault detector consists of a process monitor agent, a processor monitor agent, a network monitor agent, a service delay monitor agent, a service over-load monitor agent, a fault decision agent, and a communication coordinating agent.

4. SOSM Mechanism and Flexible Communication Protocol

4.1. *Service migration*

We put forward the service-oriented seamless migration mechanism (shortly called SOSM mechanism) for pervasive computing environments so that the problem of service fault tolerance can be resolved when certain factors result in service failure. The process failure, hardware crash, network failures, system performance degradation, communication delay, service delay failure, over-workload failure, and addition of new service resource dynamically change the pervasive computing environments. In pervasive computing environments, the service migration is the only efficient way to guarantee that the submitted service request is completed reliably and efficiently even though service resource failure occurs. Service fault tolerance is achieved by storing the states of processes periodically on a stable storage during failure-free execution. A stable storage must ensure that the recovery data persists during the period when the tolerated service failures occur and their corresponding recoveries are finished. In a system that tolerates an arbitrary number of service failures, a stable storage may consist of local service resource in each service provider or a persistent medium outside the service provider where a service program is running. Upon a failure, a failed service restarts from one of its saved states, thereby reducing the amount of lost computation. Each of the saved states is called a checkpoint [9].

If a service stop failure occurs, a rescheduling agent performs rescheduling for service migration unconditionally and a service migration agent performs service migration using a checkpoint. But if a service resource QoS failure occurs, a rescheduling agent must decide whether service migration occurs or not. Because of the service migration overhead (service-migration-overhead, α), the service migration may give rise to increase of the execution time. So if a service resource QoS failure occurs, the rescheduling agent evaluates the performance benefit that can be obtained due to service migration. The rescheduling agent requests the execution time predictor to calculate the remaining execution time (remaining-execution-time, β) of the service program if it is to continue execution on the original service node and the new execution time (new-execution-time, ζ) of the service program if it is to execute on the new selected service node (namely target node of service migration) in the reschedule. The performance benefit (performance-benefit, η) is as follows:

$$\eta = \frac{\beta - (\zeta + \alpha)}{\beta} \quad (2)$$

The service-migration-overhead is as follows:

$$\alpha = t_{rescheduling} + t_{writing-checkpoint} + t_{transfer} + t_{stating-application} + t_{reading-checkpoint} \quad (3)$$

If the performance benefit is grater than a migration threshold, the service migration may give rise to increase of the execution time and the performance may decrease due to the service-migration-overhead if a service failure happen and then service migration occurs. The migration threshold can be determined by a user.

4.2. *Evaluation method and judgment theorems of triggering and completing service migration activity*

In order to describe our evaluation method and triggering mechanism of service migration activity, we must first present some definitions for notational convenience.

DEFINITION 2: Let U be a set that represent all service resource set consisting of all service provider in pervasive computing network, and let C be a set of subsets of U. We say that C is a similar service set under U if the following conditions hold:

(a) Nontrivial Property. $G \in C \rightarrow G \neq \emptyset$.

(b) Nonnull Intersection Property. $\forall G, H \in C, G \cap H \neq \emptyset$.

(c) Minimality Property. There exist no $G, H \in C$, such that $G \subset H$.

(d) Same or Similar Function Property. $\forall G, H \in C$, the function of G is same or similar with the function of H.

The elements of similar service set are services Si, and a service consists of atomic service AS_j. Our goal is to construct a group of service migration agents based on service resource of a fully-connected in similar service set. Each service resource is to be assigned a group of agents correlative to each service migration activity according to the computing ability, current idle computing ability, connectivity of network, execution delay and need migration rating. To do this, we must first define what these computing abilities and index are. Let U is the set of service resource in whole pervasive computing network. We define the power rating assignment of U as a function $p: U \rightarrow M$, where M is the set of positive integers. The power rating assignment of service provider A is given by $p(A)$. This power rating represents an estimate of the service provider's ability to handle service program and message traffic. The stronger is computing ability of a service provider, the power rating is higher; On the contrary, the power rating is lower. Similarly, the index rating assignment of U is a function $q: U \rightarrow M$, and the index rating assignment of service provider A is denoted $q(A)$. This rating represents an estimate of a service provider's trend of migration service to other service provider in the system; i.e., it should be proportional to the percentage of service program that the service provider will be unable to execute.

DEFINITION 3: Given a computing power rating assignment p over U and C, *Total(p(U))* and *Total(p(C))* are defined as:

$$\begin{cases} \text{Total}(p(U)) = \sum_{a \in U} p(a) \\ \text{Total}(p(C)) = \sum_{a \in C} p(a) \end{cases} \quad (4)$$

DEFINITION 4: Given a computing power rating assignment p over U, *Average(p(U))* and *Average(p(C))* are defined as:

$$\text{Average}(p(U)) = \begin{cases} \frac{\text{Total}(p(U))}{2} + 1 & \text{if Total}(p(U)) \text{ is even} \\ \frac{\text{Total}(p(U)) + 1}{2} & \text{if Total}(p(U)) \text{ is odd} \end{cases} \quad (5)$$

$$\text{Average}(p(C)) = \begin{cases} \frac{\text{Total}(p(C))}{2} + 1 & \text{if Total}(p(C)) \text{ is even} \\ \frac{\text{Total}(p(C)) + 1}{2} & \text{if Total}(p(C)) \text{ is odd} \end{cases}$$

Suppose there is a ten-service-resource network for which computing power ratings have been assigned as shown in Table 1. The sum of the power ratings listed is 35. For reasonable computing power rating assignments, the construction of similar service set with similar total computing power ratings, as shown in Table 2.

Table 1. A example of computing power rating assignment

Service resource	Computing power rating	Service resource	Computing power rating
A	3	F	5
B	5	G	4
C	3	H	4
D	3	I	2
E	1	J	3

Table 2. Similar service set and total computing power ratings

Similar service set	Total computing power rating	Average computing power rating
{A, C}	6	3
{B, D}	8	4
{ F, I, J}	12	4
{G, H, E }	9	3

DEFINITION 5: Given a migration rating q for $a \in C$ is defined as:

$$q(a) = \left\lceil \frac{1}{\left(\frac{\text{idle}(p(a))}{\text{Average}(p(C))}\right)} \right\rceil \quad (6)$$

where the idle(p(a)) is the remaining computing power rating of a.

Now, support that $L(U)$ denote the connectivity of service nodes in pervasive computing network, $D(a)$

denote the execution delay of node, *T* denote the time of sampling. When a service program is loaded on a node in service resource network, the index of service migration relate to *idle(p(a)), p(a), q(a), L(U), D(a)* and time of sampling *T*. We call *idle(p(a)), p(a), q(a), L(U)* and *D(a)* as correlative context information of service migration activity.

DEFINITION 6: Suppose that *F* denotes a kind of correlative context information, which impacts on service migration activity. The impact factor of F is defined as follow:

$$\Psi(F)=\sum_{i=1}^{n(F)}\frac{H(f_j)}{|f_j|} \quad and \quad f_j\neq\Theta \tag{7}$$

where f_j, $1\leqslant j\leqslant 5$ is a focal elements set which composed of *idle(p(a)), p(a), q(a), L(U)* and *D(a),* and $|f_j|$ is radix of the body of focal elements f_j , and $n(F)$ is the number of element in the focal elements set, and $H(f_j)$ is the basic index function of service migration. Because the value of correlative context information is varying at sampling points, the affected $H(f_j)$ by *T* is

$$\begin{cases}\hat{H}(f_j,T_i)=\eta(T_i-T_{i-1})H(f_j), & f_j\neq\Theta\\ \hat{H}(\Theta,T_i)=\eta(T_i-T_{i-1})H(\Theta)+[1-\eta(T_i-T_{i-1})]\\ \eta(T_i-T_{i-1})=\rho f(T_i-T_{i-1}), & \eta(T_i-T_{i-1})\neq 0\end{cases} \tag{8}$$

where ρ is the weight factor of reliability, and $\eta(T_i\text{-}T_{i-1})$ is the time-efficient function.

According to definition 5, the correlation between two kinds of correlative context information impacting on service migration activity is

$$\Psi(F_1,F_2)=\sum_{i=1}^{n(D_{ij})}\frac{H_d(I_{ij})}{|I_{ij}|} \quad and \quad I_{ij}\neq\Theta \tag{9}$$

where I_{ij} is correlative focal elements set, and $|I_j|$ is radix of I_{ij}. If $\Psi(F_1,F_2)=0$, it denote that correlative context information F_1 and F_2 are independent each other. If $\Psi(F_1,F_2)=1$ and $\Psi(F_1)<\Psi(F_2)$, it denote that $\Psi(F_2)$ contain $\Psi(F_1)$. If $\Psi(F_1,F_2)=1$ and $\Psi(F_1)>\Psi(F_2)$, it denote that $\Psi(F_1)$ contain $\Psi(F_2)$. So we can use $\Psi(F_1,F_2)$ to describe correlation between two kinds of correlative context information.

DEFINITION 7: The correlation coefficient R_{12} between F_1 and F_2, the correlation coefficient R_{21} between F_2 and F_1, and the correlation degree $\tau(F_1,F_2)$ between F_1 and F_2 are defined as follow:

$$\begin{cases}R_{12}=\frac{1}{2}\Psi(F_1,F_2)\sqrt{\frac{\Psi(F_2)}{\Psi(F_1)}}\\ R_{21}=\frac{1}{2}\Psi(F_1,F_2)\sqrt{\frac{\Psi(F_1)}{\Psi(F_2)}}\\ \tau(F_1,F_2)=\tau(F_2,F_1)=2\sqrt{\frac{\Psi(F_1,F_2)}{\Psi(F_1)+\Psi(F_2)}}\end{cases} \tag{10}$$

When the correlation degree $\tau(F_1,F_2)\geqslant\varepsilon$ (threshold of correlation degree between F_2 and F_1), F_1 and F_2 are regarded as correlative context information of service migration activity. Because the correlation among all kinds of context information is an objective reality, we can only adopt the method of decorrelation to measure and eliminate correlation each other, and can not ignore deliberately its. Based on the thought, the adopted method of decorrelation is that conflicting portions of context information are transformed into a set of open frame of discernment and distributed into all elements rather than partial several elements. Virtually, the index of service migration is computed after correlative context information is transformed into independent context information.

According to above theory, the expression that evaluate the index of service migration relate to *idle(p(a)), p(a), q(a), L(U), D(a)* and *T* is as follow:

$$\begin{cases}\hat{H}(\phi)=0\\ \hat{H}(f,T)=\frac{1}{K}\sum\limits_{\cap f_j=f}\prod\limits_{\substack{0<i<n\\1\leq j\leq 4}}H_i'(f_j,T) \quad and \quad f\neq\phi\\ \hat{H}(\Theta,T)=(\sum\limits_{\cap f_j=\phi}\prod\limits_{0<i<n}H_i'(f_i,T))+\gamma\\ H_i'(f_j,T)=\begin{cases}H_i(f_j,T)(1-\mu_{i(n-i)}), & f_j\neq\Theta\\ 1-\sum\limits_{f_j\subset\Theta}H_i(f_j,T), & f_j=\Theta\end{cases}\\ K=1-\sum\limits_{\cap f_j=\phi}\prod\limits_{\substack{0<i<n\\1\leq j\leq 4}}H_i'(f_j,T))\\ K\neq 0\end{cases} \tag{11}$$

When the correlation degree $\tau(F_1,F_2)<\varepsilon$ (threshold of correlation degree), the context information are regarded as independent factor. The integrative index function of service migration $H(f,T)$ is

$$\begin{cases}\hat{H}(f,T)=\frac{1}{K}\sum\limits_{\cap f_j=f}\prod\limits_{\substack{0<i<n\\1\leq j\leq 4}}\{\rho_i H_i(f_i,T)[\rho_j H_j(f_j,T)+(1-\rho_j)]\} \quad and \quad f\neq\Theta\\ \hat{H}(\Theta,T)=\frac{1}{K}\sum\limits_{\cap f_j=f}\prod\limits_{0<i<n}\{\rho_i H_i(\Theta,T)+(1-\rho_i)\}\end{cases} \tag{12}$$

And then, using the method of evaluating the index of service migration, we can obtain the judgment

theorem of triggering a service migration activity as follow theorem 1.

THEOREM 1: For a service that is consisted of the *n* atomic service AS_j $(0 \le j \le n)$, if and only if $H(f,T) \geqslant \lambda$ (where λ is the predetermined threshold of triggering the service migration activity), the service migration activity relevant to the service is triggered, and $H(f,T)$ is called function of triggering service migration activity.

PROOF: For a service that is consisted of n atomic service AS_j $(0 \le j \le n)$, if there exists correlation among all kinds of context information relevant to atomic service AS_j, then the correlation degree $\tau(F_x,F_y)>0$.

Four cases are considered.

Case 1: When correlation degree $\tau(F_x,F_y) \ge \varepsilon$, those context information are regarded as correlative context information of the service migration activity. Then the $\widehat{H}(f,T)$ is computed according to Eq.(11).

Case 2: When correlation degree $0<\tau(F_x,F_y)< \varepsilon$, those context information are regarded as independent context information of service migration activity. Then the $\widehat{H}(f,T)$ is computed according to Eq.(12).

Obviously, $\widehat{H}(f,T) > 0$.

Case 3: When $\rho(F_x)\rho(F_y) = 1$, then $\widehat{H}(\phi) = \widehat{H}(\Theta) = 0$ and $\widehat{H}(f,T) = 1$.

Obviously, $\widehat{H}(f,T) = \lambda$. So the service migration activity relevant to the service is triggered.

Case 4: When $\rho(F_x)\rho(F_y) \neq 1$, then there exists that at least one weight factor of reliability between $\rho(F_x)$ and $\rho(F_y)$ is not 1. This is to say, there exists that at least one between $\widehat{H}_x$ and $\widehat{H}_y$ is not completely reliable. And $\widehat{H}(\phi) \neq 0$, $\widehat{H}(\Theta) \neq 0$, $0 < \widehat{H}(f,T) < 1$. When $\widehat{H}(f,T) \ge \lambda$, the service migration activity relevant to the service is triggered. When $\widehat{H}(f,T) < \lambda$, the service migration activity relevant to the service is not triggered.

□

There is another problem. How to evaluate and judge whether a service migration is successfully completed or not? We synthetically evaluate a successful service migration activity using four parameters including correct ratio of resuming service program and data, transferring failure ratio, satisfaction ratio of transferring delay and remnant-dependence ratio. Then, we compute the comprehensive index of completing successful service migration activity.

DEFINITION 8: The four parameters relevant to the comprehensive index of completing successful service migration activity are defined as:

(a) The correct ratio of resuming service program and data on the target node in the course of service migration is a correctly resumed ratio of service Si consisting of n atomic service AS_j $(0 \le j \le n)$ after the service migration agent has arrived at target node;

(b) The satisfaction ratio of transferring delay is a ratio of two kinds of time. One is that after agents have been encapsulated, it takes some time to migrate agents from source node to target node. Another is the expected time that user obtain results;

(c) The transferring failure ratio is a failure probability of service migration in the course of service migration activity because of the target node movement, or disconnection of network etc. resulting in the phenomena that migration agent can not arrive at the target node;

(d) The remnant-dependence ratio is a ratio between the required information that yet dependent on source node and the total required information when service program is resumed to execute at the target node.

Set δ, θ, σ, ζ, β respectively denote the correct ratio of resuming service program and data, the satisfaction ratio of transferring delay, the transferring failure ratio, the remnant-dependence ratio and the comprehensive index of completing successful service migration activity.

We use the four parameters to compute the comprehensive index of completing successful service migration activity as follow Eq.(13).

$$\begin{cases} \beta(\delta,\theta,\sigma,\zeta) = \varpi_1\delta + \varpi_2\theta + \varpi_3\sigma + \varpi_4\zeta \\ \varpi_1 + \varpi_2 + \varpi_3 + \varpi_4 = 1 \\ \delta = \dfrac{i}{K} \\ \theta = \dfrac{\Delta t}{f(n)} \quad f(n) = (n-1)m + (n-1)(t+m) \\ \sigma = \dfrac{u}{v} \\ \zeta = \dfrac{f'(x)}{f'(T)} \end{cases} \tag{13}$$

where $\varpi_i, 1 \le i \le 4$ is the comprehensive weight factor and $0 \le \delta,\theta,\sigma,\zeta,\beta,\varpi_i \le 1$.

We can obtain the judgment theorem of completing successful service migration activity as follow theorem 2.

THEOREM 2: For a service migration process that is consisted of the n atomic service AS_j $(0 \le j \le n)$, if and only if $\beta(\delta,\theta,\sigma,\zeta) \geqslant \pi$ (where π is the predetermined threshold of the successful service migration activity), the service migration activity relevant to the service is regarded as a successful

service migration activity, namely this service migration activity satisfy the request of seamlessness, and $\beta(\delta, \theta, \sigma, \zeta) \geq \pi$ is called evaluation function of seamlessness for successful service migration activity.

4.3. *Flexible communication protocol of service migration agents*

The adopted communicating network of service migration agents consists of coordinative communication agents (CCAs) and coordinative communication agents protocols (CCAPs). The CCAP is a distributed abstraction layer that provides the mechanisms both for migration and communication and for the security of the underlying system. Utilizing services from CCAPs, service migration agents can cooperate, communicate and migrate in the network.

The architecture of the adopted communicating network of coordinative communication agents is depicted in Figure 2. The adopted communicating network of coordinative communication agents is built upon the mobile agent computing network and provides group communication services for service migration agents group based on migrated service program. There are two components in the coordinative communication agents protocols: (a) reliable multicast message protocol, (b) grouping mechanism. The first component is responsible for implementing reliable multicast communications in the service migration agents group, which achieves two fundamental properties of reliable multicast, i.e. atomicity and total ordering. The second component is responsible for grouping. All kinds of service migration agents which belong to the same service migration process are clustered as a group.

Let $H=\{MA_1, MA_2, ..., MA_n\}$ be a group of n service migration agents which belong to the same service migration process, and $U=\{CA_1, CA_2, ..., CA_m\}$ be a ring of m communication coordination agents. An agent group AG can be defined as:

$$AG(H,CA_i)=\{MA_\alpha \mid MA_\alpha \in H \wedge Coordinator(MA_\alpha)=CA_i\} \quad (14)$$

where $1 \leq i \leq m$ and CA_i is the communication coordination agent of the AG.

Let each coordinative communication agent maintains a group view GV for local agent group relation to the same service, which is the seven tuple:

$$GV = (HID, AGList, B_1, B_2, Seq_1, Seq_2, Seq_3) \quad (15)$$

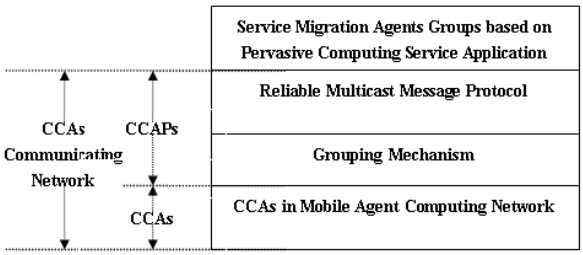

Fig. 2. Architecture of the CCAs communicating network.

where: (a) *HID* is the *ID* of the multicast group *H*; (b) *AGList* is the list of service migration agents in the local agent group; (c) B_1 buffers multicast messages that have been sent by service migration agents in the local agent group but not been multicast to group *HID*, and these messages are not assigned with the sequence number yet; (d) B_2 buffers multicast messages received from group *HID*, and these messages are ordered by their sequence numbers in B_2; (e) Seq_1 is the maximum sequence number of messages that have been delivered by local agents; (f) Seq_2 is assigned with the value of the Seq_1; (g) Seq_3 is the safe sequence number.

Let communication protocol include two types of messages: *MM* (multicast message) and *CM* (control message).

$$MM = (HID, Seq_4, SourceID, Info) \quad (16)$$

where: (a) Seq_4 is the sequence number of message *MM*; (b) *SourceID* is the *ID* of the agent sending message *MM*; (c) Info is the content of message *MM*.

$$CM = (CMID, AgentID, Seq_5) \quad (17)$$

where: (a) *CMID* is the type *ID* of the control message; (b) *AgentID* is the *ID* of the agent; (c) Seq_5 is used to record the maximum sequence number of messages, which are required to be pulled or have been pulled by local agents.

Our flexible communication protocol of service migration agents is designed to achieve two fundamental properties of reliable multicast: atomicity and total ordering. Atomicity guarantees that any multicast message is eventually delivered to all agents relevant to the service migration process. Total ordering ensures that all the multicast messages are delivered in the same order to all the group members. The ring is a common approach to implement the atomicity and total ordering multicast mechanism in pervasive computing

system. We give out the pseudocode of our flexible communication protocol of service migration agents as follow:

The pseudocode of flexible communication protocol

1. When the number of messages in the B_2 reaches to a pre-specified threshold and these messages are consecutive in their sequence numbers, the coordinator broadcasts a *CM=(CMID,AgentID,Swq$_5$)* to all agents relevant to the service migration process in the group of service migration agents;
2. On receipt of the *CM=(CMID,AgentID,Swq$_5$)*, all agents belonging to the same group of service migration agents pull the pre-specified threshold from B_2 in the sequence order and the last message pulled by all agents has the sequence number Seq_5 ;
3. After the pull operation, the agents of receipted message send the acknowledgement control message *CM=(CMID,AgentID,Swq$_5$)* to the coordinator for updating the record in the *AGList*;
4. Accept_communication_message(*Agent_ group AG*)

```
Begin
   Coordinator C=GetCoordinator();//Get the current
     coordinatior
  Groupview GV=c.getGV(Agent_group.HID);//Get the
     relevant to group view
  Call Multicast(GV.B1);
  Call Release(GV.B2);
  PassNextCoordinator(AG);//Pass the message to the
     next coordinator in the ring
End
```

5. Procedure Multicast(B_1)

```
Begin
 Repeat
  If (B1 is not empty)
    MulticastMessage MM=B1.GetMessage(); //Get the
      first message in B1
    AssignSequenceNumber(MM,AG.Seqmax+1);//Assign MM
      with the sequence number
    AG.Seqmax=AG.Seqmax
    Send(MM,AllCoordinators);//Multicast MM to all
      coordinators in the ring
    B1.removeMessage(MM);//Remove MM from B1
  End If
 End Repeat
End
```

6. Procedure *Release(B_2)*

```
Begin
  If (AG.Seq3==GV.Seq2);//Current group is the one
     that has the minimum value of Seq1 in
     previous rotation of AG
   AG.Seq2=AG.Seq3;//Moves AG.Seq3 to AG.Seq2
   AG.Seq3=GV.Seq1;//Updates AG.Seq3 with GV.Seq1
  Else
    AG.Seq3=Min(AG.Seq3,GV.Seq1);//Updates AG.Seq3 with
     the minimum value of AG.Seq3 and GV.Seq1
  End If
  GV.Seq2=GV.Seq1;//Moves GV.Seq1 to GV.Seq2
  GV.Seq3=AG.Seq2;//AG.Seq2 is the safe sequence
     number for releasing messages in B2 and copy
     it to GV.Seq3
   B2.realseMessage(GV.Seq3);//Release message in B2
     whose sequence numbers are less than or equal
     to GV.Seq3
  End
```

4.4. *SOSM algorithm*

In this section, we put forward a service-oriented seamless migration algorithm in pervasive computing system. The steps of SOSM algorithm are given out as follow:

The steps of SOSM algorithm

Step0. Begin;

Step1. Monitoring the status of fault detector and the service migration system is triggered as soon as the fault manager receive an alert message;

Step2. Calculating the value of $\widehat{H}(f,T)$, and judging that if its value is more than the threshold predefined λ ;

Step2.1 If satisfying the condition of service migration triggering, then skip to step 3;

Step2.2 If don't satisfying the condition of service migration triggering, then return step 1;

Step3. Starting the service migration activity;

Step3.1 Looking up the elements of service similar sets, and sorting elements according to the value of the service evaluating function, and selecting the service provider with the next to the highest value as the target node;

Step3.2 Scheduling the service migration route and optimizing it;

Step3.2.1 Establishing handshake signal or connecting information between source node and target node; If its can be connected, then return step 3.2.2; If its can not be connected, then return the fail information of service migration activity and return 5;

Step3.2.2 After affirming the successful connection information, looking up all possible route form source node to target node in service network;

Step3.2.3 Calculating the completion time of nodes and edges, the right of reliability of nodes and edges, the transmission delay of edges and so on;

Step3.2.4 Calculating the value of the service migration cost function, and selecting the route with the minimum as the optimal route; Then return step 3.3;

Step3.3 Packaging the service program and date migrated into a group of service migration agents, and transferring its from the source node to target node along the optimal route;

Step3.4 Restarted execution using a checkpoint on new node of service network, and changing the status of service program to waiting status;

Step3.5 Calculating the resumed-migration-node correct ratio ψ , the transferring delay ratio Φ , the remnant-dependence ratio σ , the seamless function of service migration β (δ θ σ ζ) and the value of other correlative factors and so on, then judging that if service migration activity is successful;

Step4. Other activity of service migration activity;

Step5. End

The idea is that for pervasive computing environments considered in this paper, our service selection system eliminates user participation and selects the most appropriate service from similar service set using an intelligent service selection algorithm based on artificial neural network. When the system happen the service failure, a service migration activity is trigged. According to the results of looking up the fault detector and fault manager, we decide which service should be migrated. According to the results of looking up the service resource manager and local resource manager, we decide where the service program should be migrated. Then, we decide how to optimize the service migration route. Finally, the service program is restarted execution using a checkpoint on new node of service resource network. The SOSM algorithm adopts the events, conditions and actions mechanism, shortly called ECA mechanism, is defined as follows: the part of events in an ECA mechanism is that service failure happened in service selection system and the service; the part of conditions in an ECA mechanism is a Boolean statement that must be satisfied to in order to activate the rule, i.e. conditions are that there exists a correlative service similar set and service migration activity is triggered; finally, the part of action in an ECA mechanism is the instruction that must be executed when a triggered rule is activated.

5. Experiment and Results Analysis

In order to evaluate the performance of above proposed ANN-based service selection algorithm and service-oriented seamless migration algorithm, we have developed an intelligent service selection prototype system (shortly called ISSPS), and have implemented the ANNSS algorithm, SOSM mechanism and flexible communication protocol in a pervasive web services system shown in Figure 3. We have fulfilled all kinds of simulations. We implemented the fault detector and fault manager in JAVA and our tested service migration network consists of the 8 hosts. We want to implement an example scenario: a traveller arriving at one of web-service-bars finds a barcode service token, and her mobile device reads the token through its inbuilt camera and decodes the service identifier, and he can select all kinds of pervasive web services that servers can provide, and when the selected service provider can not provide the service of satisfying QoS requests, the service program and data are migrated to a new service provider with the same or similar function and the service migration process is transparent to user, namely users do not have to handle details of service migration activity and only concern with the results of service and QoS. In the experiment platform, we premise that the web services model should form a valuable basis for pervasive computing in dynamic environments. A framework is developed for the real-world discovery and invocation of pervasive services through physical, detectable tokens and a web services approach to service invocation. Web services provide lightweight, open standards for middleware functionality that are language and platform independent. The foundations of this model are XML-based messaging, a service description, and service discovery, find and bind paradigm, where service providers publish descriptions to lookup service module. The service description is above USDL document. We implemented a token authoring tool that accepts a USDL file, such as that generated automatically by standard tools, and generates suitable service description to the lookup service module. A barcode is then returned that confers the unique service identifier in a standard encoding, and is distributed in the real-world -for example, on display at the screen of web-service-bar. The web services model proved a powerful but lightweight framework for exposing service functionality in pervasive computing applications. Services can be deployed with ease and select the optimal service for user. If the percent of successful service activity is below 80%, it is considered that the performance of service migration system is not eligible, i.e. the threshold of jitter error is set to no more than 0.2.

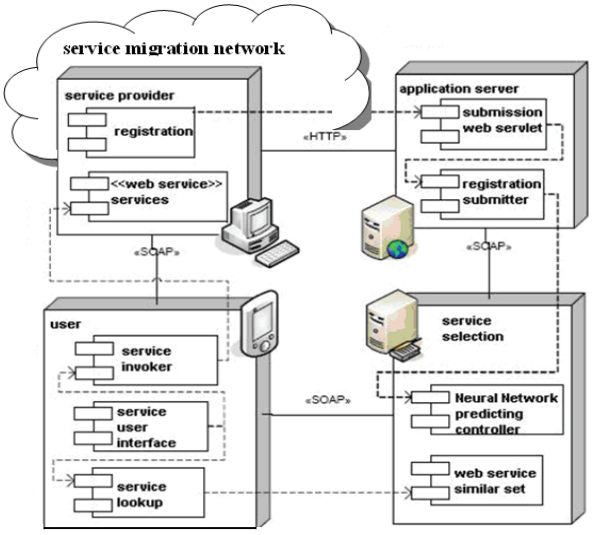

Fig. 3. Architecture of ISSPS.

There are free main tested objectives:

(a) Analysing the effect of the number of atomic services that consist of service entity for service migration activity, which is beneficial to optimize the performance of service migration system by setting appropriate number of atomic services that consist of service entity;

(b) Analysing execution time and the latency caused by service migration activity, which is one of the important parameters, and calculating the value of others parameters including the function of triggering service migration activity *H(S, T)*, and so on;

(c) Comparing system performance for various service selection systems when service failures happen, including the system with service migration mechanism and without service migration mechanism.

Figure 4 shows the comparison of the execution time for three different approaches to select service resources: i.e. the ANNSS algorithm, random selection, and minimal selection. Random selection service approach is that a user randomly selects the service resources for service program execution. Minimal selection service approach is that a resource manger selects the service resources of service program execution using ranking. The ranking is decided by the execution time of each atomic service and has high ranking when the execution time is small. Figure 4(a) is the expected execution time given by artificial neural network predicting controller. Figure 4(b) shows the actual execution time by simulation. In Figure 4(b), the performance of service execution with the ANNSS algorithm is better than with random selection or minimal selection approach. The total execution time of service depends on the longest execution time of atomic services. Thus it is important that the longest execution time of atomic services is minimized and optimize the performance of service migration system by setting appropriate number of atomic services that consist of service entity.

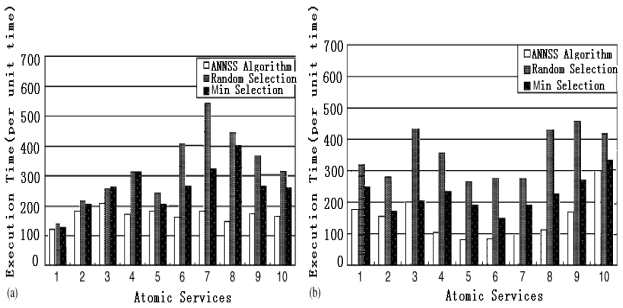

Fig. 4. The comparison of three different approaches to select service resources.

Table 3 shows those important parameters relevant to service migration activity. We suppose that the threshold of triggering service migration activity is 0.80 and the threshold of seamlessness of completing successful service migration activity is 0.80. We found the service migration activity occurred if the value of function of triggering service migration activity is more than the threshold of triggering service migration activity. We also found the service migration process is successfully completed if the value of evaluation function of seamlessness is more the threshold of seamlessness. The value of cost function of service migration activity is increasing with the value of evaluation function of seamlessness for successful service migration activity.

Table 3. Important parameters of relevant to service migration activity.

NO. / Parameters	1	2	3	4	5	6	7	8	9	10
$\bar{H}(f,T)$	0.83	0.80	0.86	0.80	0.84	0.81	0.82	0.85	0.89	0.96
λ	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80	0.80
$\beta(\delta,\theta,\sigma,\zeta)$	**0.93**	**0.81**	**0.87**	**0.84**	**0.82**	**0.81**	**0.80**	**0.89**	**0.91**	**0.95**
π	**0.80**	**0.80**	**0.80**	**0.80**	**0.80**	**0.80**	**0.80**	**0.80**	**0.80**	**0.80**

Figure 5 shows the system performance benefits due to service migration when service failures happen. In Figure 5, we found that service migration occurred 10 times if migration threshold was 0.8. Notice that most of the times when service failures occurred, service migration didn't occur because service migration overhead was high.

Figure 6 shows the comparison of reliability between the service selection system with the SOSM algorithm and without the SOSM algorithm. The horizontal axis represents time (per unit time) and the vertical axis represents reliability of service selection system. The higher it is, the better stability of the system is. Results show that the value of system reliability with SOSM algorithm increases when more times have been executed. Besides, the fluctuation of system reliability with SOSM algorithm is less than without SOSM algorithm, which denotes the service selection system with SOSM algorithm has higher precision of web services selection and execution than without SOSM algorithm. Therefore, the service-oriented seamless migration scheme with SOSM algorithm is superior to the traditional service selection system. The novel method can exactly improve the reliability of the service selection system and provide the optimal service performance to users.

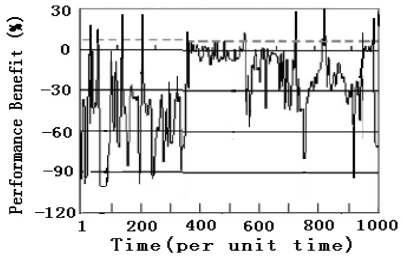

Fig. 5. Performance benefits due to service migration when service failures happened.

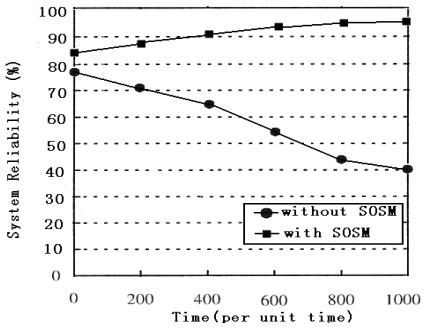

Fig. 6. Comparison of reliability of the service selection system between with SOSM and without SOSM.

Through implementation and simulation results, we confirmed that: (a) the service selection controller can select the optimal service provider from the service similar set which are managed by the service resource manager; (b) service migration guarantees that the submitted service program and data are reliably completed and improves the performance of service execution.

6. Conclusion and Future Works

In this paper, we proposed, based on service selection model, the service seamless migration mechanism among the nodes of service resources network when service failure happen in pervasive computing systems. The contributions of this work are as follows.

(a) We extended the definition and formal description of service from the conventional notion in a distributed systems in order to provide a fault tolerance service that deals with various types of service resources failures that don't satisfy QoS requirements.

(b) We put forward the architecture of service migration system. At the same time, we put forward two judgment theorems of triggering a service migration activity and completing a successful service migration activity. We have implemented these in intelligent service selection prototype system (i.e. ISSPS), which provide the pervasive web services for users in a pervasive computing environments. Namely, the ANNSS model and SOSM mechanism are implemented in this system. We performed the simulation to measure the performance improvement due to optimal service selection from service similar set and service migration when service failure happens. We tested our approaches through extensive simulations and obtained promising experimental results. The results of simulation show that the proposed service selection and migration scheme is not only reliable but also efficient.

In the future, we plan to implement our SOSM algorithm for more complex pervasive computing environments with large numbers of service resource nodes and make various experiments for measuring efficiency of the service selection and migration. Another important future work is to how to implement load balance control mechanism in the course of service migration activity for pervasive computing environment.

Acknowledgements

This research is partially supported by the Natural Science Foundation of China under Grant No.91118008 and 61021004, the National High-Tech Research and Development Plan of China under Grant No.2011AA010101, and Shanghai Knowledge Service Platform for Trustworthy Internet of Things under Grant

No. ZF1213. This research is also partly supported by the Specialized Research Fund for Doctoral Program of Higher Education of China under Grant No.20100076120011 and State Key Laboratory of Rail Traffic Control and Safety (Beijing Jiaotong University) No.RCS2011K014.

References

1. M. Weiser, The computer for the 21st Century. Scientific American, 1991,282(3):94-104
2. M. Weiser, Some computer science issues in pervasive computing. Communications of the ACM, 1993,36(7):75-84
3. Y. Artsy, R. Finkel, Designing a process migration facility: the Charlotte experience, IEEE Computer 22(9), 1989
4. F. Douglis, J.K. Ousterhout, Transparent process migration: design alternatives and the Sprite implementation, Software Practice and Experience 21(8), 1991
5. M.L. Powell, B.P. Miller, Process migration in DEMOS/MP, in: Proceedings of the 9th ACM Symposium on Operating Systems Principles, Bretton Woods, NH, October 1983
6. M. Theimer, K. Lantz, D. Cheriton, Preemptable remote execution facilities for the V-system, in:Proceedings of the 10th Symposium on Operating system Principles, Orcas Island, WA, December 1985
7. V.C. Zandy, B.P. Miller, M. Livny, Process hijacking, in: 8th International Symposium on High Performance Distributed Computing, Redondo Beach, CA, August 1999
8. E. Zayas, Attacking the process migration bottleneck, in: Proceedings of the 11th ACM Symposium on Operating System Principles, Austin, TX, November 1987
9. M. Litzkow, T. Tannenbaum, J. Basney, M.Livny, Checkpoint and migration of Unix processes in the condor distributed processing system, Technical Report 1346, University of Wisconsin-Madison Computer Science, 1997,57-64
10. J.J. Kistler, M. Satyanarayanan, Disconnected operation in the Coda file system, ACM Transactions on Computer Systems, 10 (1), (1992).
11. H.A. Simon, Designing organizations for an information-rich world, in: M. Greenberg (Ed.), Computers, Communications and the Public Interest, Johns Hopkins Press, Baltimore, MD, 1971
12. M. Stemm, R.H. Katz, Vertical handover in wireless overlay network, Mobile Network Application 3 (1998) 335–350
13. J. Indulska, S. Balasubramaniam, Vertical Handover based Adaptation for Multimedia Applications in Pervasive Systems, Joint International Workshop on Interactive Distributed Multimedia Systems and Protocols for Multimedia Systems (IDMS/PROMS 2002). Lecture Notes in Computer Science, LNCS 2515., 2002, pp. 61–72
14. S. Helal, C. Lee, Y. Zhang, G.G. Richard III, An architecture for wireless LAN/WAN integration, IEEE Wireless Communication and Networking Conference (WCNC), Chicago, 2000 September (2000).
15. M.P. Tiemeyer, J.S.K. Wong, A task migration algorithm for heterogeneous distributed computing systems. The Journal of Systems and Software, 41(1998), pp:175-188
16. H. Cai et al., A novel ANN-based service selection model for pervasive computing environments. Journal Network and Computer Applications (2007), pp:1-22
17. M. Queyranne, Structure of a simple scheduling polyhedron, Math. Programming 58(1993)263-285
18. L.A. Hall, A.S. Schulz, D.B. Shmoys, J. Wein, Scheduling to minimizing average completion time: off-line and on-line algorithms, Math. Oper. Res. 22(1997)513-544
19. C. Haibin, P. Fang, H. Yuncai, C. Qiying, A Novel ANN-Based Service Selection Model for Pervasive Computing Environments. Journal of Network and Computer Applications, 2007, 28(8):88-108
20. F. Vraalsen, R. Aydt, C. Mendes, D. Reed, Performance Contracts: Predicting and Monitoring Grid Application Behavior, in: Proceedings of the 2th International Workshop on Grid Computing, 2001
21. A. Waheed, W. Smith, J. Georage, J. Yan, An Infrastructure for Monitoring and Management in Computational Girds, in Proceedings of the 5th Workshop on Languages, Compilers, and Run-time Systems for Scalable Computers, March 2000
22. A. Itzkovitz, A. Schuster, L. Shalev, Thread Migration and its Applications in Distributed Shared Memory Systems, The Journal of Systems and Software, Vol. 42, 1998, 71-87
23. K. Shudo, Y. Muraoka, Asynchronous Migration of Execution Context in Java Virtual Machines, Future Generation Computer Systems, Vol. 18, 2001,225-233