PAPER Special Section on Knowledge-Based Software Engineering

ACIS: A Large-Scale Autonomous Decentralized Community Communication Infrastructure

Khaled RAGAB[†a)], **Naohiro KAJI**[†], ***Nonmembers, and*** **Kinji MORI**[†], ***Member***

SUMMARY This paper presents ACIS, an Autonomous Community Information System. ACIS is a proposition made to meet the rapidly changing users' requirements and cope with the extreme dynamism in current information services. ACIS is a decentralized bilateral-hierarchy architecture formed by a community of individual end-users (*community members*) having the same interests and demands at specified time and location. It allows those members to mutually cooperate and share information without loading up any single node excessively. In this paper, *autonomous decentralized community construction and communication technologies* are proposed to assure a productive cooperation, a flexible and timely communication among large number of community members. The main ideas behind the proposed communication technology are: content-code communication (service-based) for flexibility and multilateral benefits 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 has been shown by simulation. Thus, the autonomous decentralized community communication technology reveals interesting results when the total number of members in the community increases dramatically.

key words: *autonomous community information system, multilateral community communication, information service systems*

1. Introduction

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. 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 [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 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

Manuscript received June 23, 2003.
Manuscript revised October 7, 2003.

[†]The authors are with the Department of Computer Science, Tokyo Institute of Technology, Tokyo, 152–8552 Japan.
a) E-mail: ragab@mori.cs.titech.ac.jp

members. ACIS is completely decentralized in the sense that each member of the community performs the same set of tasks. 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 construction/maintenance and communication technologies for large-scale information systems. The remainder of this paper is organized as follow. Section 2 clarifies the ACIS concept, exhibits the system architecture and illustrates the community network construction and maintenance technology. Section 3 exposes the proposed *autonomous decentralized community communication technology* for achieving timeliness. Section 4 presents evaluation and simulation results showing improvement. We review related work on application level multicast protocols in Sect. 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. These systems are characterized by *unilateral benefits* because peers coordinate together for the satisfaction of only one of them, which requests the information. Thus, the satisfaction rate for M peers in the systems is approximately 1/M and converges to zero as M increases. 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. Blending the spirit of cooperation in the social communities, and the Autonomous Decentralized System (ADS) concept [10], [11], 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 at specific place and 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 as both 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 shown later in Sect. 3. Community members cooperate, not only for the satisfaction of one of them, but also for all of them. Thus, the satisfaction rate of M members is approximately one.

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 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 that fosters the concept of the ACIS as follows.

2.2 Architecture

Community nodes will be connected on a bilateral hierarchy 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 bilateral-hierarchy, the symmetric connectivity and the existence of loops. The bilateral-hierarchy means that a node x can be parent of a node y in an information flow and child of the node y in another information flow. Each node keeps track of its immediate neighbors in a table. The immediate neighbors' set of the community node x is defined as the set of nodes

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

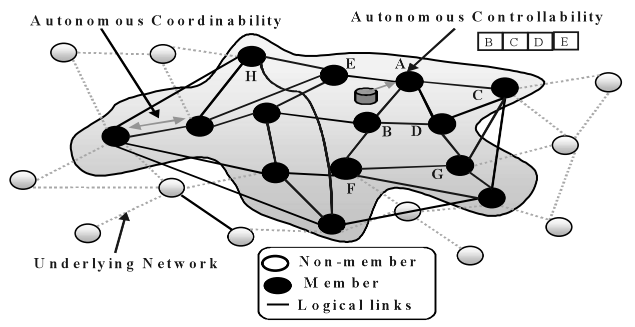


Fig. 1 ACIS architecture.

Where $h(x, y)$ is the number of 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 lines represent the logical bilateral-link among the community nodes. Each node judges autonomously whether 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's boundary change with the dynamic change of its members requirements.

2.3 Autonomous Decentralized Community Network Construction/Maintenance Technology

The autonomous decentralized community network is to be symmetric in the sense that each node in the network must have identical capabilities and duties on the network. This excludes the existence of central servers that might be involves in organizing the network. Our first goal is to construct a community network that can support broadcast efficiently. In addition, we should avoid hotspots in the community network by distributing the network traffic evenly among the community nodes during the broadcast. The second goal is to make our community network construction is highly scalable. Finally, the community network topology has to provide redundancy. Node failures must not lead to the community network disconnecting or severely hampering broadcast properties. Next subsection presents our proposed construction technique for the community topology.

2.3.1 Organizing Community in Regular Graph

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 $[M] = \{1, 2, \ldots, M\}$ and E is the set of edges. Due to frequent join and leave, E and V are changing with time. The graph G is composed of independent d edge disjoint *Hamilton* cycles. Each node has 2d neighbors (node connectivity). Those neighbors are labeled as $\boldsymbol{r}_p^{(1)}, \boldsymbol{r}_s^{(1)}, \boldsymbol{r}_p^{(2)}, \boldsymbol{r}_s^{(2)}, \ldots, \boldsymbol{r}_p^{(d)}, \boldsymbol{r}_s^{(d)}$. For each i, $\boldsymbol{r}_p^{(i)}$ denote the neighbor node's predecessor and $\boldsymbol{r}_s^{(i)}$ denote the neighbor node's successor on the ith 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. Discover Community Network

When an end-user wants to join a community, he has to discover at least one community node X. The end-user's node can either use information from an out-of-band bootstrap mechanism similar to Narada [22] and CAN [28] or can employ network broadcasting and discovery techniques such as IP multicast. In this paper we do not address these issues.

2. Join Process

As soon as the end-user's node A discovered a community node X, it sends a join request to node X. Node X is responsible to find d neighbors for node A to connect. If some joining nodes connect to the neighbors of the same node X, then the community network diameter increases linearly (e.g. completely ordered regular graph) [30]. In order to avoid such situation, each node determines autonomously the number of the new connected nodes to itself within period t that is called the node join-rate $\varphi(t)$. Node X broadcasts join-request to all the community nodes within $O(\log_{2d} M)$ layers. Each node judges autonomously to reply "Ok to join" or not based on its join-rate $\varphi(t)$. Node X receives some "OK to join" messages and autonomously selects d nodes from them in different Hamilton cycles. Then, each node from these d nodes calls the following routine to add joining node A in each ith Hamilton cycle:

```
$Add\ (A, i)\{$
    $Successor\_node \leftarrow (Calling\_node \Rightarrow r_s^{(i)});$
    $Edge\ (Calling\_node, A, i);$
    $Edge\ (A, Successor\_node, i); \}$
$Edge\ (B, C, i)\{$
    $(B \Rightarrow r_s^{(i)}) \leftarrow C;$
    $(C \Rightarrow r_p^{(i)}) \leftarrow B; \}$
```

The expression $h \Rightarrow$ Var means that we seek the value of Var from node h, the expression $(h \Rightarrow \text{Var}) \leftarrow y$ 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 as shown in Fig. 2 (a). The Edge (B, C, i) rou-

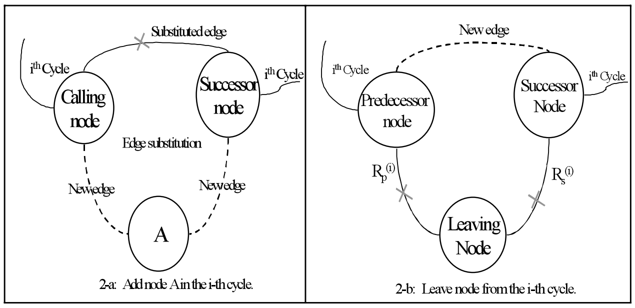
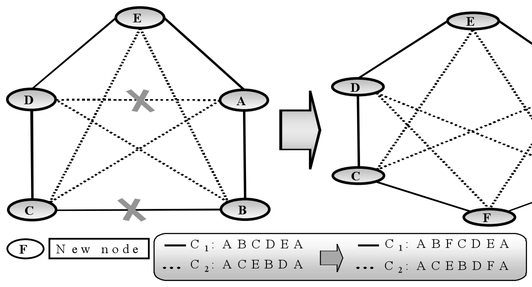
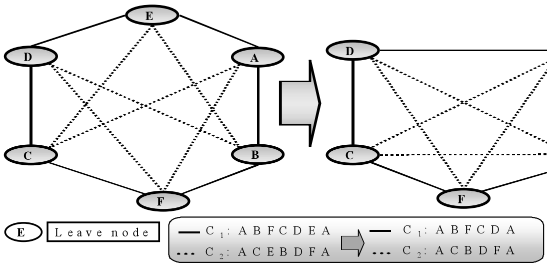


Fig. 2 Join and leave process diagram in *i*-th cycle.



(a)



(b)

Fig. 3 (a) Joining process. (b) Leaving process.

tine 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 requires only local changes in the community network. Figure 3 (a) shows an example for the joining process of new node F. Assume F discovers node E in community network then node F sends join request to E that is forward this request to all members in the community. Each node replies to node E based on its $\varphi(t)$. Then, node E selects two nodes for example, B and D from different cycles. Then, both B and D call the add routine. For clarity, we show only two Hamilton cycles C_1 and C_2. Node B has a successor node C in cycle 1 while, node D has a successor node A in cycle 2.

3. 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 ( ) { // LN is leaving node
    For $i := 1, \ldots, d$ in parallel
            do Edge $(LN \Rightarrow r_p^{(i)}, LN \Rightarrow r_s^{(i)}, i)$ }
```

The leave routine creates edges between the successor and predecessor node of the leaving node at d cycles as shown in Fig. 2 (b). For example, Fig. 3 (b) shows the leaving process of node E. It substitutes the edges (E, A) and (E, D) by new edge (D, A) in cycle 1. Similarly, it substitutes edges (E, B) and (E, C) by new edge (C, B) in cycle 2. 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. For fault-tolerance, we assumed that each node knows the predecessor node of its predecessor node and the successor node of its successor node in each cycle. The node failure is detected locally as follows. The neighboring nodes in the INS_x periodically exchange keep-alive messages with 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 execute the following instance of code.

```
FT $(x, i)$ { // x is the failed node in cycle i.
    MyPredecessor← (Calling_node $\Rightarrow r_p^{(i)}$);
    MySuccessor← (Calling_node $\Rightarrow r_s^{(i)}$);
    If (x==MyPredecessor) then
        (Calling_node$\Rightarrow r_p^{(i)}$) ← (MyPredecessor$\Rightarrow r_p^{(i)}$);
    if (x==MySuccessor) then
        (Calling_node $\Rightarrow r_s^{(i)}$) ← (MySuccessor $\Rightarrow r_s^{(i)}$); }
```

The FT routine connects two nodes around the failed node in the same cycle and sets the predecessor and successor nodes to its predecessor and successor. It maintains the Hamilton cycles and the same number of links for all nodes as well. 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, this technique maintains the community network (G) composed of edge disjoint Hamilton cycles. If a Hamilton path connects every two nodes of G then G is Hamilton-connected [31]. For example, if we construct the community network as 4-regular, 4-connected graph (cf. Fig. 3) then this graph should have two edges disjoint Hamiltonian cycles [32]. Thus, the community network is (connected) non partition-able because of considering the Hamilton cycles with number of nodes large than or equal five.

2.4 Node Autonomy

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 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. 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 autonomous node's actions are realized by the community construction technology as shown in 2. 3 and the community communication technology that will be shown in Sect. 3.1.

ACIS architecture has no central server whatsoever, as can be seen in Fig. 1. It is a fully decentralized model, where each participated node has equal responsibilities, and does not rely on any central authority to organize the network. Thus, it does not load up any single node excessively and enables the development of the high assurance information systems with adaptability, flexibility and high availability characteristics. For a timely communication among the community members this paper proposes the technology that will be described in Sect. 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 review 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 ACIS (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. Figure 4 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.

The second main idea behind the autonomous decentralized community communication technology is *multilateral benefits* 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 community members, as follow.

The proposed autonomous decentralized community communication technology performs the communication among the community members that has called "1 $\rightarrow$ N" [26], [27]. A brief scenario of the 1 $\rightarrow$ 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 community nodes received the message. The autonomy of the 1 $\rightarrow$ N communication can be seen as follow. Each

CC	**CH_i**	**Data/Request**

Fig. 4 Community communication message format.

community node recognizes autonomously a member from non-member and judges autonomously to forward community messages to only N community neighbor's nodes.

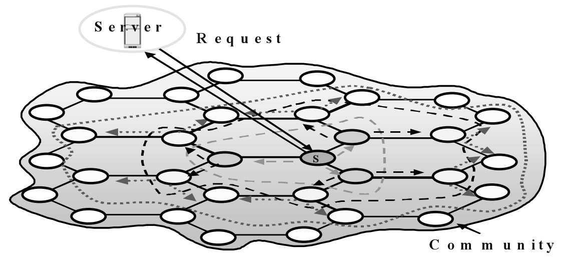
In order to avoid the congestion that may happen if some of the community nodes send simultaneously identical messages, each node keeps a short memory of the recently routed messages and judges autonomously to forward only one copy of the received messages to the other neighbor's nodes. This paper evaluates the community network traffic in such case that will be shown in Sect. 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 $1 \rightarrow N$ 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].

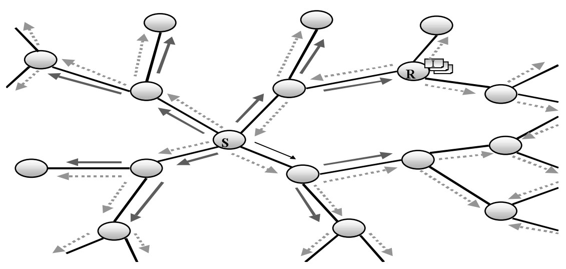
3.2 Community Communication Protocols

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 "$1 \rightarrow N$". 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) of moderate size. The publish-based protocol offers an effective solution to the flash crowd problem as shown in Fig. 5 (a). 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 community members, thereby relieving the server of this task and alleviating the load on the server. Thus, the load is distributed among the community nodes. When the number of nodes increased dramatically, 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 other 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 "$1 \rightarrow N$". 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 community members. Figure 5 (b) 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 community members enrich their experiences and/or get to know new services without requesting, what they individually cannot get to know. Thus, the multilateral benefits characteristic of the community can be satisfied and the satisfaction rate for the community members is converged to one. In addition, it decreases the traffic per node by avoiding multiple requests for the same content.



(a)



(b)

Fig. 5 (a) Publish based protocol. (b) Messages flow in request/replay-all based protocol.

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 $1 \rightarrow N$ community communication all members cooperate for the satisfaction of all community members contrary to the peer-peer (P2P) communication techniques. In P2P, peers cooperate for the satisfaction of only one, which requests 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 ACIS 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 forward 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.

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

Where L is the number of layers or depth of the communication tree. The term $\sum_{i=0}^{L-1} (k-1)^i$ is a geometric series has summation $((K-1)^L - 1)/(k-2)$. Then the number of layers can be calculated as follows.

$$L \approx \left\lceil \frac{\log\left[\frac{(M-1)\times(K-2)}{K} + 1\right]}{\log\,(K-1)} \right\rceil \leq \frac{\log(M)}{\log(k-1)} \tag{3}$$

Under the assumption that the average communication cost between each node is approximately 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 $1 \rightarrow N$ communication as follows.

$$\begin{aligned} \frac{d\tau}{dN} &= \frac{d}{dN}(N \times \log_N(M)) = \frac{d}{dN}\left(N \times \frac{\log(M)}{\log(N)}\right) \\ &= \log(M) \times \left(\frac{\log(N) - 1/\ln(10)}{(\log(N))^2}\right) \end{aligned} \tag{4}$$

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 $(1 \rightarrow 3)$ community communication technology is the optimal. Similarly, Fig. 6 shows that ever-increasing the number of nodes the optimal communication is $1 \rightarrow 3$ under the assumption that the average communication cost between each node is one unit of time.

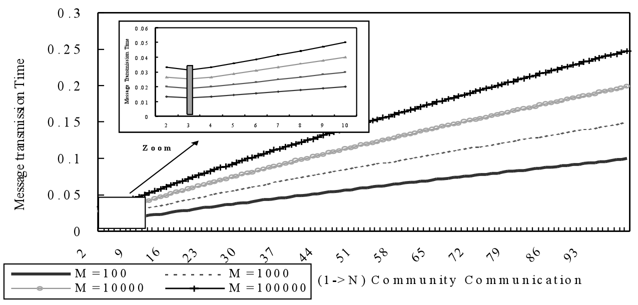
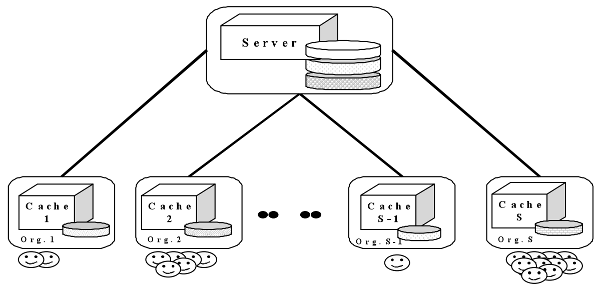
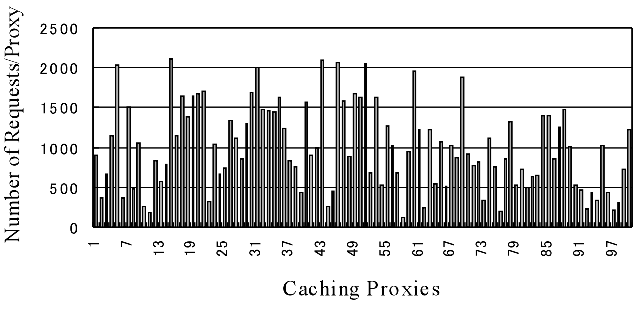


Fig. 6 Optimal $1 \rightarrow N$ community communication.



(a)



(b)

Fig. 7 (a) Caching proxies: One-to-one communication simulation model. (b) Distribution of users' requests per caching proxies.

4.1 Simulation and Results

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. Figure 7 (a) 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 will randomly assigned to S caching proxies. Figure 7 (b) shows an example of the distribution of the number of users' requests per caching proxy (e.g. proxy 5 has 2035 requests). It has been proved that a chasing proxy has an upper bound of 30–50% in its hit

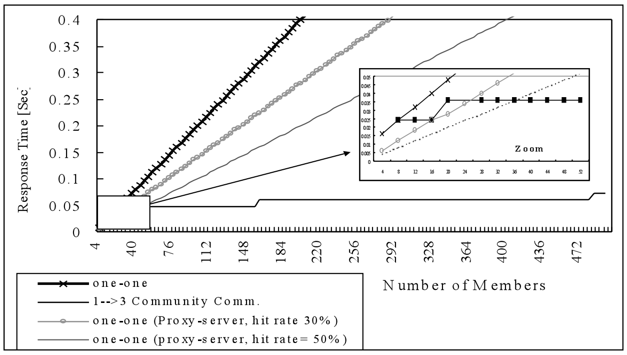


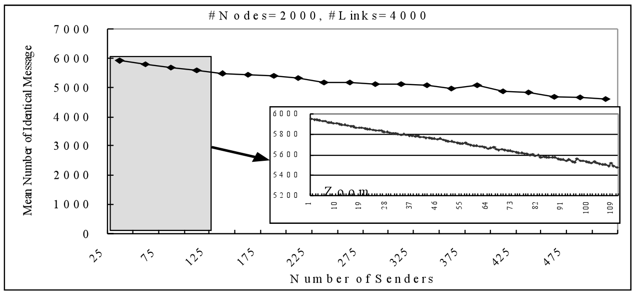
Fig. 8 Simulation result: Scalable announcement.

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 $1 \rightarrow 3$ communication technology and is constituted of average communication cost between each node $\tau_{cc} = 1$ ms. $\tau_m = 1$ ms, the average time that each node needs to monitor the recent received messages to avoid the congestion. Thus the transmission time τ to send a message asynchronously from any node to all the other community members is bounded by $L \times N \times (\tau_{cc} + \tau_m)$.

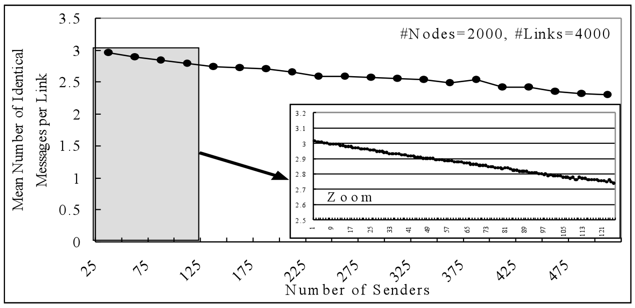
We concentrate in these experiments on the comparison between the conventional one-to-one communication techniques without and with caching proxy (hit rate of 30%, 50%) and ($1 \rightarrow N$) community communication technology. ACIS gathers those clients. As soon as one client (member) has downloaded an interesting content for the community from the server, she/he publishes it to all the community members using "$1 \rightarrow N$". Figure 8 shows the variations of the number of the members in the community with the worst transmission time of a message to all members. Figure 8 depicts the effectiveness of the proposed communication technology in compared with the conventional ones. The $1 \rightarrow N$ communication technology is able to send a message to all the community members within an average of less than about 6 times in compare with the one-one communications. Furthermore, it shows that the community communication technology is scalable of the response time with the number of the members. For a very small number of members, our proposed community communication technology is not effective but it reveals interesting results when the total number of members in the community increases dramatically as shown in the zoom part in Fig. 8.

4.2 Community Network Traffic

We ran an additional experiment to evaluate the network traffic in case of 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 2000 nodes and 4000 logical links, which were generated as regular graph [16]. The delay of each link was set



(a)



(b)

Fig. 9 (a) Network traffic with multiple senders. (b) Network traffic with multiple senders.

to 1 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. Figures 9 (a) and 9 (b) 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 number of senders is increased from one sender to about 25% senders from the participated nodes in the community network. In addition, zoom parts in both Figs. 9 (a) and 9 (b) show the variation of MINM with the increases of the number of senders from 1 to 125. The increase of the number of senders reduces both MNIM in the community network and MNIM per link. Figure 9 (a) shows about 22% improvement of MNIM when 25% nodes send identical message at once. The standard deviation of MNIM per link is about 0.078. 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 community system size. We conclude that the community's network traffic is reduced when many members send an identical message at once.

5. Related Work

ACIS, 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 target collaborative applications with a small number of group members. However, ACIS is a framework for collaborative applications with a large number of group members. ALMI takes a centralized

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
ACIS	Decentralized Loosely control	Autonomous Members	Large	Multi

approach to the tree creation problem. Clearly, it constitutes a single point of failure for all control operations related to the group. In contrast, ACIS 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 ACIS, 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. ACIS 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 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 ACIS 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 (ACIS) 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.

6. Conclusion

This paper clarifies the concept, architecture, construction/maintenance and communication technologies of the community information system. Inspired from the constructive cooperation in the social community and the ADS concept, the ACIS concept 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 bilateral-hierarchy system architecture has been developed in order to sustain the proposed concept. Finally, the autonomous decentralized community construction/maintenance and communication technology have been proposed to achieve a productive cooperation and a flexible and timely communication among the community members. The proposed 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. peer-peer (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.

References

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

[2] P. Lyman and H.R. Varian, "How much information," J. Electronic Publishing, vol.6, no.2, pp.17–24, Dec. 2000.

[3] B. David, 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," Computer, vol.31, no.4, pp.42–44, 1998.

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

[6] M. Arlitt and T. 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," 11th Int. World Wide Web Conference, Hawaii, USA, May 2002.

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

[9] J. 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. First Int. Sym. on ADS, (ISADS'93), pp.28–34, IEEE, Kawasaki, 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, pp.1056–1063, Nov. 1986.

[12] V.N. Gudivada, V.V. Raghavan, W.I. Grosky, and R. Kasanagottu, "Information retrieval in the World Wide Web," IEEE Internet Comput., vol.1, no.5, pp.58–68, Oct. 1997.

[13] K. Mori, "Assurance system architecture for information service by utilizing autonomous mobile agents," Fifth IEEE International High-Assurance Systems Engineering Symposium, Nov. 2000.

[14] R. Dornfest, "Dark matter, sheep and the cluster: Resolving metaphor collision in P2P," O'Reilly Peer-to-Peer and Web Services Conference, Washington, D.C., Nov. 2001.

[15] M. Ripeanu, "Peer-peer architecture case study: Gnutella network," Proc. Int. Conf. on P2P Computing, 2001.

[16] K. Ragab, T. Ono, N. Kaji, and 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," Proc. Fourth Symposium on Operating System Design and Implementation (OSDI), pp.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," Proc. 11th Int. Workshop (NOSSDAV), June 2001.

[19] K. Ragab, T. Ono, N. Kaji, and K. Mori, "Community communica-

tion technology for achieving timeliness in autonomous decentralized community systems," Proc. IEEE IWADS, pp.56–60, Beijing, China, Nov. 2002.

[20] FIPS 180-1 Secure hash standard, Washington D.C., April 1995.

[21] S. Williams, "Caching proxies: Limitations and potentials," 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," Proc. ACM Sigmetrics, pp.1–12, June 2000.

[23] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure," Proc. 3rd 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," IEEE INFOCOM 2003, 2003.

[26] K. Ragab, N. Kaji, K. Moriyama, and K. Mori, "Scalable multilateral communication technique for large-scale information systems," IEEE COMPSAC 2003, 2003.

[27] K. Ragab, N. Kaji, and K. Mori, "Service-oriented autonomous decentralized community communication technique for a complex adaptive information system," Proc. IEEE/WIC, Halifax, Canada, Oct. 2003.

[28] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," Proc. SIGCOMM'01, California, USA, 2001.

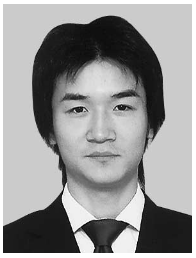
[29] S. Deering and D. Cheriton, "Multicast routing in datagram Internetworks and extended LANs," ACM Trans. Computer Systems, vol.8, no.2, pp.85–110, May 1990.

[30] A. Oram, Peer-to-Peer Harnessing the Power of Disruptive Technologies, O'Reilly & Associates, March 2001.

[31] J.A. Bondy and U.S.R. Murty, Graph Theory with Applications, Macmillan Press, 1976.

[32] B. Grünbaum and J. Malkevitch, "Pairs of edge-disjoint Hamiltonian circuits," Aequationes Math., vol.14, no.1/2, pp.191–196, 1976.

Khaled Ragab received the B.Sc. and M.Sc. degress in Computer Science from Ain Shams University, Cairo, Egypt in 1990, 1999, respectively. He has worked in Ain Shams University, Cairo Egypt in 1990-1999 as assistant lecturer. He has worked as research scientist in Computer Science Dept., Technical University of Chemnitz, Germany in 1999-2001. He joined Department of Computer Science, Tokyo Institute of technology in 2001 as doctoral student under the supervision of professor Kinji Mori. His research interests includes autonomous decentralized systems, cooperative information systems and application-level multicast.



Naohiro Kaji received the B.E. and M.S. degrees in information engineering from Tokyo Institute of Technology, Japan, in 2002, 2004, respectively. In 2004 he joined Ministry of Economy, Trade and Industry. His research interests are autonomous decentralized systems and mobile computing.

Kinji Mori received the B.S., M.S., and Ph.D. degrees in the Electrical Engineering from Waseda University, Japan in 1969, 1971, and 1974, respectively. From 1974 to 1997 he was with System Development Lab., Hitachi, Ltd. In 1997 he joined Tokyo Institute of Technology, Tokyo, Japan as a professor. His research interests include the distributed computing, the fault tolerance computing, the assurance and the mobile agent. He proposed Autonomous Decentralized Systems (ADS) in 1977 and since then he has involved in the research and the development of ADS.