TELKOMNIKA, Vol.11, No.11, November 2013, pp. 6964~6971
e-ISSN: 2087-278X

A Novel Wisdom Network Design and Its Principles

Fuhong Lin*[1], Senqing Lin[2], Xianwei Zhou[1], Qian Liu[1], Wei Song[3]
[1]School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, 100083, P. R. China
[2]Institute of Physics and Information Engineering, Shanxi Normal University, Shanxi 041000, P. R. China
[3]School of Information Engineering, Minzu University of China, Beijing 100081, P. R. China
*Corresponding author, e-mail: FHLin_ustb@yeah.net

Abstract

Next generation network is a hot topic that lots of researchers have paid their attention to this point. The main idea is making people more comfortable and convenient. This paper presents a novel wisdom network which tends to achieve the above mentioned advantages. This network is composed by User Part and Server Part. Further, Server Part is divided into Gateway Server part and Data Server part. According to the network architecture, some design principles are analyzed and modeled, including the following aspects: resource registration and retrieval, data pieces replication, Gateway Server performance and load balance achievement.

Keywords*: wisdom network, resource registration and retrieval, data replication, load balance*

1. Introduction

The development of network architecture can be divided into three stages. In the first stage, C/S model is the frequently-used, and an example is traditional Internet [1-8]. In the traditional Internet, all the resources are stored in the server part which is formed by high-powered computers, and clients who need certain resources need to contract with the servers to retrieve them. The advantage is that users need not store any resource, while the disadvantage is as follows: firstly, servers needs high-powered and high-bandwidth, secondly, some privacy or security problems appear, such as DoS attack.

To reduce the burden of servers, the second stage is emerged. In this stage, collaboration is the mainline and an representative example is peer-to-peer network in which all resources are stored in own terminal and the server-part is discharged [9-17]. Person, who needs resources, uses a peer-to-peer search algorithm locating the proper resources. The advantage is that high-power and high-bandwidth servers are disappeared, while the disadvantage is that privacy or security problems still have not well solved and users need to afford lots of computing resources which is not accord with the development of user terminal equipments. Taking mobile phone as an example, it does not have the ability to deal with some high-bandwidth consumption usage.

Recently, a new topic is proposed which aims at combing the advantage of C/S model and peer-to-peer model. And it can be treated as the third stage. The representing network is cloud computing network [18-24] or information-centric network [26-30]. These networks have the following characteristics. Firstly, the architecture is a C/S(O) model in which C represents clients who need not store anything and only need to contract to S(O) to retrieval resource, and S(O) means servers who are composed by a large number of ordinary computers and they use certain algorithm to store resources. Secondly, when the servers store a certain resource, the resource is split into many pieces and these pieces are registered in different servers which can copy with the privacy and security attacks.

In this paper, we focus on the future network which we call wisdom network. We borrow some ideas from the above mentioned architecture and propose that the wisdom network is composed by User Part and Server Part, and the Server Part can be divided into Gateway Server system and Data Server system. Then, we analyze some design principles related to the wisdom network. Firstly, we explain how resources are registered or retrieved in this network. Secondly, we introduce how to replicate data pieces and analyze the performance of data

Received June 4, 2013; Revised August 1, 2013; Accepted August 14, 2013

pieces replication. Thirdly, we use queue theory to research on the performance of Gateway Servers. Lastly, load balance is study using game theory.

This paper is organized as follows. In Section 2, the wisdom network architecture is proposed. Some design principles are analyzed in Section 3, including resource registration and retrieval, data pieces replication, Gateway Server performance and load balance achievement. A conclusion is drawn in Section 4.

2. Wisdom Network Architecture

In this section, we give a brief description of our designed network. The idea of wisdom network architecture is a combination of advantages of traditional Internet architecture, peer-to-peer network architecture, cloud computing architecture and information-centric network architecture. It can be describe as definition 1and Figure 1.

Definition 1. A wisdom network WN is composed by two parts which are User Part and Server Part. The User Part is a set of end users $UP = \{u_1, ..., u_n\}\ n \in N^+$. The Server Part is a set of Gateway Servers and Data Servers $SP = \{g_1, ..., g_l; d_1, ..., d_k\}\ l, k \in N^+$. Where g_i represents the i 'th Gateway Server and d_i represents the i 'th Data Server.

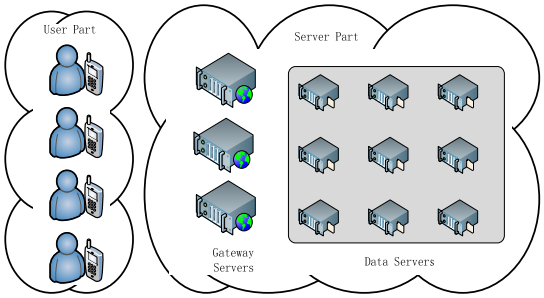

Figure 1. The Wisdom Network Architecture

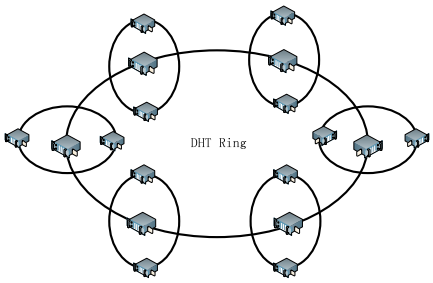

Figure 2. The Structure of Data Server System

For each user $u_i \in UP$, an online account is opened for him. Using this account, he can access to the Server Part and register or retrieval resources. So he need not store any resource which can shift afford to the Server Part and save store and computing resource himself. Weak terminals, such as mobile phone and tablet computer, can be well accessed.

As defined in definition 1, Server Part is composed by Gateway Servers and Data Servers. The functionality implement is defined in definition 2 and a brief explanation of Data Servers is shown in Figure 2.

Definition 2. The implement of Server Part is as following. Gateway server g_i, $i \in N^+$ connects to a set of Data Servers D^S, $D^S = \{d_1 d_2, ..., d_k\}$, $k \in N^+$. Data Servers form a DHT ring DR, $DR = \{d_1^r d_2^r, ..., d_k^r\}$, $k \in N^+$ and a backup mechanism is used in each DHT ring point.

Gateway servers are in charge of the following tasks.

(1) Registration issues. When gateway server gets a register request, it splits the resource, and register these piece in the Data Servers;
(2) Retrieval issues. When a user want to get a certain resource, the gateway server needs to find the proper one in the Data Servers and return it to the user;
(3) Privacy issues. As mentioned in registration issues, the resource will be split, so no attacker can obtain the integrated resource;
(4) Security issues. When a user want to access to the Server Part, Gateway Servers will check the legality of each user;
(5) Load balance issues. When the number of retrieval request is very large, Gateway Servers need to balance the load among Data Servers;

The Data Servers are in the charge of storing data pieces and there are two key points we should pay attention to. The first one is that all the Data Servers are ordinary computer and they are easily broken down, so how to cope with single point failure is a design point. Secondly, there may be some hot resources that are frequently being searched, so we need design a replication strategy to deal with this hot point issue.

3. Design Principle

3.1. Process of Resource Registration and Retrieval

In this section, we just detailed explain how a resource is registered and retrieved. Firstly, we study on the resource registration. The flow chart is shown in Figure 3(a). The user who wants to register a resource sends a registration request to one of the Gateway Server who is connected to him. After checking the legality of the user, the Gateway Server splits the resource R into n data pieces DP_i, $R = \bigcup_{i=n} DP_i$. Using Chord algorithm which is a peer-to-peer network [9], Gateway Server registers each data piece on a certain Data Server DS_i and a set of another Data Servers.

$$\begin{bmatrix} DP_1 \\ DP_2 \\ \cdots \\ DP_n \end{bmatrix} \xrightarrow{Mapping} \begin{bmatrix} DS_1 \\ DS_2 \\ \cdots \\ DS_m \end{bmatrix}$$

The retrieval process is opposite to the registration process and the flow chart is shown in Figure 3(b). If a user wants to get a resource R, he sends a retrieval request to a Gateway Server. After checking the legality of the user, the Gateway Server abstracts the resource ID and retrieves each data pieces DP_i and replies it to the user.

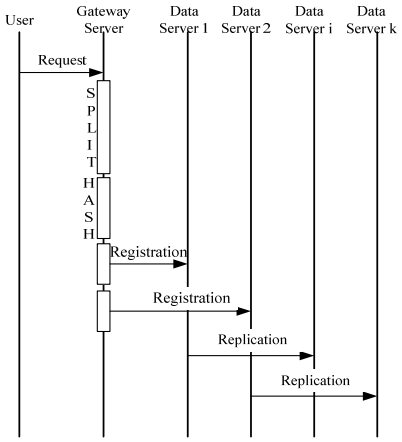

Figure 3(a). The Registration Process

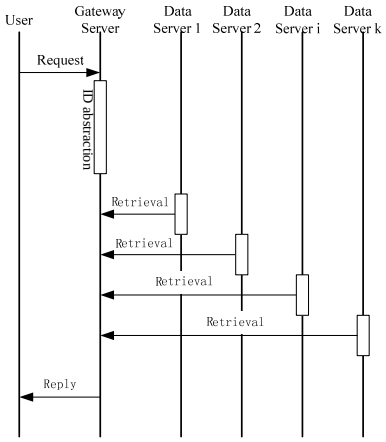

Figure 3(b). The Retrieval Process

3.2. Process of Data Replication

We mentioned that all the data pieces are stored in the Data Servers which are composed by a large number of ordinary computers who have not strong computing ability and easily break down. So we need to design certain algorithm to copy with nodes failures.

We use three methods to accomplish this design. Firstly, hot backup mechanism is used. As shown in Figure 2, each Data Server node is hot backup by the other two Data Servers. Secondly, the Chord algorithm itself owns the ability to copy with node broken down. When storing a data, Chord system not only stores the data on the node who is responsible to, but also stores the data on the following m nodes($m = \log N$). Where N represents the number

of nods in the Chord system and $N \in N^+$. Thirdly, path's replication is used. Path's replication is introduced in [31], and tells the story of replicating data on the nodes which were passed when resource was retrieval in order to increasing the number of data in the system. Next we give a theorem to analyze the data robust.

Theorem 1. In the Data Server system, the probability of a node being alive at a certain time is p_n, $p_n \in R^+$. The number of Data Server is N, $N \in N^+$. The number of path's replication of data is pr, $pr \in N^+$. There exit a success retrieval probability P_s,($P_s \in R^+$), which can be written as:

$$P_s = 1-(1-p_n)^{3+\log N+E(pr)} \tag{1}$$

Proof. There are three replication methods. Using the first method, we get the unsuccessful probability of data piece retrieval being $(1-p_n)^3$. According to the second method, the unsuccessful probability of data piece retrieval is $(1-p_n)^{\log N}$. For rhe third method, we cannot know the number of path's replication of a certain data. However, we can use the history to estimate. So the failure probability writes as $(1-p_n)^{E(pr)}$. Combining these three unsuccessful probability, we can get the outcome.

3.3. Process of Gateway Servers' Abilities

In section 2, we mentioned that Gateway Servers need to do lots of tasks. So a performance threshold needs to be gotten in order to give some advice on the deploy phase. Generally speaking, the procedure of Gateway Server serving is receiving a request, processing it and then passing it to the Data Servers. The characteristic of this event is a queue process. So we use queue theory to model Gateway Servers' abilities. We assume that serving request arrivals follows Poisson distribution, and the arrival rate is λ. Assuming there are M Gateway servers each of which has the service rate of μ. The buffer length of each Gateway server is k which means that if the number of serving request is more than k, additional serving requests are dropped. For the buffer length is dynamic, so we use $k(d)$ as our buffer length. A classical queue theory $M/M/n/m$ can be used here to do our research [32]. According to the above description, we can get the state transition process in Figure 4.

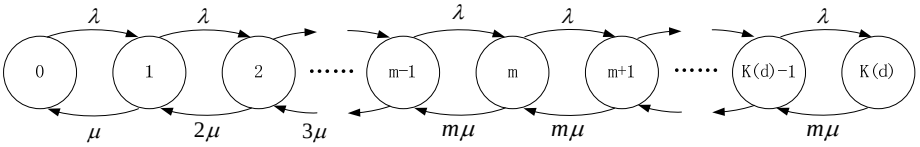

Figure 4. The State Transition Process

Theorem 2. In the Gateway Server system, the load is $\rho = \lambda/\mu$, $\rho \in R^+$. The number of server is m, $m \in N^+$. The buffer length is $k(d)$, $k(d) \in N^+$. The probability of all the servers being free can be written as:

$$p_0 = \begin{cases} [\sum_{i=0}^{m-1} \frac{(m\rho)^i}{i!} + \frac{(m\rho)^m}{m!} \times \frac{1-\rho^{k(d)-m+1}}{1-\rho}]^{-1} & \rho \neq 1 \\ [\sum_{i=0}^{m-1} \frac{m^i}{i!} + \frac{m^m}{m!} \times (k(d)-m+1)]^{-1} & \rho = 1 \end{cases} \tag{2}$$

Proof. According to the state transition process in Figure 4, we can get the equation of steady state distribution being:

$$\begin{cases} \lambda p_i = (i+1)\mu p_{i+1} & 1 \le i \le m \\ \lambda p_i = m\mu p_{i+1} & m < i < k(d) \end{cases} \quad (3)$$

Owing to $\sum_{i=0}^{k(d)} p_i = 1$, we can get the conclusion in theorem 2.

Lemma 1. In the Gateway Server system, the average process time of a serving request T can be written as:

$$T = \begin{cases} \dfrac{m^m \rho^{m+1}[1-(k(d)-m+1)\rho^{k(d)-m}+(k(d)-m)\rho^{k(d)-m+1}]p_0}{m!(1-\rho)^2\mu[m-\sum_{i=0}^{m-1}(m-i)p_i]} + \dfrac{1}{\mu} & \rho \ne 1 \\ \dfrac{m^m p_0 (k(d)-m)(k(d)-m+1)}{2m!\mu[m-\sum_{i=0}^{m-1}(m-i)p_i]} + \dfrac{1}{\mu} & \rho = 1 \end{cases} \quad (4)$$

Proof. The serving time can be divided into two parts: queue time delay and request process time delay. The average request process time delay can be written as $T_q = 1/\mu$. Using little theorem, the average queue time delay can be written as:

$$T_w = \begin{cases} \dfrac{m^m \rho^{m+1}[1-(k(d)-m+1)\rho^{k(d)-m}+(k(d)-m)\rho^{k(d)-m+1}]p_0}{m!(1-\rho)^2\mu[m-\sum_{i=0}^{m-1}(m-i)p_i]} & \rho \ne 1 \\ \dfrac{m^m p_0 (k(d)-m)(k(d)-m+1)}{2m!\mu[m-\sum_{i=0}^{m-1}(m-i)p_i]} & \rho = 1 \end{cases} \quad (5)$$

Combining these two time delay, we can get the conclusion in lemma 1.

Lemma 2. In the Gateway Server system, the probability that the serving request cannot be processed P_l can be written as:

$$P_l = \begin{cases} \dfrac{\frac{m^m \rho^{k(d)}}{m!}}{\sum_{i=0}^{m-1}\frac{(m\rho)^i}{i!} + \frac{(m\rho)^m}{m!}\times\frac{1-\rho^{k(d)-m+1}}{1-\rho}} & \rho \ne 1 \\ \dfrac{\frac{m^m \rho^{k(d)}}{m!}}{\sum_{i=0}^{m-1}\frac{m^i}{i!} + \frac{m^m}{m!}\times(k(d)-m+1)} & \rho = 1 \end{cases} \quad (6)$$

Proof. When the number of serving requests bigger than $k(d)$, there is not space for storing and being dropped. So P_l equals to the $p_{k(d)}$, which means all the probability of all the Gateway Servers are busy. Using formula (2) and (3), P_l can be written as:

$$P_l = p_{k(d)} = \frac{m^m \rho^{k(d)}}{m!} \times p_0 \quad (7)$$

Then we can get the conclusion in lemma 2.

3.4. Process of Load Balance in Data Server System

All the data pieces are stored in the Data Server system, including the data pieces themselves and corresponding replications. So there exists a point that how many data pieces

one Data Server should store to achieve load balance. We just use game theory to solve this problem. The parameters are defined as following:

j , $j \in [1,n]$ The j'th Data Server;

s_i , $i \in N$ The number of data pieces stored in one Data Server;

$S = \sum_{i=1}^{n} s_i$ The overall number of data pieces stored in Data Server system;

$\overline{S}$ The theoretical number of data pieces a Data Server system can store;

c The consuming of each data piece, including space, energy and so on;

$H = \alpha(\overline{S} - S)$ The number of request of each data pieces is hit in a unit of time, and α , $\alpha > 0$ being the constant factor;

$c_r = \beta H$ The consuming of ROM of each data pieces and β , $\beta \in (0,1)$ being the constant factor.

Using the parameters above, we can build our model for Data Server j . The payoff of each Data Server j can be written as:

$$u_j = \alpha \times (\overline{S} - \sum_{i=1}^{n} s_i) \times s_j - c \times s_j - c_r \times s_j \text{ , } j \in [1,n] \tag{8}$$

Theorem 3. In the Data Server system, Data Server j , $j \in [1,n]$ has an optimal storing point on which every Data Server could achieve the best payoff. The optimal storing point and payoff can be written respectively as:

$$s_j^* = \frac{\alpha(1-\beta)\overline{S} - c}{\alpha(1-\beta)(1+n)} \tag{9}$$

$$u_j^* = \frac{(\alpha(1-\beta)\overline{S} - c)^2}{\alpha(1-\beta)(1+n)^2} \tag{10}$$

Proof. Using the game theory on Nash equilibrium, we can find an optimal strategy $s^* = (s_1^*, s_2^*, ..., s_j^*, ..., s_n^*)$, which follows the following formula [33]:

$$u_j(s_1^*, s_2^*, ..., s_{j-1}^*, s_j^*, s_{j+1}^*, ..., s_n^*) \geq u_j(s_1^*, s_2^*, ..., s_{j-1}^*, s_j^{'}, s_{j+1}^*, ..., s_n^*) \tag{11}$$
$$\forall s_j^{'} \in S_j \text{ , } s_j^{'} \neq s_j^* \text{ , } \forall i \in N$$

In order to solve this problem, we take the first derivative of formula (8) and get:

$$\alpha(1-\beta)\overline{S} - c - \alpha(1-\beta)\sum_{i \in N \setminus \{j\}} s_i^* - 2\alpha(1-\beta)s_j^* = 0 \text{ , } j \in [1,n] \tag{12}$$

Then we can get the equation:

$$s_j = \frac{\alpha(1-\beta)\overline{S} - c - \alpha(1-\beta)\sum_{i \in N \setminus \{j\}} s_i^*}{2\alpha(1-\beta)} \text{ , } j \in [1,n] \tag{13}$$

In order to check the concavity, we take the second derivative of formula (8) and get:

$$-2\alpha(1-\beta) < 0 \tag{14}$$

So we can get the optimal number of pieces each Data Server needed to store is:

$$s_j^* = \frac{\alpha(1-\beta)\overline{S} - c}{\alpha(1-\beta)(1+n)} \quad (15)$$

Then we can get the optimal payoff of each Data Server is:

$$u_j^* = \frac{(\alpha(1-\beta)\overline{S} - c)^2}{\alpha(1-\beta)(1+n)^2} \quad (16)$$

These two outcomes are the conclusion in theorem 3.

4. Conclusion

In this paper, we proposed a new network architecture called wisdom network which is composed by User Part and Server Part. The Server Part is divided into Gateway Server system and Data Server system. The advantage of this design is that users just need to submit their requests and the Server Part needs to store all the resources and solve the requests submitted by clients. Then some design principles were analyzed and modeled. The function of resource registration and retrieval was accounted. Path's replication was used in resources replication and a retrieval failure model is proposed. The performance of Gateway Server abilities was modeled to show the factors affecting the performance of wisdom network. Load balance was achieved by data pieces storing strategy according to game theory.

Acknowledgments

We gratefully acknowledge anonymous reviewers who read drafts and made many helpful suggestions. This work is supported by the Project supported by the Foundation for Key Program of Ministry of Education, P. R. China (No.311007), National Science Foundation Project of P. R. China (61170014, 61202079), China Postdoctoral Science Foundation (2013M530526), and the Fundamental Research Funds for the Central Universities (FRF-TP-09-015A, FRF-TP-13-015A).

References

[1] Hung-Sheng Chiu, Chyan Yang: *Migrating Client/Server Architecture Towards Internet*. PDPTA 2000.
[2] Yiwei Thomas Hou, Dapeng Wu, Bo Li, Takeo Hamada, Ishfaq Ahmad, H. Jonathan Chao: *A differentiated services architecture for multimedia streaming in next generation Internet.* Computer Networks (CN). 2000; 32(2): 185-209.
[3] Alencar de Melo Jr, Juan Manuel Adán Coello. *Packet Scheduling Based on Learning in the Next Generation Internet Architectures*. ISCC. 2000: 773-778.
[4] Hung-Sheng Chiu, Chyan Yang. *Migrating Client/Server Architecture Towards Internet.* PDPTA. 2000.
[5] Kambiz Asrar-Haghighi, Yaser Pourmohammadi, Hussein M Alnuweiri. *Realizing MPEG-4 Streaming Over the Internet: A Client/Server Architecture Using DMIF*. ITCC. 2001: 23-29
[6] Björn Knutsson, Honghui Lu, Jeffrey C Mogul, Bryan Hopkins. *Architecture and performance of server-directed transcoding.* ACM Trans. Internet Techn. (TOIT) 2003; 3(4): 392-424.
[7] Purvi Sheth. Book Review: *A Great Reference Book for Networking Professionals (A review of Internetworking with TCP/IP Principles, Protocols, and Architectures, 4th edition by Douglas E. Comer).* IEEE Distributed Systems Online (DSONLINE). 2002; 3(6).
[8] Wei Kang Tsai, Mahadevan Iyer, Jordi Ros. Revisit the Strings versus Clouds Debate for the Internet Architecture - Part II: QoS, Control, Management, and TCP*. Journal Network Syst. Manage. (JNSM).* 2002; 10(3): 267-275.
[9] Ion Stoica, Robert Morris, David R Karger, M Frans Kaashoek, Hari Balakrishnan. *Chord: A scalable peer-to-peer lookup service for internet applications*. SIGCOMM. 2001:149-160.
[10] icholas Nicoloudis, Christine Mingins Psachno. *A Dynamic and Generic Discovery Framework within a Peer-to-Peer Network Model*. APSEC. 2003: 542-551.
[11] Matei Ripeanu. *Peer-to-Peer Architecture Case Study: Gnutella Network.* Peer-to-Peer Computing 2001: 99-100.
[12] Georges Da Costa, Olivier Richard: *Impact of a Realistic Workload in Peer-to-Peer Systems. A Case Study: Freenet*. Sci. Ann. Cuza Univ. (CUZA). 2002; 11: 259-270.

[13] Petar Maymounkov, David Mazières. *Kademlia: A Peer-to-Peer Information System Based on the XOR Metric.* IPTPS. 2002: 53-65.
[14] Dongyu Qiu, Rayadurgam Srikant: *Modeling and performance analysis of BitTorrent-like peer-to-peer networks.* SIGCOMM. 2004: 367-378.
[15] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard M Karp, Scott Shenker. *A scalable content-addressable network.* SIGCOMM. 2001: 161-172.
[16] Ben Y Zhao, John Kubiatowicz, Anthony D Joseph. *Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing*. No.UCB/CSD-01-1141. University of California Berkeley.
[17] Antony Rowstron, Peter Druschel. *Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer system.* IFIP/ACM International Conference on Distributed Systems Platforms. 2001.
[18] Liang-Jie Zhang, Qun Zhou. *CCOA: Cloud Computing Open Architecture.* ICWS. 2009: 607-616.
[19] Jeff Sedayao. *Implementing and operating an internet scale distributed application using service oriented architecture principles and cloud computing infrastructure.* iiWAS. 2008: 417-421.
[20] Bu-Qing Cao, Bing Li, Qi-Ming Xia. *A Service-Oriented Qos-Assured and Multi-Agent Cloud Computing Architecture.*CloudCom. 2009: 644-649.
[21] Vijay Sarathy, Purnendu Narayan, Rao Mikkilineni: *Next Generation Cloud Computing Architecture: Enabling Real-Time Dynamism for Shared Distributed Physical Infrastructure*. WETICE. 2010: 48-53.
[22] Zhenyu Fang, Changqing Yin. *BPM Architecture Design Based on Cloud Computing.* Intelligent Information Management (IIM). 2010; 2(5): 329-333.
[23] Weider D Yu, Radhika Bhagwat. *Modeling Emergency and Telemedicine Heath Support System: A Service Oriented Architecture Approach Using Cloud Computing.* IJEHMC. 2011; 2(3): 63-88.
[24] Abdulrahman Almutairi, Muhammad Sarfraz, Saleh Basalamah, Walid G Aref, Arif Ghafoor. *A Distributed Access Control Architecture for Cloud Computing.* IEEE Software (SOFTWARE). 2012; 29(2): 36-44.
[25] Konstantinos V Katsaros, George Xylomenos, George C Polyzos. *MultiCache: An overlay architecture for information-centric networking.* Computer Networks (CN). 2011; 55(4): 936-947.
[26] Alexandros Kostopoulos, Ioanna Papafili, Costas Kalogiros, Tapio Levä, Nan Zhang, Dirk Trossen. *A Tussle Analysis for Information-Centric Networking Architectures.* Future Internet Assembly. 2012: 6-17.
[27] George Pavlou. *Keynote 2: Information-centric networking: Overview, current state and key challenges*. ISCC. 2011.
[28] Muhammad Shoaib Saleem, Eric Renault, Djamal Zeghlache: *Information Centric Networking based Handover Support for QoS Maintenance in Cooperative Heterogeneous* Wireless Networks CoRR abs/1108.3239. 2011.
[29] Ali Ghodsi, Scott Shenker, Teemu Koponen, Ankit Singla, Barath Raghavan, James Wilcox. *Information-centric networking: seeing the forest for the trees*. HotNets. 2011.
[30] Xicheng Lu, Qianbing Zheng, Peidong Zhu, Wei Peng: *LQPD: An Efficient Long Query Path Driven Replication Strategy in Unstructured P2P Network*. ICN. 2005: 793-799.
[31] Robert B Cooper. *Introduction to Queueing Theory.* 2nd Edition. Elsevier/North-Holland, Amsterdam, 1981.
[32] David WK Yeung, Leon A Petrosyan, Markus CC Lee. *Dynamic Cooperation: A Paradigm on the Cutting-edge of Game Theory.* China Market Press. 1997.