IEICE TRANS. COMMU. VOL.E87-B, NO 3. MARCH 2004

PAPER Special Issue on Internet Technology IV

# Scalable Multilateral Autonomous Decentralized Community Communication Technique for Large-Scale Information Systems

*Khaled Ragab*[*], *Naohiro Kaji*[*], *Nonmembers, and Kinji Mori*[*], *Regular Member*

**SUMMARY** Autonomous Decentralized Community Information System (ADCS) is a proposition made to meet the rapidly changing users' requirements and cope with the extreme dynamism in current information services. ADCS is a decentralized architecture that forms a community of individual end-users (*community members*) having the same interests and demands in specified time and location. It allows those members to mutually cooperate and share information without loading up any single node excessively. In this paper, an *autonomous decentralized community communication technology* is proposed to assure a productive cooperation, a flexible and timely communication among the community members. The main ideas behind this communication technology are: content-code communication (service-based) for flexibility and multilateral communication for timely and productive cooperation among members. All members communicate productively for the satisfaction of all the community members. The scalability of the system's response time regardless of the number of the community members is shown through simulation. Thus, the autonomous decentralized community communication technology reveals significant results when the total number of members in the community increases sharply.

***Key words:*** *Autonomous Decentralized Community Information System, Multilateral Community communication, large-scale Information systems.*

## 1. Introduction

The Internet's phenomenal impact, the subsequent growth and the evolving in social and economic environments promote more severe and complex requirements for the information service systems. Current Internet information services are provided for anyone, anywhere, anytime. These systems are constructed from the service providers (SP)' point of view. SPs provide information regardless of the end-users' demands and situations. There is no discernment between differences in place and time; end-users in any situation receive the same contents. In addition, end-users know in advance what content will satisfy their demands and then access the SP to obtain it. In a rapidly changing environment, the large-scale information systems are confronted to some challenges. First, the number of worldwide Internet and mobile users are predicted to exceed 1 billion by the end of 2005 [1]. Those users have rapidly, and dynamically changing demands and interests. Second, about 300 terabytes of information every year the world publishes on-line [2]. Constantly, new information services are added, others are modified, removed or in fault, making it more and more intractable to maintain a coherent image of the information environment. Therefore, customizing the service to the end-users is increasingly difficult, whereas end-users require well-customized, timely, continual, reliable, and available information services [3]. In addition, under the evolving situations they have heterogeneous and dynamically changing requirement levels of timeliness [4]. Timeliness is an essential component in modern high-assurance and large-scale information systems [5].

As the end-users demands are dynamically changing, anywhere or somewhere at specified time there are significant numbers of users sharing the same interests and demands. Consequently, a rapid and dramatic surge in the volume of requests arriving at a server often results in the server being overwhelmed and response times shooting up. Current information systems do not sustain such situation. For example, on the web the ubiquitous access of browsers and rapid spread of news about an event, lead to a flash crowd when a huge number of users simultaneously access a popular web site. Flash crowds are typically triggered by events of great interest; either planned ones such as sport events (e.g. FIFA 1998 world cup event [6]) or unplanned ones such as an earthquake, etc. However the trigger need not necessarily be an event of widespread global interest. Depending on the capacity of a server, even a humble flash crowd can overwhelm the server. Obviously, current Internet information systems have failed to fulfill the stringent Internet users' requirements in such situations [7]. Moreover, the complexity and dynamism of those systems promote an imperative need for high-assurance in those systems. In addition, we believe that increasing system complexity shall reach a level beyond human ability to manage. These information systems are characterized by a decentralized control, large scale and extreme dynamism of their operating environment. They can be seen as instances of the *Complex Adaptive Systems* alike *social communities* [8]. Cooperation is the key of the evolution and continuity of the social communities. The potential benefits of cooperation among people should drive progress evolution in culture, technologies and business [9].

Inspired from both the spirits of cooperation in the social communities and the Autonomous Decentralized System (ADS) concept [10] [11], the concept of an *Autonomous Community Information System* (ACIS) is proposed to meet the rapidly changing users' requirements. It customizes the service for the

Manuscript received June 24, 2003.
Manuscript revised September 12, 2003.

[*] The authors are with the Department of Computer Science, Tokyo Institute of Technology, Tokyo, 152-8552, Japan.

specific end-users having interests in that service, in somewhere/anywhere, at specified time. ACIS allows individual end-users (community members) having the same preferences and requirements in somewhere/anywhere, at specified time, to communicate directly with one another and share information without relying on any specified servers. Community members mutually cooperate to assure the high quality and well-customized information service provision and utilization as well for all members. ACIS is completely decentralized in the sense that each member of the community performs the same set of tasks. It is dedicated for large number of users (e.g. target number is about 100,000). Moreover, it is highly required to achieve fairness of the load among the community members.

The contribution of this paper is the proposition of the ACIS concept, architecture and communication technology for large-scale information systems. We propose an *autonomous decentralized community communication technology* for achieving timeliness. The remainder of this paper is organized as follow. Section 2 clarifies the autonomous community information system concept and exhibits the system architecture. Section 3 exposes our proposed communication technology. Section 4 presents evaluation and simulation results showing improvement. We review related work on application level multicast protocols in section 5. The last section draws conclusions.

## 2. Autonomous Community Information System: Concept and Architecture

The main concern of the information systems has been in the past to efficiently retrieve relevant data for a particular request from immense repositories [12]. The research in information systems has turned to identify the location of the services and efficiently make the demands meet the offers [13]. In such distributed systems, two actors are coexisted: *Service Providers* and *End-users*. Service Providers offer the information content in the system. End-users consume the information services. The information systems based on the centralized model do not sustain the flash crowd problem. Accordingly, they have failed to satisfy the Internet users' requirements of timeliness. Currently, 90% of Internet resources are invisible and untapped [14]. *Peer-Peer information sharing systems* have turned to take into account the data and processing power that resides at the end-users. They drag information out of the centralized service providers onto end-users PC's. These systems are characterized by *unilateral benefits* because peers coordinate together for the satisfaction of only one of them, which requests the information. One request is required for the satisfaction of one peer. Peers share efforts for identifying the location of the required information. Then, information downloads are done directly between two peers [15]. These systems have two lacks. First, the number of the identical requests is increased by the growth of the number of peers who send the same request. As a result, a constant increase in traffic per peer is too high. Second, these peer-peer systems do not specify how many connections a peer may initiate, accept, or simultaneously maintain. Consequently some peers may have high load than others. Unfairness among users pushes them to give up from such systems. Obviously, these systems have failed to satisfy the Internet users' requirements (e.g. timeliness) too.

The main importance in the large-scale and very dynamic information systems is to meet the rapidly changing user's demands. We have identified that the constructive cooperation among end-users assure the well-customized information service's provision and utilization. Inspired from both Autonomous Decentralized System (ADS) concept [10] [11], and the spirit of cooperation in the social communities, we have proposed the concept of *Autonomous Community Information System* (ACIS), [16].

### 2.1 Concept

The basis of the ACIS concept is to provide the information to specific users in specific place at specific time. On the contrary, current information systems provide the information to anyone, anywhere and anytime. Thus, we have defined *Autonomous Community* as a place of a coherent group of autonomous members having individual objectives, common interests and demands at specified time and somewhere/anywhere. The community members are autonomous, cooperative and active actors and they mutually cooperate to enhance the objectives for all of them timely. In ACIS, each community member acts both as an information sender and a receiver. Furthermore, each message from a participant is meaningful to all the other community members and at the same time every member is typically interested in data from all other senders in the community. Contrary to the peer-peer systems, the communication among the community members is conducted on multilateral basis, as will be shown later in section 3. Community members cooperate not only for the satisfaction of one of them but also for all of them.

ACIS is a promising concept for information services operating at the edge of the network. It realizes the large-scale information system that successfully able to carry out, and enhance community members' objectives (e.g. timely information sharing) in a very dynamic environment. It guarantees the constructive cooperation and fairness among the community members with a very high degree of autonomy among them. We have developed a system architecture, called *Autonomous Decentralized Community System* (ADCS), that fosters the concept of the autonomous community information system.

### 2.2 Architecture

Community nodes will be connected on a bilateral basis. The bilateral logical contact between two community nodes will occur considering that the users of these nodes have the same interests and demands, at specific time and location. It is likely that in bilateral contacts, community nodes connect each other and share information. The autonomous decentralized community network is a self-organized logical topology. It is a set of nodes V that consider the symmetric connectivity and the existence of loops. Each node keeps track of its immediate neighbors in a table. The immediate neighbors' set of the community node x is defined as a set of nodes

$$INS_x = \{y;\ x, y \in V,\ h(x, y) = 1\} \qquad (1)$$

Where h(x, y) is the number of the logical hops between nodes x and y. For example, Fig. 1 shows that the immediate neighbors of node A are $INS_A = \{B, C, D, E\}$. Each node knows its neighbor's nodes and shares this knowledge with other nodes for forming a loosely connected large number of nodes. In Fig. 1 the solid (bold, gray) lines represent the logical bilateral-links among the community nodes. Each node judges autonomously to join/leave the community network by creating/destroying its logical links with its neighbor's members based on its user's preferences. The community boundary changes with the dynamic change of its member's requirements.

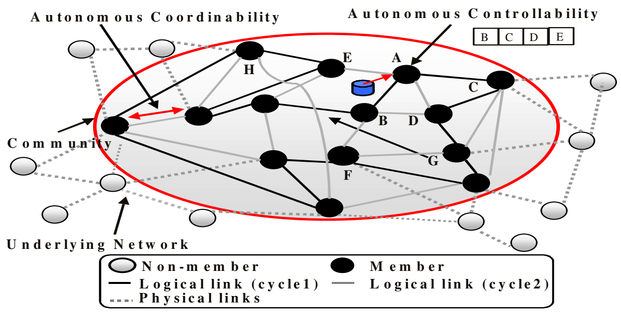

**Fig. 1** Autonomous decentralized community architecture.

### 2.2.1 Community Network Construction/ Maintenance

Any node can join and leave the community network at any time and through a node already exist in the community network. If no scheme is imposed on the way nodes join and leave, then the network is likely to grow to become exponential network. This uncontrolled evolution may lead to some hotspots in the community network. For example, peer-peer systems do not specify how many connections a peer may initiate, accept, or simultaneously maintain. Consequently some peers may have high load than others. In that respect, we have proposed an autonomous decentralized community construction technique for making the potential hotspots very unlikely [16]. Community network construction polices the nodes joining and leaving the community network and organizes them in a *2d-regular* graph G = (V, E), such that V is the set of nodes with labels [M] ={1, 2, …, M} and E is the set of edges. The 2d-regular graph is the graph that each node has 2xd neighbors (node connectivity) and composed of independent *d Hamilton* cycles. For example, Fig. 1 shows the community network as 4-regular graph composed of two cycles where the connectivity of each node is 4. Those neighbors are labeled as $\mathbf{r_p^{(1)}}$, $\mathbf{r_s^{(1)}}$, $\mathbf{r_p^{(2)}}$, $\mathbf{r_s^{(2)}}$ , …, $\mathbf{r_p^{(d)}}$, $\mathbf{r_s^{(d)}}$. For each i, $\mathbf{r_p^{(i)}}$ denotes the neighbor node's predecessor and $\mathbf{r_s^{(i)}}$ denotes the neighbor node's successor on the i-th Hamilton cycle. The advantage of using the Hamilton cycles is that nodes joining or leaving will only require local changes in the community network as will be described as follows.

**1. Join Process**

When a node wishes to join a community, ADCS assumes that the node is able to get at least one community node X by an out-of-band bootstrap mechanism similar to Narada [22] and CAN [28]. In this paper we do not address the issue of the bootstrap mechanism. The joining node (A) sends a join request to one community node X. Node X is responsible to find d neighbors for node A to connect. If the majority of the joining nodes connect to the neighbors of the same node X, then the community network diameter increases linearly and produces a network similar to the completely ordered regular graph [32]. In order to avoid such situation, each node determines autonomously φ(t), the node joining rate. φ(t) is defined as the number of the new connected nodes to itself within period t. Node X broadcasts join-request to all the community nodes within $O(\log_{2xd} M)$ layers. Each node judges autonomously to reply "Ok to join" or not based on its joining rate φ(t). Node X receives some "OK to join" messages and autonomously selects d nodes from them. Then, each node from these d nodes calls the following *Add(A, i)* routine to add joining node A in each i-th Hamilton cycle:

```
Add (A, i){
        Successor_node := Calling_node → $r_s^{(i)}$ ;
        Edge (Calling_node, A, i);
        Edge (A, Successor_node , i); }
Edge (B, C, i){
          ( B→ $r_s^{(i)}$ ) := C;
          ( C→ $r_p^{(i)}$ ) := B; }
```

The expression (H→Var) means that we seek the value of Var from node H. In addition, the expression (H→Var:=Y) means that we set the variable Var of node H to value Y. The add routine inserts the joining node between the calling node and the successor of the calling node in the i-th Hamilton cycle. It substitutes the edge between calling node and its successor by two edges, one between calling node and joining node and the other one between joining node and successor of the calling node. The Edge(B,C,i) routine makes C the successor of B and B the predecessor of C. Thus, it creates the communication session between nodes B and C. Obviously; the join process only requires local changes in the community network.

**2. Leave/failure Process**

When a member leaves the community, it notifies its neighbors and calls the following leave routine to leave from each Hamilton cycle.

```
Leave () {
    For i :=1, …, d in parallel do
        Edge (Leave_node→ $r_p^{(i)}$, Leave_node→ $r_s^{(i)}$, i); }
```

The leave routine creates edges between the successor and predecessor node of the leaving node at d cycles. Obviously, the leave process only requires local changes in the community network with O(d) messages [16].

It is also required to consider the difficult case of node failure. In such a case, failure should be detected locally as follows. The neighboring nodes in the $INS_x$ periodically exchange keep-alive message with the node x. If node x is unresponsive for a period T, it is presumed failed. All neighbors of the failed node update their INS sets and each couple of them connects one another to keep the same number of links for all nodes. This technique scales well: fault detection is done by exchanging messages among small number of nodes, and recovery from faults is local; only a small number of nodes $|INS_x|$ is involved. In addition, it is possible that some nodes failure can cause the community network to become partitioned. In such case, nodes must first detect the existence of a partition and then repair it by adding another links to reconnect the community network.

### 2.2.2 Node Autonomy

The ADS concept has defined the node autonomy based on two characteristics [10]. First, *autonomous controllability* is that, if any node leaves, fails and joins the community system, the other community members still can continue to manage themselves and to perform their responsible functions. Second, *autonomous coordinability* is that, if any node leaves, fails and joins the community, the other community members still can coordinate their individual objectives among themselves. Consequently, each member is able to operate in a coordinated fashion. The evaluation of these characteristics has been shown in [31][33][34].

Each node recognizes autonomously a member from a non-member and cooperatively forwards the community information to only its neighbor's members. Community node does not forward the community information/request out of the community. Moreover, each node "*think globally and act locally*" by taking a decision autonomously based on its local information to store the relevant received information. The decision is taken not only according to the node situation (e.g. limited resources) and the importance of the offered information but also according to the other members' requirements. Each community node keeps a short memory of the recently routed messages in order to avoid the congestion in the community network. Each node autonomously coordinates (cooperates) with the others for locating, and/or providing the information in the community.

ADCS architecture has no central server whatsoever, as can be seen in Fig 1. It is a fully decentralized model and does not rely on any central authority to organize the network and service provision. The conditions to guarantee the autonomous and fairness characteristics in large-scale systems are: First, each node does not know the total community system but knows only 2xd neighbors (local information) and can communicate only with them. Second, all nodes have same responsibilities in membership management as shown in 2.2.1 and service provision in section 3. Thus, ADCS does not load up any single node excessively and enables the development of the large-scale information systems. For a timely communication among the community members this paper proposes the technology that will be described in section 3.

## 3. Autonomous Decentralized Community Communication Technology

The conventional communication, typically through Web browsers, has been built on the one-to-one communication protocol. In one-to-one, data travels between two users, e.g., e-mail, e-talk. This protocol gobbles up the network bandwidth and makes the real time services unresponsive. Caching most popular web pages on the proxy server reduces the network bandwidth consumption and the access latency for the users. However, the web caches techniques have some disadvantages as follows. First, a single proxy server is a single point of failure. Second, the limited number of users per proxy manifests bottleneck affects. Third, data does not updated automatically. Finally, cache misses increase in the latency (i.e. extra proxy processing). In the conventional one-to-many group's communication the message travels primarily from a server to multiple users, e.g., web download and software distribution. For very large groups (thousands of members) or very dynamic multicast groups (frequent joins and leaves), having a single group controller might not scale well. Currently, there is no design for the application-level multicast protocol that scales to thousands of members. For example, *Overcast* [17] builds the mesh per group containing all the group members, and then constructs a spanning tree for each source to multicast information. The mesh creation algorithm assumes that all group members know one another and therefore, does not scale to large groups. *Bayeux* [18] builds a multicast tree per group. Each request to join a group is routed to a node acting as the root. This root keeps a list of all the group members. All group management traffic must go through that root. It generates more traffic for handling a very dynamic group membership. *Bayeux* ameliorates these problems by splitting the root into several replicas and partitioning members across them. But this only improves scalability by a small factor. Section 5 will show some application level multicast systems.

### 3.1 Community Content-code and Multilateral Communication Technique

Conventional communication technologies use the destination address (e.g. unicast address, multicast address) to send the data. In very changing environment likes ADCS (i.e. end-users are frequently joined and left), these conventional communication technologies are not applicable. Thus, the autonomous decentralized community communication technology has broached [19] to assure a productive cooperation and a flexible and timely communication among members. The main ideas behind our proposed communication technology are: content-code communication (community service-based) for flexibility and multilateral benefits communication for timely and productive cooperation among members. The first main idea behind the autonomous decentralized community communication technology is the separation of the logical community service's identifier from the physical node address. In this communication technology, the sender does not specify the destination address but only sends the content/request with its interest *content Code* (CC) to its neighbor's nodes. CC is assigned on a type of the community service basis and enables a service to act as a logical node appropriate for the community service. Fig. 2 shows the community communication message format. CC is uniquely defined with respect to the common interest of the community members (e.g. politic, news, etc.). The information content is further specified by its *Characterized Code* (CH). The CH is the hash of the message content. It is uniquely specified with respect to the message content (e.g. data or request). It can be computed by the *collision resistance hash* function (e.g. SHA-1 [20]) that ensures a uniform distribution of CH.

| | | |
|---|---|---|
| **CC** | **CH$_i$** | **Data/Request** |

**Fig. 2** Community communication message format.

The second main idea behind the autonomous decentralized community communication technology is *multilateral* communication for timely and productive cooperation. The multilateral communication likely occurs among the community members that are already networked on a bilateral basis. All members communicate productively for the satisfaction of all the community members, as follow.

The proposed autonomous decentralized community communication technology performs the communication among the community members that has called "*1→N*" [26], [27]. A brief scenario of the 1→N community communication is described as follows. The community node asynchronously sends a message to each one from N neighbor's nodes. Then, those N nodes forward the same message to another N nodes in the next layer and so on, until all the community nodes received the message. The autonomy of the 1→N communication can be seen as follow. Each community node recognizes autonomously a member from non-member and judges autonomously to forward community messages to only N community neighbor's nodes. In order to avoid the congestion that may be happening if some of the community nodes simultaneously send identical messages, each node keeps a short memory (CH) of the recently routed messages and judges autonomously to forward only one copy of the received messages to the other neighbor's nodes. This paper evaluates the community network traffic in such case that will be shown in section 4.2. Moreover, each node autonomously takes a decision to keep or delete the short memory of the received message based on the frequency of receiving such message.

The 1→N communication technology does not rely on any central controller. Each community node has its own local information and communicates only with specified number (N) of the neighbor's nodes. There is no global information such as IP multicast group address [29] or multicast service nodes [24], [25].

## 3.2 Community Communication Protocols

The autonomous decentralized community communication technology has two communication protocols: *publish based* and *request /reply-all based.*

- *Publish based protocol.* When one of the community members has new information, she/he publishes it to all the community members using "1→N". A typical application is news information sharing among users having the same interests and demanding to know specific news at specific time and or location. This paper addresses only non-multimedia contents (news) having moderate size. The publish-based protocol offers an effective solution to the flash crowd problem as shown in Fig. 3. The solution scenario is as follows. As soon as one of the community members S has downloaded an interesting content for the community from the server (e.g. news server), she/he publishes it to all the community members, thereby relieving the server of this task and alleviating a load on the server. Thus, the load is distributed among the community nodes. When the number of nodes increased sharply, the load at each node is increased slightly. In addition, it represents a scalable solution for large-scale information dissemination systems.

- *Request/reply-all based protocol.* When a community member wants to locate information, she/he emits a request message. Then the others community members cooperate to locate the requested information. When any community node receives the requested message, it processes the request. If no results are found at that node, the node will forward the request to its neighbor's nodes by using "1→N". Otherwise, the node will produce results, such as pointers to the information or the whole content based on the size of the information. Then that node will send a reply message not only to the node, which requested the information but also to all the community members. Fig. 4 shows the message flow when the community node S sends a request (solid arrows) to its neighbor's and node R replies (dotted arrows) to all the community members by the required information I. The *reply to all* protocol affords the other community members to send the same request. Consequently, all the community members enrich their experiences and/or get to know new services without requesting, in which individually they cannot get to know. Thus, the multilateral benefits characteristic of the community can be satisfied. In addition, it decreases the traffic per node by avoiding multiple requests for the same content.

The originality of our proposed communication technology does not come only from the *content-based* communication but also from the *reply-all* that satisfies the *multilateral benefits.* In 1→N community communication all members cooperate for the satisfaction of all the community members contrary to the peer-peer (P2P) communication techniques. By 1→N, one request is required for the satisfaction of all members in the community.

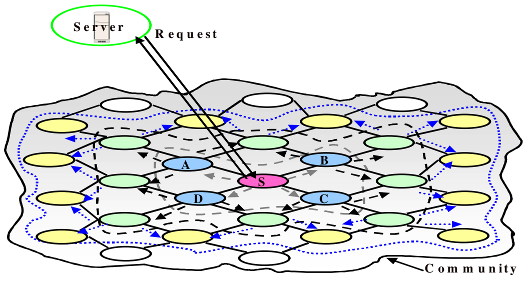

**Fig. 3** Publish based protocol.

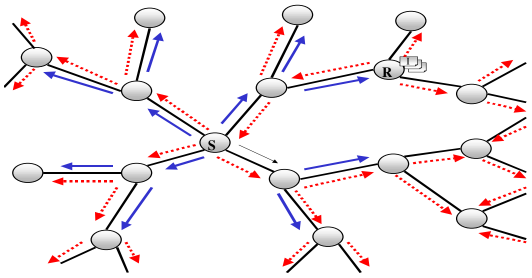

**Fig. 4** Messages flow in request/replay-all based protocol.

In P2P, peers cooperate for the satisfaction of only one, which request the information (unilateral benefits). The comparisons between the community information system and the conventional information systems: client/server and peer-peer are tabulated in table 1. From this table we conclude that the community communication is: service-based, cooperative, relationship and multilateral benefits communication [26], [27]. Moreover, the system is scalable of the response time with a huge number of members. Thus, it guarantees a timely communication among the community members.

Table 1: Comparison

| | | | Conventional (client/server) | Peer-Peer | Community |
|---|---|---|---|---|---|
| Technological | Membership-Management | | Centralized (Groupware) | Centralized (e.g., *Overcast, Bayeux*) | Decentralized Loosely control |
| | Communication | Model | Address-based | Address-based | Service-based |
| | | Request | One-one | One-many | Cooperative(1→N) |
| | | Reply | One-one | One-one | Cooperative(1→N) |
| Characteristics | Benefits | | Unilateral | Unilateral | Multilateral |
| | Users | | Passive | Active | Active |
| | Load | | Servers-congestions | Peers-congestions | No-Congestion (Fairness) |

## 4. Evaluation

To evaluate the performance of our proposed technology, we consider the community network topology as regular graph [16]. Assume the number of the community nodes is M and each node has k neighbors. The information is broadcasted in a tree as follow. The source node sends asynchronously a message to each one from k neighbors (children) and then each neighbor forwards asynchronously the same message to another k-1 neighbors nodes in the next layer and so on, until all the community nodes received the message. Thus, the number of the community nodes take part in the broadcasting tree can be written as follows.

$$M \le 1 + \sum_{i=0}^{L-1} k(k-1)^i \qquad (2)$$

Where L is the number of layers or depth of the tree. Then the number of layers can be calculated as follows.

$$L \approx \left\lceil \frac{\log\left[\frac{(M-1)*(K-2)}{K}+1\right]}{\log(K-1)} \right\rceil \le \frac{\log(M)}{\log(k-1)} \qquad (3)$$

Under the assumption that the communication cost between each node is one unit of time then, the transmission time τ to send a message from one member to all the other members is bounded by O(Nlog$_N$(M)), where N=k-1. Consequently we can drive the optimal 1→N communication as follows.

$$\frac{d\tau}{dN} = \frac{d}{dN}\left(N * \log_N(M)\right)$$
$$= \log(M)\left(\frac{\log(N) - 1/\ln(10)}{(\log(N))^2}\right) \qquad (4)$$

From equation 4, we conclude that $d\tau / dN = 0$, $d^2\tau / dN > 0 \Rightarrow N \approx 3$. τ is concave up. For any number of nodes M, the (1→3) community communication technology is the optimal. Similarly, Fig. 5 shows that ever-increasing the number of nodes the optimal communication is 1→3 under the assumption that the communication cost between each node is one unit of time.

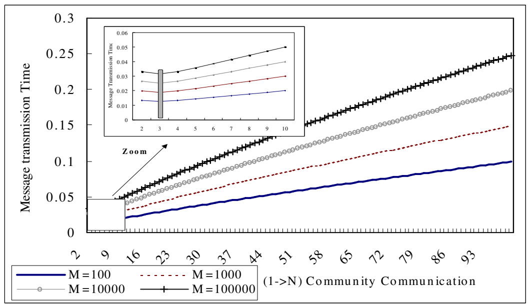

**Fig. 5** Optimal 1→N community communication

### 4.1 Simulations and results

We have developed two simulations to prove the validity of the proposed ADCS system. The target number of users of the ADCS is 100,000. Thus, both simulations ran over the community network contains 100,000 members. The first simulation is based on the assumption that the communication cost between each node is same and equal one unit of time. In order to show the validity of the proposed ADCS in reality we have developed the second simulation over a random generated network with different communication cost between nodes.

#### 4.1.1 Simulation-I

We simulated the one-to-one communication based on the client/server model. The experiments have been conducted over

100,000 of requests. Each client accesses the server and sends a request simultaneously to the server (e.g. news server). Obviously, the surge of simultaneous requests arriving at the server results in the server overwhelmed and response time shooting up. Caching web pages on the proxy servers reduces the access latency for the clients. Thus, the webs caching techniques having slightly effect in the response time. Fig. 6 shows the one-to-one communication simulation model based on caching proxies as follows. We assume that each caching proxy is located at an organization and clients' requests are assigned randomly to S caching proxies. It has been proved that a chasing proxy has an upper bound of 30-50% in its hit rate [21]. In addition, we simulated the proposed community communication technology on a network spending 4-array connectivity for each community node. The experiments have been conducted over 100,000 community members, using 1→3 communication technology and is constituted of communication cost between each node $\tau_{cc}$= 10 ms. $\tau_m$ =10 ms, the average time that each node needs to monitor the recent received messages to avoid the congestion. Thus the transmission time $\tau$ to send a message asynchronously from any node to all the other community nodes is bounded by $L*N*(\tau_{cc}+\tau_m)$.

We concentrate in these experiments on the comparison between the conventional one-to-one communication techniques without and with caching proxy (hit rate of 30%, 50%) and (1→N) community communication technology. ADCS gathers those clients. As soon as one client (member) has downloaded an interesting content for the community from the server, she/he publishes it to all the community members using "1→N". Fig. 7 shows the variations of the number of the members in the community with the worst transmission time of a message to all members. Fig. 7 depicts the effectiveness of the proposed communication technology compared with the conventional ones. The 1→N communication technology is able to send a message to all the community members (target number 100,000) with imprecision close to 93% compared with (one-one) unicast, 91% compared with unicast with hit rate 30% and 87% compared with unicast with hit rate 50%. These results reflect that unicast with caching proxies technologies improve the communication performance slightly compared with the proposed communication technology. Thus, it shows that the community communication technology is scalable of the response time with the number of the members. As shown in the zoom part in Fig. 7, for small number of members (less than 1000) the proposed community communication technology

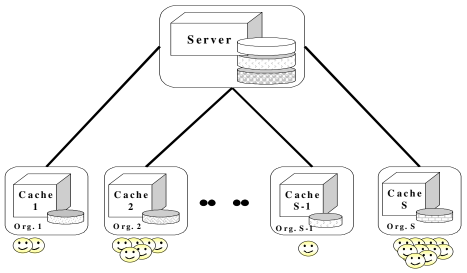

**Fig. 6** One-to-one communication simulation model based on caching proxies

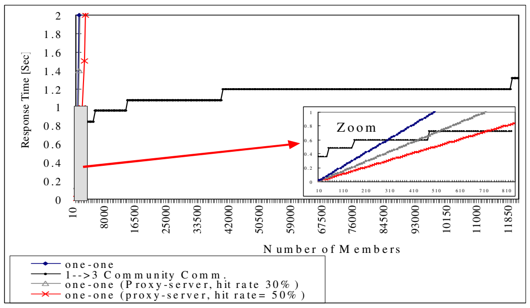

**Fig. 7** Simulation-I result: scalable announcement.

is not effective but it reveals meaningful results when the total number of members in the community increases sharply. The results of this simulation suggest that ADCS can achieve good performance for large number of members (target number 100,000) under the assumption that the communication cost between each node is one unit of time. The question then is: can ADCS support large number of members (target number 100,000) with different communication cost. Next section, the simulation II will answer this question.

### 4.1.2 Simulation-II

#### 4.1.2.1 Simulation Setup

We ran a simulation on a network topology with 100 routers linked by core links. The Georigia Tech [30] random graph generator using the transit-stub model has generated it. Random link delay of 4-12ms was assigned to each core link. The average delay of core links (computed by the graph generator) is approximately 13ms. The community end-nodes were randomly assigned to routers in the core with uniform probability. Each community end-node was directly attached by a LAN link to its assigned router. The delay of each LAN link was set to be 1ms. End-nodes join the community network with joining rate 100 nodes/Sec with equal distribution. Members leave the community network with leaving rate 10 nodes/Sec with random distribution. Thus, the community network size grows to the target number of member within about 20 minutes (running time of our simulation). The community network spends 4-array connectivity for each end-node.

#### 4.1.2.2 Communication Results

We have conducted a simulation to compare ADCS with both unicast and Peer-Peer communication technologies. We have constructed Peer-Peer overlay network as a completed order regular graph [32]. The construction scenario of this network is as follows. The majority of joining nodes connect to the neighbors of a specific node as a result the peer-peer overlay network diameter is increased linearly with the increasing of the network size. In each run of the simulation, one member from the community is picked as source at random and then the required communication cost to send a message to all nodes is evaluated. For simplicity, Fig. 8 shows only the simulation results of the first 3.5 Seconds from the simulation running time. It plots the variations of the mean communication cost required to send a message from a node to all nodes participated at each instance of time during the experiment. ADCS has shown about 70% improvement of the mean communication cost compared with peer-peer technology and 90% compared with unicast. We argue that ADCS shows 70% imprecision compared with the peer-peer technology to the proposed construction technology that yields a short community network diameter. For more efficient communication, the further research of ADCS is to consider latency between members as an important criterion that needs to be optimized.

#### 4.1.2.3 Construction and Maintenance Overhead Results

We conducted a simulation to evaluate the community network construction and maintenance overheads. The joining communication cost is the required communication time to forward the join request message within the community network. We ran this simulation for 20 minutes as a result the community network size becomes 108,000 members and about 384ms is the required communication cost for the construction and maintenance of the community network. In the simulation each second 10 nodes are chosen randomly to leave the community network. Each leave node calls the leave() routine with maintenance cost that varies with the communication cost to its successors and predecessor neighbors. To get better understanding of the simulation results, Fig. 9 presents part of the simulation results of 5.5 seconds and 500 members. In this figure, the peaks represent the leaving cost. The results show that the required communication cost for the construction (joining) and maintenance of the community network is increased logarithmically with standard deviation approximately equal 0.0033. Thus, these results indicate that the community network construction and maintenance techniques are scalable for large number of members.

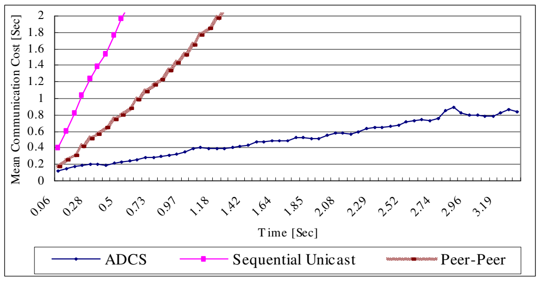

**Fig. 8** Mean communication cost comparison.

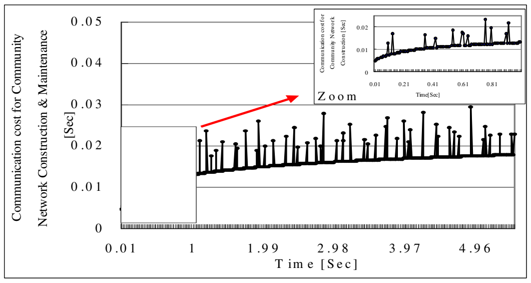

**Fig.9** Community Network construction and maintenance overhead

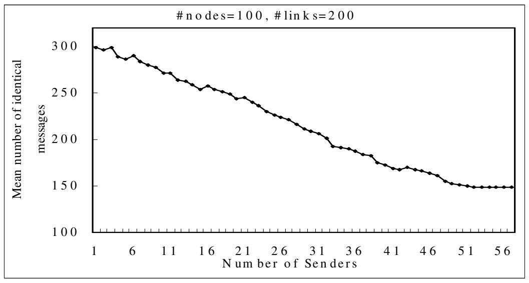

**Fig. 10** Network traffic with multiple senders

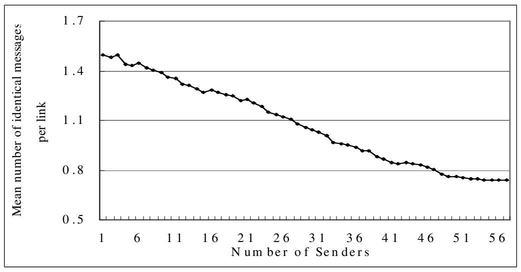

**Fig. 11** Network traffic with multiple senders

## 4.2 Community Network Traffic

We ran an additional experiment to evaluate the network traffic when many nodes in the community send an identical message at once. Each node monitors the received messages and forwards only one from the received identical messages. This experiment ran on a community network with 100 nodes and 200 logical links, which were generated as regular graph [16]. The communication cost of each link was set to 10 ms. We ran the simulation 1000 times to determine the mean number of identical messages (MNIM) in the community network. Each time we ran the experiment, the senders were selected randomly. Fig.10 and Fig. 11 plot the variation of both the MNIM in the community network and the MNIM carried by a logical link with the number of senders. The increase of the number of senders reduces both MNIM in the community network and MNIM per link. The standard deviation of MNIM per link is 0.253. In this experiment more than 50% of community nodes sends an identical message at once as a result the community network traffic decreases and the MINIM per logical link converges to about 0.6. Thus, the community network traffic is reduced regardless to the increasing of both the number of members up to the target number and the number of sender of an identical message at once. This result indicates that community system does an efficient job in distributing loads over all nodes; each node is responsible for forwarding messages only to a small number of nodes. This is important to achieve scalability with the increasing of the community system size up to the target number.

## 5. Related Work

ADCS, like Overcast [17], Narada [22] and ALMI [23], implement multicast, uses a self-organizing overlay network and assume only unicast support from the underlying network layer. Narada and ALMI are dedicated for collaborative applications with a small number of group members. However, ADCS is a framework for collaborative applications with a large number of group members. ALMI takes a centralized approach to the tree creation problem. Clearly, it constitutes a single point of failure for all control operations related to the group. In contrast, ADCS takes a decentralized approach (i.e. no node knows the total system [16]). Scattercast [24] and OMNI [25] are designed for global content distribution. They argue for infrastructure support, where proxies are deployed in the Internet to support large number of clients. For large-scale data distributions, such as live webcasts, a single source exists. In contrast in the ADCS, the nodes are considered to be equal peers and are organized in the community network. The community concept is a "*real*" end-system multicast approach. The end-systems (autonomous members) work cooperatively to deliver the data on the whole community members. ADCS is dedicated for multi-sender applications with large number of participants. It does not depend on the multicast support by the routers (e.g. IP multicast) and does not depend on the multicast service nodes MSNs (e.g. Scattercast and OMNI). A rapid and dramatic surge in the volume of requests arriving at MSN often leads to a flash crowd. Clearly, MSN constitutes a single point of failure for information provisions to the group. Scattercast and Narada take a mesh-based approach to the communication tree creation problem. In this approach, every member should keep a full list of all other members. Therefore, this approach does not scale well to the large group sizes. In the other side, the ADCS scales well to the large number of members because each member is required to know a small number of other members (neighbors). The proposed community information system (ADCS) is a framework for both information sharing and large-scale data distribution applications. A comparison of different application level multicast systems with the community system is tabulated in table 2.

Table 2: Application level multicast systems

| | Control approach | Overlay structure | Group size | Senders |
|---|---|---|---|---|
| ALMI | Centralized | Peers | Small | Multi |
| NARADA | Distributed | Peers | Small | Multi |
| Scattercast | Distributed | MSNs | Small | Single |
| OMNI | Distributed | MSNs | Small | Single |
| ADCS | Decentralized Loosely control | Autonomous Members | Large | Multi |

## 6. Conclusion

Information service systems have radically altered the world of social and business and offer enormous potentials for e-social and e-commerce. The necessity of high-assurance and large-scale in information systems has been fortified by the introduction of critical applications. The main importance in those systems is to meet the rapidly changing user's demands. However, the current information technologies do not sustain the rapid and dramatic surge in the volume of requests arriving at a server. Thus, it is necessary to design a high assurance and large-scale information system that meets the rapidly changing users' requirements for services that can cope the extreme dynamism of the operating environment.

In this paper, we clarify the concept, architecture and communication technology of the community information system. Inspired from the constructive cooperation in the social community and the ADS concept, the autonomous community information system concept (ACIS) has been proposed. In that respect, community members are active actors and they mutually cooperate to assure the quality of the information service provision and utilization among them, since individually they cannot. In addition, the *Autonomous Decentralized Community System* (ADCS) has been developed in order to sustain the proposed concept. Finally, the autonomous decentralized community communication technology has been proposed to achieve a productive cooperation and a flexible and timely communication among the community members. This communication technology is not only content-code communication (service-based) but also multilateral communication in which all members cooperate for the satisfaction of all the community members, contrary to the other communication technologies (e.g. P2P communication). The simulation results has depicted that the community communication technology is scalable of the response time with the number of the members. Thus, timeliness, an essential component in large-scale information system is achieved.

We are currently extending this work in several directions. First, ADCS considers latency between members as an important criterion that need to be optimized. For that reason we will organize the community as a number of sub-communities. Each sub-community has a leader where, the latency from any member to the leader is bounded by specific value. Second, we are studying how to construct and maintain sub-communities. Finally, we are studying how to enhance the communication cost with acceptable construction and maintenance overheads.

## Reference

[1] L. G. Roberts, "Beyond Moore's law: Internet growth trend ", IEEE Computer, Vol. 33, no. 1, p. 117, 2000.

[2] Peter Lyman and Hal R. Varian, "How Much Information", J. Electronic Publishing, Vol. 6, no 2, pp. 30-35, Dec. 2000.

[3] David B, S. Shrivastava and F. Panzieri, "Constructing Dependable Web Services," IEEE Internet Computing, Vol. 4, No. 1,pp. 25-33, 2000.

[4] K. Mori, "Applications in Rapidly Changing Environments," IEEE Computers, Vol. 31, No. 4, PP. 42-44, 1998.

[5] K. Mori, "Towards integrated methods for high assurance systems," IEEE Computer, vol. 31, No. 4, pp. 32-34, 1998

[6] Martin Arlitt, Tai Jin, "Workload Characterization of the 1998 World Cup Web Site," Hewlett Packard Co. 1999.

[7] J. Jung, . B. Kirshanmurthy, and M. Rabinovich,, "Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites", The 11th Int. WWW2002, Hawaii, USA, May 2002.

[8] C. G. Langton, "Complex Adaptive Systems," MIT Press, 1995.

[9] John Stewart, "Evolution's Arrow: The direction of evolution and the future of the humanity", Chapman press, Australia, 2000.

[10] K. Mori, "Autonomous Decentralized Systems: Concept, Data Field Architecture and Future Trends," Proc. ISADS'93, IEEE, Japan, 1993.

[11] K. Mori, H. Ihara, Y. Suzuki, K. Kawano, M. Koizumi, M. Orimo, K. Nakai, and H. Nakanishi, “Autonomous Decentralized Software Structure and its Application,” Proc. IEEE FJCC'86, Nov. 1986.

[12] V.N. Gudivada, V. V. Raghavan, W. I. Grosky, and R. Kasanagottu., “Information retrieval in the World Wide Web”, IEEE Internet Computing Magazine, Vol. 1, no. 5, pp. 58-68, Sept. 1997.

[13] K. Mori, “Assurance System Architecture for Information Service by Utilizing Autonomous Mobile Agents”, 5th HASE Int. Sym., Nov. 2000.

[14] R. Dornfest, “Dark Matter, Sheep and the Cluster: Resolving Metaphor Collision in P2P,” The O'Reilly Peer-to-Peer and Web Services Conference Washington, D.C., November 5-8, 2001

[15] M. Ripeanu, “Peer-Peer Architecture Case Study: Gnutella Network,” In Proc. of Int. Conf. on P2P computing, 2001.

[16] K. Ragab, T. Ono, N. Kaji, K. Mori, “ Autonomous Decentralized Community Concept and Architecture for a Complex Adaptive Information System,” Proc. IEEE FTDCS, Puerto Rico, May 2003.

[17] J. Jannotti, D. K. Gifford, K. L. Johnson and M. F. Kaashoek, “Overcast: Reliable Multicasting with an Overlay Network,” In Proc. 4th Sym. OSDI, pages 197-212, Oct. 2000.

[18] S. Q. Zhuang, B. Y. Zhao, A. D. Hoseph, R. H. Katz and J. D. Kubiatowicz, “Bayeux: An Architecture for Scalable and Fault tolerant Wide-area Data Dissemination,” 11th Int. NOSSDAV, June 2001.

[19] K. Ragab, T. Ono, N. Kaji, K. Mori, “Community Communication Technology for Achieving Timeliness in Autonomous Decentralized Community Systems,” Proc. IEEE IWADS, China, pp 56-60, Nov., 2002.

[20] FIPS 180-1 Secure hash standard. Technical Report Pub. 180-1, Federal Information Processing Standard (FIPS), National Inst. Of Standards and Technology, US Dept of Commerce, Washington D.C., April 1995.

[21] Stephen Williams, and E. A. Fox, “*Caching proxies: Limitations and potentials,*” In Proc. 4th Int. World Wide Web Conference, Dec. 1995.

[22] Y. H. Chu, S. G. Rao, and H. Zhang, “A case for end system multicast,” in Proc. Of ACM Sigmetrics, June 2000, pp. 1-12.

[23] D. Pendarakis, S. Shi, D. Verma and M. Waldvogel, “ALMI: an application level multicast infrastructure”, In Proc. of 3rd Usnex Symp. USITS, March 2001.

[24] Y. Chawathe, "Scattercast: An Architecture for Internet Broadcast Distribution as an Infrastructure Service," Ph.D. Thesis, University of California, Berkeley, Dec. 2000.

[25] S.Banerjee, C. Kommareddy, K. Kor, B. Bobb and S. Khuller, "Construction of an Efficient Overlay Multicast Infrastructure for Realtime Applications," Proc. of INFOCOM 2003.

[26] K. Ragab N. Kaji, K. Moriyama, K. Mori, “Scalable Multilateral Communication Technique for Large-Scale Information Systems," Proc. IEEE COMPSAC 2003, Nov., 2003, Dallas, USA.

[27] K. Ragab, N. Kaji, K. Moriyama and K. Mori, “Service-Oriented Autonomous Decentralized Community Communication Technique for a Complex Adaptive Information System," Proc. IEEE/WIC WI 2003, Oct. 2003, Canada.

[28] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker, “A Scalable Content-Addressable Network,” Proc. SIGCOMM’01. California, USA.

[29] S. Deering, and D. Cheriton, “Multicast routing in datagram internetworks and extended LANs,” ACM Trans. on Comp. Sys., 8(2):85-110, May 1990.

[30] E.Zegura, K. Calvet and B. Samrat “How to model an Internetwork,” INFOCOM96, 1996.

[31] K. Mori. “Proposal of the Autonomous Decentralized Concept,” IEEJ, C-vol 104, No 12, pp 302-310, Dec., 1984.

[32] Andy Oram, “Peer-to-Peer Harnessing the Power of Disruptive Technologies,” O’Reilly & associates, Inc, March 2001.

[33] K. Mori, K. Sano and H. Ihara, “Autonomous Controllability of Decentralized System aiming at Fault-tolerance,” Proc. 8th Triennial World Cong., Japan, 1981.

[34] M. Koizumi and K. Mori “Autonomous Coordinability of Decentralized System considering Subsystems Failures,” Proc. 7th Int. Conf. On Multiple Criteria Decision Making, Kyoto, Japan, 1986.