ISSN:0975-9646

Amena Nasreen et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (3) , 2012,4508-4513

A Novel Method for Divide and Convergent Coherent Mechanism in Cloud Computing

Amena Nasreen, SK.Haseena, G.Srinivasa Rao
Anurag Engineering college, Kodad,A.P., India

Abstract—Current cloud computing systems always focus on the high bandwidth local area network environment, for example, an LAN of a corporation. This paper uses mobile agent to implement the software and data service for cloud user in Internet environment, and make the cloud computing system adaptable to work in Internet environment, such as an international corporation with branches all over the world. The works in this paper as follows: Introducing mobile agent into cloud computing system and presenting the mobile agent based service for cloud computing system: Service as an Agent Service (SaaAS). The SaaAS uses mobile agents as the underlying facility to offer the service for user; presenting a high performance code and data of service load mechanism based mobile agent for SaaAS, which can effectively reduce the heavy communication overhead in Internet; presenting a novel data coherence mechanism: Divided-Cloud and Convergent Coherence Mechanism (DCCM).

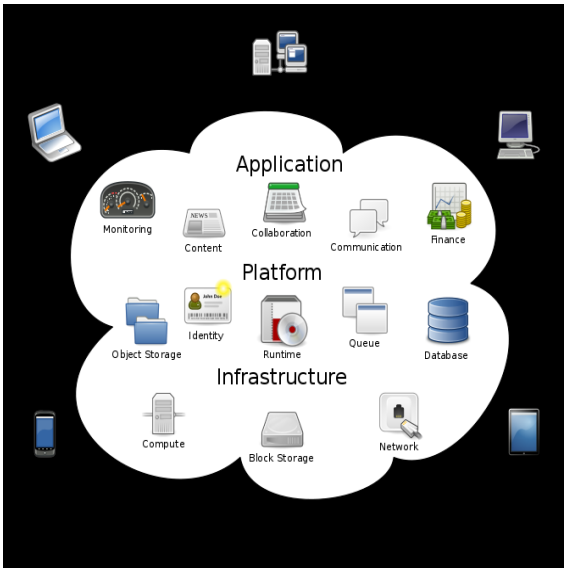
Keywords: Cloud Computing; Mobile Agent;Codes Load;Convergent Coherence

I.INTRODUCTION

Cloud computing refers to the delivery of computing and storage capacity as a service to a heterogeneous community of end-recipients. The name comes from the use of clouds as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts services with a user's data, software and computation over a network. It has considerable overlap with software as a service (SaaS).End users access cloud based applications through a webbrowser or a light weight desktop or mobile applications while the business software and data are stored on servers a remote location. Proponents claim that cloud computing allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and enables IT to more rapidly adjust resources to meet fluctuating and unpredictable business demand.

Cloud computing relies on sharing of resources to achieve coherence and economies of scale similar to a utility (like the electricity grid) over a network (typically the Internet). At the foundation of cloud computing is the broader concept of converged infrastructure and shared services.

Cloud computing is a paradigm that focuses on sharing data and computations over a scalable network of nodes [1, 2].Examples of such nodes include end user computers, data centers, and web services. Such a scalable network of nodes is called cloud. An application based on such clouds is taken as a cloud application. A computing cloud is a massive network of nodes. Thus, scalability should be a quality feature of the computing cloud. The most important scalability is Horizontal cloud scalability, which is the ability to connect and integrate multiple clouds to work as one logical cloud [3,4]. For instance, a cloud providing calculation services (calculation cloud) can access a cloud providing storage services (storage cloud) to keep intermediate results. Two calculation clouds can also integrate into a larger calculation cloud. Scalability should be transparent to users. For instance, users may store their data in the cloud without the need to know where it keeps the data or how it accesses the data.



For example, every cloud has only a finite amount of physical storage entities. Therefore, a cloud c1 may seek help from another cloud c2 for shared storage entities to fulfill some demands on storage. Such sharing requirement may result in the data to migrate among multiple clouds. Nevertheless, the cloud user should not be aware of the distributed storage of the data. For instance, when the stored data needs to be accessed, the user may directly retrieve it from the cloud c1. Then c1 is responsible for gathering the data from both c1 and c2, and returns the collected data to the user. The cloud provides location transparency to applications.From this scenario, the cloud data storage and access may need not only intra-cloud communications, but also inter-cloud communications.

That is to say, the cloud data need not only be accessed in a LAN, but also migrate in WAN. In LAN environment, cloud computing system can use Remote Procedure Call (RPC) or Remote Method Invocation (RMI) as the underlying facility, to implement the service directory coherence and service migration. RPC and RMI can achieve good performance in LAN, but is not suitable for Internet or WAN. Mobile agents on the Internet or WAN have the characteristics as follows: Autonomy, Personality,Communication, Mobility, High Performance and Fault-tolerance. Mobile agents are mainly intended to be used for applications distributed over wide area (slow) networks because they can save communication costs by moving the

resource and service to the remote target environment which is near the user. A mobile agent based cloud computing system for WAN (SaaAS) is presented in this paper. By using the mobile agent rather than RPC/RMI as the underlying facility to implement the service directory coherence and service migration, SaaAS is more suitable to work in Internet.

This paper presents a code and data of service load mechanism based mobile agent and divided-cloud and convergent coherence mechanism of SaaAS, which can effectively reduce the heavy communication overhead in Internet.These eatures above enable SaaAS to have good flexibility, adaptability and usability and to be more suitable to work in Internet. This paper is organized as the following: section 2 introduces The Mobile Agent based Service for Cloud Computing in Internet Environment (SaaAS); section 3 describes code and data of service load mechanism based mobile agent; section 4 presents divided-cloud and convergent coherence mechanism; section 5 is the conclusion of this paper.

II.MOBILE AGENT-BASED CLOUD COMPUTING SYSTEM IN INTERNET

Mobile agents are autonomous programs that can travel from computer to computer in a network, at times and to places of their own choosing. The state of the running program is saved, by being transmitted to the destination. The program is resumed at the destination continuing its processing with the saved state. They can provide a convenient, efficient, and robust framework for implementing distributed applications and smart environments for several reasons, including improvements to the latency and bandwidth of client-server applications and reducing vulnerability to network disconnection. In fact, mobile agents have several advantages in the development of various services in smart environments in addition to distributed applications.

• **Reduced communication costs:** Distributed computing needs interactions between different computers through a network. The latency and network traffic of interactions often seriously affect the quality and coordination of two programs running on different computers. As we can see from Figure 1, if one of the programs is a mobile agent, it can migrate to the computer the other is running on communicate with it locally. That is, mobile agent technology enables remote communications to operate as local communications.

• **Asynchronous execution**: After migrating to the destination-side computer, a mobile agent does not have to interact with its source-side computer. Therefore, even when the source can be shut down or the network between the destination and source can be disconnected; the agent can continue processing at the destination. This is useful within unstable communications, including wireless communication, in smart environments.

• **Direct manipulation** A mobile agent is locally executed on the computer it is visiting. It can directly access and control the equipment for the computer as long as the computer allows it to do so. This is helpful in network management, in particular in detecting and removing device failures. Installing a mobile agent close to a real-time system may prevent delays caused by network congestion.

• **Dynamic-deployment** of software Mobile agents are useful as a mechanism for the deployment of software, because they can decide their destinations and their code and data can be dynamically deployed there, only while they are needed. This is useful in smart environments, because they consist of computers whose computational resources are limited.

• **Easy-development of distributed applications** Most distributed applications consist of at least two programs, i.e., a client-side program and a server side program and often spare codes for communications, including exceptional handling. However, since a mobile agent itself can carry its information to another computer, we can only write a single program to dine distributed computing. A mobile agent program does not have to dine communications with other computers. Therefore, we can easily modify standalone programs as mobile agent programs.

Cloud computing provides cheap and efficient solutions of storing and analyzing mass data. It is very important to research the data mining strategy based on cloud computing from the theoretical view and practical view. In this paper, the strategy of mining association rules in cloud computing environment is focused on. Firstly, cloud computing, Hadoop, MapReduce programming model, Apriori algorithm and parallel association rule mining algorithm are introduced. Then, a parallel association rule mining strategy adapting to the cloud computing environment is designed. It includes data set division method, data set allocation method, improved Apriori algorithm, and the implementation procedure of the improved Apriori algorithm on MapReduce. Finally, the Hadoop platform is built and the experiment for testing performance of the strategy as well as the improved algorithm has been done. The results show that the strategy designed in this paper can archive higher efficiency when doing frequent item set mining in cloud computing environment.

As Fig.1 showing, the mobile agent based cloud computing system in Internet is consisted of a lot of domain. Every domain is a cloud, always in a high bandwidth LAN. All domains are linked to each other through low bandwidth and high latency WAN. So, the SaaAS is divided into a number of domains in which are connected through high bandwidth LAN and linked to each other through low bandwidth WAN. In SaaAS, one domain acts as the major domain and is in charge of the all others domains. Every domain has a server called Domain Server (DS) which run the environment for mobile agent. The whole SaaAS which contains lots of DS is a large running platform for mobile agent.

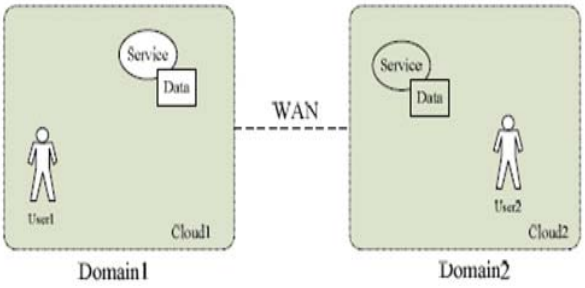


Figure 1. SaaAS Cloud Computing System in Internet

SaaS and DaaS are the two important concept in cloud computing system. (1) Software as a Service (SaaS): Software or an application is hosted as a service and provided to customers across the Internet. This mode eliminates the need to install and run the application on the customer's local computers [8, 9]. (2)Data as a Service (DaaS): Data in various formats and from multiple sources could be accessed via services by users on the network.

Users could, for example, manipulate the remote data just like operate on a local disk or access the data in a semantic way in the Internet [10, 11]. In cloud computing system, the cloud user should not be aware of the distributed storage of the data [10]. As Fig.1 showing, the user1 in cloud1want to access the data, he will directly retrieve it from the cloud1. Then cloud1 is responsible for gathering the data from both cloud1 and cloud2, and returns the collected data to the user. The cloud provides location transparency to applications. From this scenario, the cloud data storage and access may need not only intra-cloud communications, but also inter-cloud communications. That is to say, the cloud data need not only be accessed in a LAN, but also migrate in WAN. But in fact, in normal cloud computing system, the software and data are always separated, and the association of software and data is depended on the user's operation.

For example , if a user use browser to download a data to the local disk files system. He must choice correctable software to process the data. With the separation of software and data, the software and data always may distribute in differ domain across the WAN. This distribution of software and data always result in head communication overhead in WAN, and heavily decrease the performance of cloud computing system. If the software and data can be combined into a single entity, this problem can be resolved gracefully because the software and the data always in a single entity and avoid the communication overhead in WAN. The mobile agent has good flexibility, adaptability and usability and to be more suitable to work in Internet. In SaaAS, the software and data are combined into a single mobile agent. As Fig. 2 showing, the mobile agent in SaaAS consisted of three parts: data, software and common run-time code library.

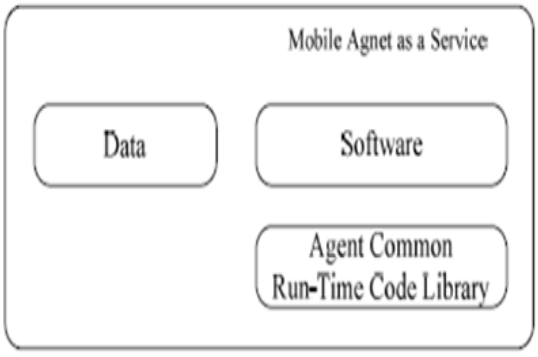


Figure 2. The Componet of WA in SaaAS

The architecture of cloud computing system is showed in Fig. 3, which the DaaS, SaaS and HaaS are separated.

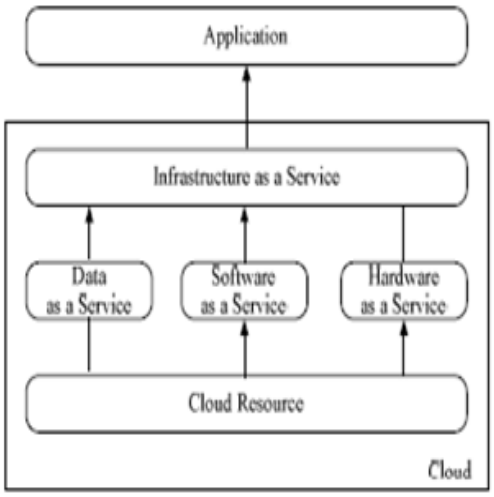


Figure 3. Architecture of Cloud Computing System

The architecture of mobile agent based cloud computing system is showed in Fig. 4, which the DaaS and SaaS are combined into a mobile agent, and the HaaS evolutes to a mobile agent run environment.

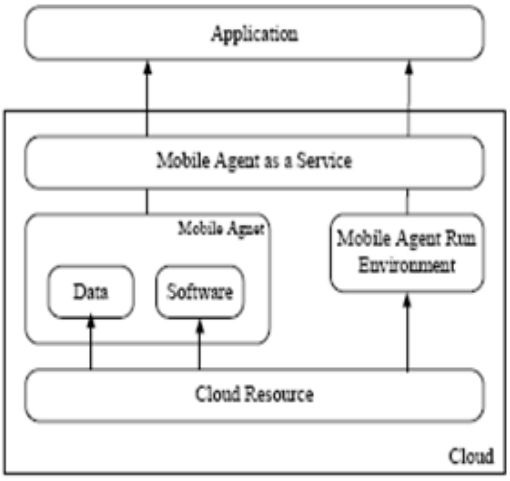


Figure 4. Architecture of Mobile Agent based Cloud Computing System

In SaaAS, mobile agent is in charge of not only the migration of software and data but also the system management. As Fig. 5 showing, in SaaAS, all agents can be classified as following: (1) IA (Interface Agent). IA runs on the user host and interaction with the user. Then, IA processes computing task by dispatching, controlling or coordinating with other agents. (2) WA (Working Agent). WA accepts the instruction from IA, and packs the software and the data to move to the target domain server to execute the operation, and then return the result of execution. (3) DMA (Domain Manage Agent). DMA is responsible for domain management in a domain. DMA can duplicate itself and actively move to target server in order to be close to the data to gain higher processing performance. (4) MMA (Main Management Agent). MMA is responsible for the management and coordination of all DMAs in SaaAS. MMA and DMA can cooperate with each other to accomplish the management in SaaAS.

The advantages in the architecture of SaaAS are as following: (1) DMA is responsible for the management of a domain and MMA is responsible for the management of all the DMAs. This architecture can share all the overheads of migration and management over all DMA, and avoid the single central server to be the bottleneck of system. (2) Users in a Domain are connected with each other through high bandwidth LAN which has wide bandwidth and short transfer latency. Therefore, the communication in domain can gain a better performance by using the protocol designed for LAN. When the mobile agent which contains the software and the data migrates across domains, SaaAS could use the special protocols and mechanisms that are designed for WAN to gain better performance. (3) The cooperation between the agents in domain can effectively improve the efficiency of data coherence and service migration. The (2) and (3) will be discussed in the section 3 and section 4.

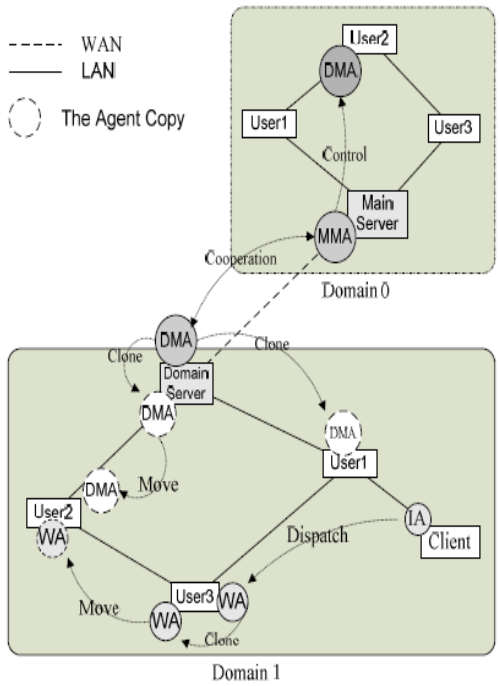


Figure 5:Work Division and Cooperation of Agent in SaaAS

III. CODE AND DATA OF SERVICE LOAD MECHANISM BASED MOBILE AGENT

In SaaAS, the software and data migration of is always by means of the code migration of the agent. Paper [12] analyzed the performance of three agent platform, and find that the load latency has close relationship with the runtime code loading mechanism. If all the runtime code is involved in agent, the load latency will be very low, but the network communication overhead will be high. If only part of runtime code is involved in agent, agent must load the runtime code firm code server and make the load latency too high, but the network communication overhead will be low. A compromise mechanism is put the agent common code server into every domain. When agent wants to load the runtime code, it can get the codes from the code server in local domain.

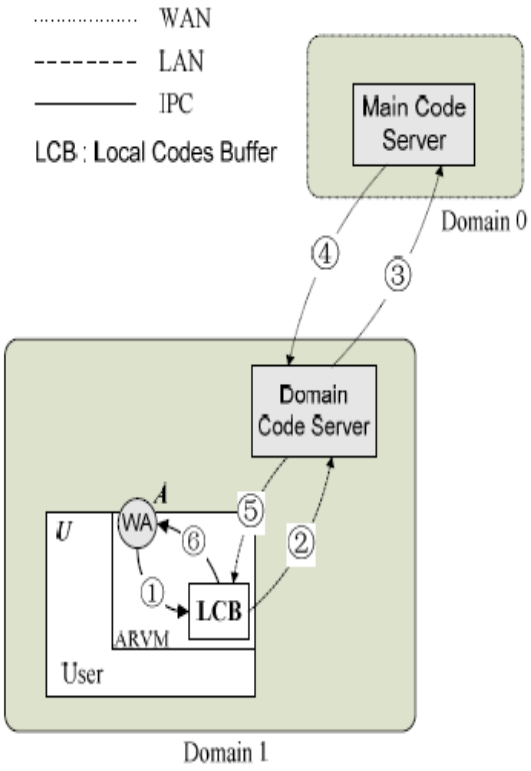


Figure 6. Runtime Codes Load Mechanism

In SaaAS, there is a MCS (Main Code Server), and every domain has a DCS (Domain Code Server). All the runtime codes of agent are stored in MCS, and DCS can get the code from MCS and stores them in local buffer. These codes in DCS can be used by the agents in own domain. Besides MCS and DCS, every client also can cache the recently used code to the agent that runs in this client's device. As showing in Fig. 6, in the worst condition, the code an agent need is not on both local host and DCS, so the code must load from MCS.

Let agent A that runs in user U want load the codes, this load process has six steps as following: (1) A checks the code buffer of U where has the runtime code; (2) the U request the codes to DCS; (3) DCS request the codes to MCS; (4) MCS return the codes to DCS; (5) DCS returns the codes to the U; (6) U return the codes to A. In the steps described above, steps (3) and (4) run in WAN; steps (2) and (5) run in LAN, step (1) and (6) run in U by IPC (Inter-Process Communication). Because the codes that agent require will be cached in buffer on client and DCS and to reuse by other agent, then at most commonly conditions, only step (1) and (6) are required (Loading From Local Client), or only step (1),(2),(5),(6) are required (Loading From Local Domain). When agent loads code from local code server in LAN, the latency is much lower than loading code in WAN. From the discussion above, we can find that MCS, DCS and host are composed into a dynamic hierarchical code loading mechanism. This mechanism can make mobile agents work as best as it can to load codes from local host or local domain, and then the load latency is low. Meanwhile, this mechanism avoids the heavy communication overhead in internet.

IV. DIVIDED-CLOUD AND CONVERGENT COHERENCE MECHANISM

SaaAS is designed for Internet, and then the design goal of SaaAS is not only can reduce the traffic in network, but also can support the UNIX semantic when access the data in cloud computing system. For that purpose, a Divided-Cloud and Convergent Coherence Mechanism (DCCM) is presented in SaaAS. DCCM can reduce the communication in WAN, with supporting UNIX semantic. In SaaAS, every data has the "Read Lock" and "Write Lock". The read or write operation only can be performed if a user has obtained the read lock or write lock. A user can cache the data in local buffer if it obtain the read lock, and can perform arbitrarily write operation without communication with server if obtaining the write lock. The server can not only assign a data lock, but also can require the user to release the lock. When releasing a read lock, user must invalidate its own local data cache. When releasing a write lock, user must write the dirty data back to the server. There is only one Main Consistence Server (MCS) in SaaAS, and each domain has a Domain Consistence Server (DCS). The cooperation between MCS and DCS can guarantee to achieve the cache coherence in SaaAS.

An important work of DCS is to converge the lock request of all users in own domain. Converging can reduce the communication in WAN and improve the performance of cache coherence management. In SaaAS, every DCS has a lock table, with recording the all read and write lock that it has. Let's suppose that user A in a domain wants to obtain the read lock of data D. So, user A firstly sends a lock request to DCS. After receiving the request, DCS checks own lock table to find whether it already has this read lock. There may be two conditions: (1) if DCS find that it does not have the read lock for D, then it will send this request to MCS. After MCS receiving this request, it will assign the read lock to DCS and DCS will send this lock to user A. After that, DCS will record this lock in own lock table. While user A gets the read lock, A can retrieve the data from remote server and store it into local buffer. (2) DCS find that the user B in this domain already has the read lock for D. So, DCS will not send a request to MCS to apply the read lock, but only directly assigns this read lock to user A, and then add a record that user A has the read lock for D. While getting this read lock, user A is not necessary to retrieve D from remote server, and can only copy the cache of D from the buffer of user B. From the discussion above, we can find that DCS can make the lock request in a domain converged. That is to say, if many users in a domain request the read lock for a same data, the DCS only will send one request to MCS when the first user sends its request, and the second user request the read lock, the lock requesting and buffer copying can be all completed in the domain.

The lock converging capability of DCS has a lot of advantages: (1) reducing the network communication for managing data lock and duplicating the data buffer, particularly the communication in WAN; (2) MCS do not need to maintain the lock state of every user, but only need to maintain the lock state of all DCS. Hence, converging can reduce effectively the overload of the maintaining the lock state. The divided-cloud and convergent coherence mechanism of SaaAS not only has the advantages above, but also uses the specialty of mobile agent to improve the performance. As figure 7 showing, let User5, User6 and User7 have the read lock of D, and at this moment User3 want to apply the write lock of D.

The whole process of applying lock and releasing lock is as follow: (1) User3 firstly creates an agent A, and then dispatches A to DCS1 with the task of applying the write lock of D. (2) When A arriving at DCS1, A will check the lock record table of DCS1, and finds that there is no user that has the write lock of D. Thus, A moves to MCS, and applies the write lock to MCS. (3) When arriving at MCS, A will check the lock table on MCS. Let's suppose that A finds DCS1 and DCS2 have the read lock of D. According as the multiplereading/single-writing protocol, DCS1 and DCS2 must release the read lock, thus A duplicates itself into two agents (assume they are A1 and A2), and A1 move to DCS1, A2 move to DCS2, showing as (3) and (3) in Fig. 7. (4) After A1 arriving DCS1, it will check the lock table of DCS1, and finds the User5 in domain1 has the read lock of D. Then, A1 moves to User5, and requires User5 to release the read lock. User5 will release the read lock as soon as receiving the request form A1.

While receiving the acknowledgement of releasing read lock from User5, A1 will move back to DCS1, and then move back to MCS. A2 does the same process in domain2, and moves back to DCS2, MCS after receiving the releasing lock acknowledgement from User6 and User7. (5) A1 and A2 exchange the information while arriving at MCS and A2 will terminate after having exchanged information with A1. A1 deletes the records that DCS1 and DCS2 have the read lock of D, and add a record that DCS1 has the write lock of D. After that, A1 moves to DCS1. (6) While A1 arriving DCS1, it will add a record in the lock table of DCS1 that User3 has the write lock of D. (7) A1 moves back to User3 and informs user3 that the write lock request of D is success. While User3 getting the write lock, it will retrieve the data of D from the remote server and save the data in local buffer. Then, User3 can read and write D in local buffer without communicating with server.

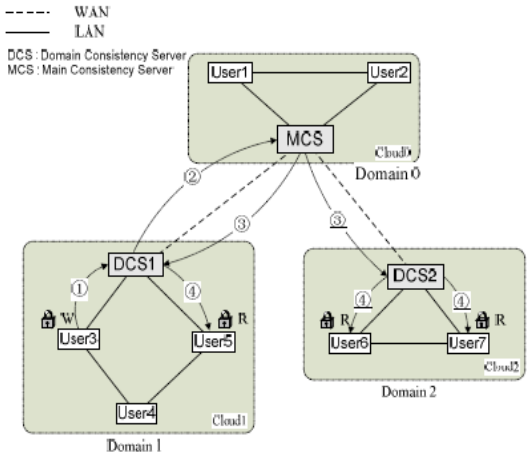


Figure 7: Divided-Cloud and Convergent Coherence Mechanism

From the steps described above, we can find that SaaAS can efficaciously reduce the communication overload in WAN and the cache coherence management over-

head in DCS and MCS by using the mobility, reliability and cooperatility of mobile agent.

CONCLUSION

For the unstable bandwidth and long transfer delay of Internet, the mobile agent is a better underlying facility to implement the software and data migration in wide area cloud computing system. In this paper, The Mobile Agent based Service for Cloud Computing in Internet Environment (SaaAS) is presented. The code and data of service load mechanism based mobile agent and dividedcloud and convergent coherence mechanism of SaaAS are also discussion in this paper.By using the agent rather than RPC/RMI as the underlying facility to implement the software and data migration, SaaAS is more suitable to the cloud computing system which works in Internet. In conclusion, the application of mobile agent enables SaaAS to have good flexibility, daptability and usability and to be more suitable to work in Internet than conventional cloud Computing system.

REFERENCES

[1] IBM Blue Cloud project [URL]. http://www-03.ibm.com/press/us/en/pressrelease/22613.wss/.

[2] Lijun Mei, W.K. Chan, and T.H. Tse, "A tale of clouds: paradigm-comparisons and some thoughts on research issues", in Proceedings of2008 IEEE Asia-Pacific Services Computing Conference (APSCC2008), pp.464-469.

[3] Cloud computing. Wikipedia. Available at http://en.wikipedia.org/wiki/Cloud_computing.

[4] R. Wang, T. Anderson and M. Dalin, “Experience with a distributed filesystem implementation”, Technical report, University of California,Berkeley, Computer Science Division, June 1997.

[5] L. Ismail and D. Hagimont, “A Performance Evaluation of the MobileAgent Paradigm”, ACM SIGPLAN Notices, October 1999, 34(10),pp.306-313.

[6] K. Hartig. What is cloud computing? SOA World Magazine. Availableat http://soa.sys-con.com/read/ 579826.htm.

[7] C. Lee, S. Ko, S. Lee, W. Lee, and S. Helal. Context-aware service-composition for mobile network environments. In UbiquitousIntelligence and Computing, volume 4611 of Lecture Notes in ComputerScience, pages 941–952. Springer, Berlin, Germany, 2007.

[8] L. Ismail and D. Hagimont, “A Performance Evaluation of the MobileAgent Paradigm”, ACM SIGPLAN Notices, October 1999, 34(10),pp.306-313.