Supporting an Experiment of a Community Support System:

Community Analysis and Maintenance Functions in the Public Opinion Channel

Tomohiro Fukuhara[1], Masaki Chikama[2], Toyoaki Nishida[3]

[1]Synsophy Project, Communications Research Laboratory, Japan

[2]Graduate School of Information Science, Nara Institute of Science and Technology, Japan

[3]Graduate School of Information Science and Technology, The University of Tokyo, Japan

tomohi-f@acm.org, masaki-c@is.aist-nara.ac.jp, nishida@kc.t.u-tokyo.ac.jp

Abstract. Community analysis and maintenance functions of a community support system called Public Opinion Channel (POC) are described. A field-test and a psychological experiment using a community support system are important for investigating activities in a community. Existing community support systems, however, have limitations on having an experiment smoothly because few systems consider functions for supporting an experiment. To investigate activities in a community smoothly, community support systems should have community analysis and maintenance functions that support an analyst and a community organizer in an experiment. We implemented those functions in the POC system, and used in a field-test and a social psychological experiment. From these experiments, we found that proposed functions enabled an analyst and a community organizer to (1) find up-to-date state of a community during an experiment, (2) analyze relations between messages efficiently and objectively, and (3) maintain communities easily. Requirements for the functions, implementation, and experience in the experiments are described.

1. Introduction

The Internet enabled us to find information and people easily. We can find people who have similar interests or goals, and form or join a network community. A network community has possibilities for creating artifacts, sharing knowledge, and solving problems more effectively than traditional organizations such as companies and universities. Linux and free software communities are good examples of successful network communities. We consider that fine interactions, which enable community members to create and share knowledge effectively, exist in those successful communities. We call those communities *knowledge-creating (KC) communities*.

Besides KC communities, there are *sick communities* where too few or too much information is exchanged. There occurs a *flaming*, which is an endless quarrel between community members, frequently in a sick community. In another community, the deflation of communication called *spiral of silence* hinders exchange of opinions (Noelle-Neumann, 1984). There are few fine interactions in sick communities

Our concerns are factors that form and sustain a KC community. What are differences between KC and sick communities? How can we form a KC community? What kind of interaction sustains a KC community? Although we don't have clear answers for these questions, we are tackling them by creating and evaluating a community support system called PUBLIC OPINION CHANNEL (POC) (Fukuhara et al., 2003).

Through development and experiments of the POC system, we found the importance of an experiment stage in the development cycle of a community support system. We can acquire valuable data in an experiment. Based on acquired data, we can improve the system and have a next experiment. Meanwhile, having an experiment is not easy because maintaining communities and analyzing data are burdens for researchers. Functions for supporting researchers to have an experiment smoothly are needed for investigate a KC community efficiently.

In this paper, we propose community analysis and maintenance functions of a community support system. We implemented those functions in the POC system. Through a field-test and a social psychological experiment using the POC system, we found feasibilities of the proposed functions for having an experiment smoothly.

This paper is organized as follows. First, we analyze the development cycle of a community support system, and describe requirements for community analysis and maintenance functions (Section 2). We will then describe an overview of the implemented functions in the POC system (Section 3). Next, we will describe our experience in a field-test and a social psychological experiment using the POC system (Section 4). We will compare our proposal with relevant works (Section 5). We summarize arguments, and describe the future work (Section 6).

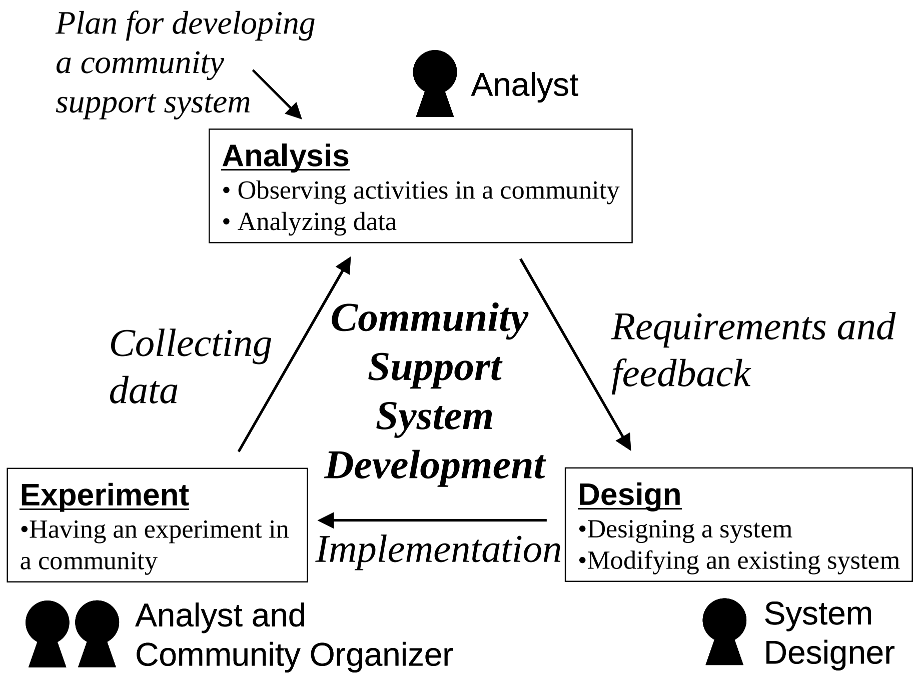

Plan for developing a community support system
Analyst
Analysis
• Observing activities in a community
• Analyzing data
Collecting data
Community Support System Development
Requirements and feedback
Experiment
•Having an experiment in a community
Implementation
Design
•Designing a system
•Modifying an existing system
Analyst and Community Organizer
System Designer

Figure 1. Development cycle of a community support system.

2. Community analysis and maintenance functions in a community support system

In this section, we describe requirements for community analysis and maintenance functions by analyzing the development cycle of a community support system.

2.1 Development cycle of a community support system

To investigate a KC community, repeating a cycle that consists of implementation and evaluation of a community support system smoothly is important. Figure 1 shows the development cycle of a community support system.

The cycle consists of (1) *analysis*, (2) *design*, and (3) *experiment* stages.

(1) *Analysis stage*. In this stage, an analyst observes activities in a community, and specifies requirements for supporting those activities. S/he tells the requirements to a system designer for designing a system.

(2) *Design stage*. The system designer designs and implements a prototype system according to the requirements. The system is improved in case of an existing system. The system is tested in the experiment stage.

(3) *Experiment stage*. The analyst cooperates with a community organizer to have an experiment. During an experiment, the analyst observes activities of community members, and records data such as utterances and

operations of members. The organizer supports the analyst by doing miscellaneous works such as creating new communities, configuring settings of them, adds or removes accounts of community members, and removes unnecessary messages and communities. The analyst analyzes data, and gives feedback to the designer for improving the system.

Repeating this cycle smoothly is important for both of development of the system, and investigation into a KC community.

2.2 Problems in an experiment of a community support system

Having an experiment is not easy because analyzing data and maintaining communities are burdens for an analyst and a community organizer. In a long-term field-test that is held for several months, a large volume of data is acquired. To find important facts from those data is not easy for an analyst (Nishida et al., 1998; Bruckman et al., 2001). Furthermore, it is difficult for an analyst to follow the latest state of a community during a field-test because s/he is busy observing activities and recording data. Furthermore, s/he might also have to maintain communities.

From the viewpoint of a community organizer, s/he has to engage in miscellaneous works during a field-test. An organizer has to prepare communities before a field-test. S/he also has to prepare community members' accounts (for account-based systems). During a field-test, an organizer has to check messages posted by members. In case of inappropriate messages, an organizer has to remove them. Without an easy interface for configuring settings of a community, an organizer has to edit configuration files of the system manually. To edit configuration files, an organizer has to have much knowledge on the system such as internal structure for changing settings of a community. It is not practical for an organizer to have such miscellaneous knowledge because s/he is not a designer.

Community analysis and maintenance functions are needed for having an experiment smoothly. Previous works on field-tests of community support systems (Nishibe et al., 1998; Matsuda et al., 2002), however, do not mention those functions. Although frameworks of a community support system are proposed (Koch and Lacher, 2000; Yoshida et al., 2000), their focuses are not on community analysis and maintenance works. The lack of community analysis and maintenance functions hinders system development and investigation into KC communities.

Community support systems should have community analysis and maintenance functions. Those functions should facilitate (1) an analyst to collect and analyze data, and (2) a community organizer to configure settings of a community. We describe requirements for these functions in the following subsections.

2.3 Requirements for a community analysis function

Followings are requirements for a community analysis function.

2.3.1 Logging and analyzing server logs

Logging and analyzing *server logs* that are stored in a server application are needed. In case of a server-client system, actions taken by a client application can be identified by the server. Recording and analyzing server logs are important for understanding activities of community members.

Server logs should be analyzed automatically because too many logs to analyze manually are collected in an experiment. A community analysis function should provide basic statistic data that indicate an overview of activities for an analyst.

2.3.2 Storing and analyzing messages

A community analysis function should store and analyze *messages* that are information exchanged between community members. Analyzing messages, especially relations between them, is important for finding key messages that activated or hindered discussions in a community. Because there are implicit relations between messages that cannot be identified by reference structures, the function should analyze both of implicit and explicit relations.

2.4 Requirements for a community maintenance function

Followings are requirements for a community maintenance function.

2.4.1 Maintaining settings of a community

The function should allow a community organizer to maintain a community. During an experiment, there often occur needs for changing settings of communities such as adding another accounts to a community. The function should not require an organizer for having much knowledge on the system. An easy interface is needed because an organizer might not be a computer expert.

2.4.2 Maintaining accounts of community members

The function should allow an organizer to maintain accounts of community members. In an experiment, there occur needs for adding accounts to a community, changing properties of accounts, and remove unnecessary accounts from the community. The function should enable an organizer to engage in those works.

2.4.3 Maintaining messages

The function should enable an organizer and an analyst to check and maintain messages. In case of a field-test that is held in an open environment such as the Internet, there might be spam messages that disturb ordinary discussions. The function should allow organizer and analyst to check contents of messages, and remove spam messages.

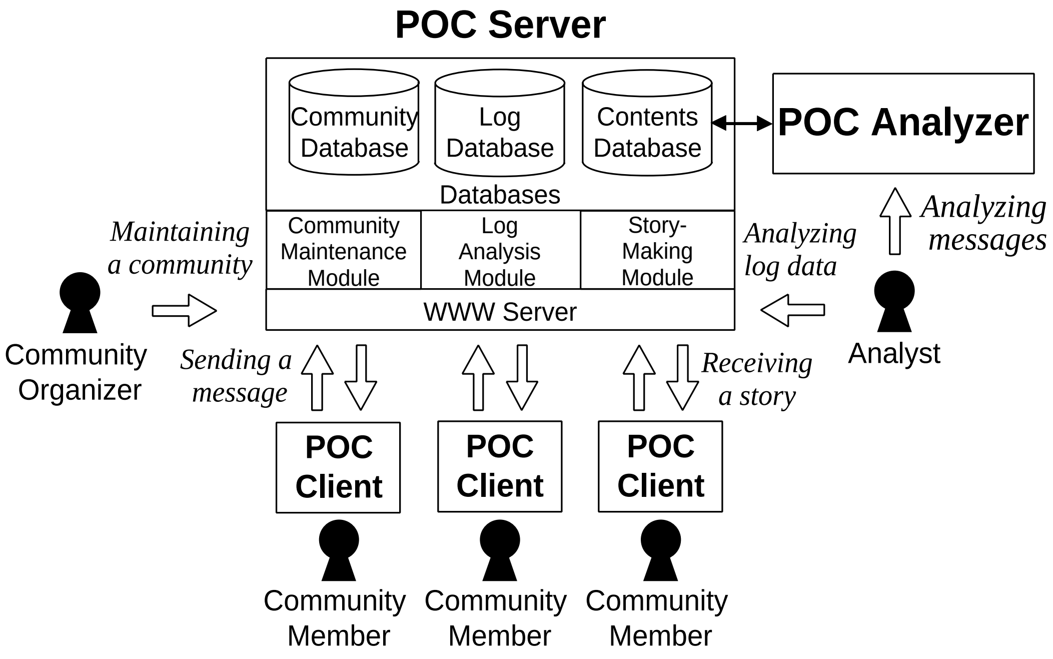


Figure 2. Architecture of the POC system.

3. Community analysis and maintenance functions in the POC system

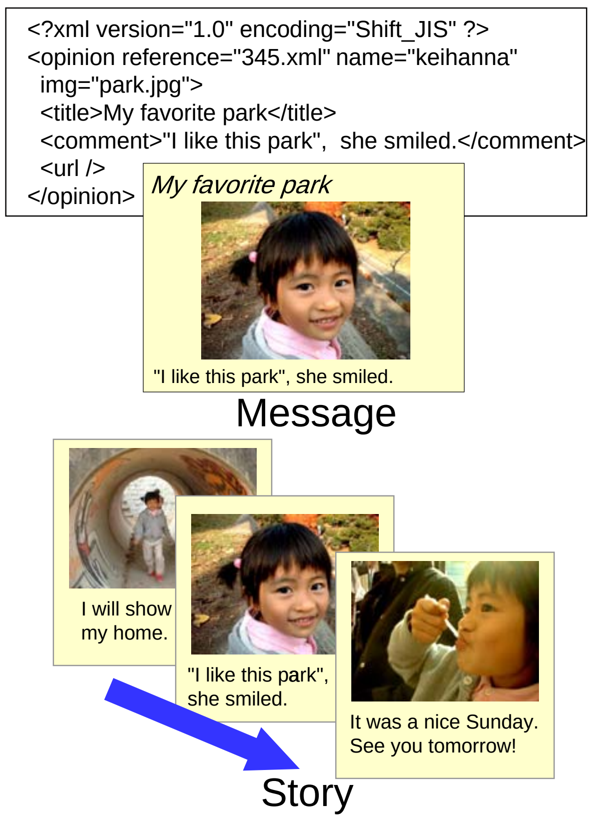
We implemented the requirements for community analysis and maintenance functions in the POC system. We describe architecture of the system, and implemented functions.

3.1 Architecture of the POC system

Figure 2 shows the architecture of the POC system. The system consists of (1) *POC server*, (2) *POC clients*, and (3) POC ANALYZER.

(1) *POC server*. POC server is the main part of the POC system. The server provides basic services for POC clients, and community maintenance and analysis functions for a community organizer and an analyst. The server consists of (a) *log analysis module*, (b) *community maintenance module*, and (c) *story-making module*.

   a) *Log analysis module*. This module analyzes server logs stored in the log database. The module generates *reports* that contain graphs and basic statistic data on a community and its members. The server records specific actions of POC clients in the log database. An example of a server log and a list of actions are described in Section 3.2.1.



```
<?xml version="1.0" encoding="Shift_JIS" ?>
<opinion reference="345.xml" name="keihanna"
  img="park.jpg">
  <title>My favorite park</title>
  <comment>"I like this park",  she smiled.</comment>
  <url />
</opinion>
```

My favorite park
"I like this park", she smiled.
Message
I will show my home.
"I like this park", she smiled.
It was a nice Sunday. See you tomorrow!
Story

Figure 3. Message and story.

b) *Community maintenance module*. This module allows a community organizer to configuring settings of a community. The module also manages messages and stories of the POC system (see Figure 3). A message consists of meta-information (in the opinion tag), title, body (in the comment tag), and URL. A story consists of several messages.

c) *Story-making module*. This module creates a story automatically based on relevant messages (Kamada et al., 2002).

(2) *POC clients*. The clients are tools for community members. There are several clients. POC COMMUNICATOR is a client for sending a message to the POC server (Fukuhara et al., 2002). POCTV is a client for watching stories in a talk show style (Kubota and Nishida, 2001).

(3) POC ANALYZER. This is a tool for analyzing relations between messages. It accesses to the contents database of the POC server, and analyzes messages in the database. Relations between messages are represented as a network. POC ANALYZER shows the network graphically.

3.2 Community analysis function

As community analysis functions, we describe (1) the log analysis module and (2) POC ANALYZER.

.

Line	Event
1	TRY LOGIN account=person1 community=kc_talk [2002-07-10 19:09:55]
2	ACK LOGIN account=person1 community=kc_talk [2002-07-10 19:09:56]
3	LIST COMMUNITIES IP=xx.xxx.xx.xx account=person1 clientName=COMMUNICATOR [2002-07-10 19:09:58]
4	LIST MESSAGES client=COMMUNICATOR account=person1 [2002-07-10 19:10:00]
5	SEARCH MESSAGE query='wind' clientName=COMMUNICATOR account=person1 community=kc_talk found=47 [2002-07-10 19:16:20]
6	ACK MESSAGE IP=xx.xxx.xx.xx file=bbs107.xml account=person1 community=kc_talk [2002-07-10 19:23:31]
7	LOGOUT community=kc_talk IP=xx.xxx.xx.xx account=person1 [2002-07-10 19:48:43]

Table 1. Example of a server log

Token	Description
ACK LOGIN	A client succeeded in logging in.
NACK LOGIN	A client failed to log in.
LIST COMMUNITIES	A client requested a community list.
LIST MESSAGES	A client requested a message list.
LIST STORIES	A client requested a story list.
SEARCH MESSAGES	A client retrieved messages.
SEARCH STORIES	A client retrieved stories.
ACK MESSAGE	A client posted a message.
ACK STORY	A client posted a story.
LOGOUT	A client exited from a community.

Table 2. List of tokens.

The log analysis module and POC ANALYZER are components of the *SIQ (Social Intelligence Quantity) package* that is a set of metrics for evaluating communication tools (Yamashita and Nishida, 2002). The log analysis module and POC ANALYZER are tools for analyzing quantitative data in the SIQ package.

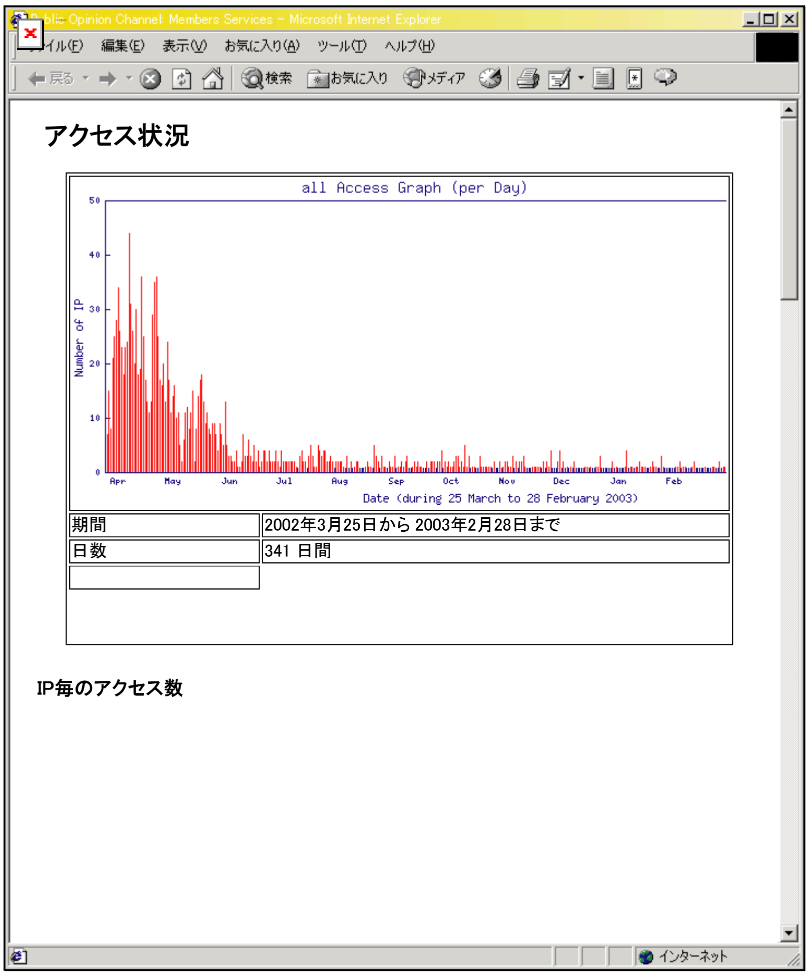
3.2.1 Log analysis module in the POC server

The log analysis module supports an analyst to analyze server logs. Table 1 shows an example of a server log that indicates a series of actions taken by an account called 'person1'. A log consists of several *events*. An event indicates specific actions taken by a POC client. In Table 1, each line represents an event. For example, line 6 indicates that 'person1' posted a message whose filename is 'bbs107.xml' to the 'kc_talk' community.

Event type can be identified by *token*. Table 2 shows a list of tokens. An event consists of token, IP, date, and optional parameters such as account, and client name.

The log analysis module consists of several CGI scripts (written in Perl) that run on a WWW server. An analyst can analyze log on a Web browser. S/he can browse various reports exploratory by changing parameters such as a period of time, community name, and account.

There are following reports in the POC server.


Public Opinion Channel: Members Services - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る 検索 お気に入り メディア
all Access Graph (per Day)
Number of IP
0
10
20
30
40
50
Apr May Jun Jul Aug Sep Oct Nov Dec Jan Feb
Date (during 25 March to 28 February 2003)
インターネット

アクセス状況

期間	2002年3月25日から2003年2月28日まで
日数	341 日間

IP毎のアクセス数

Figure 4. Report on number of access.

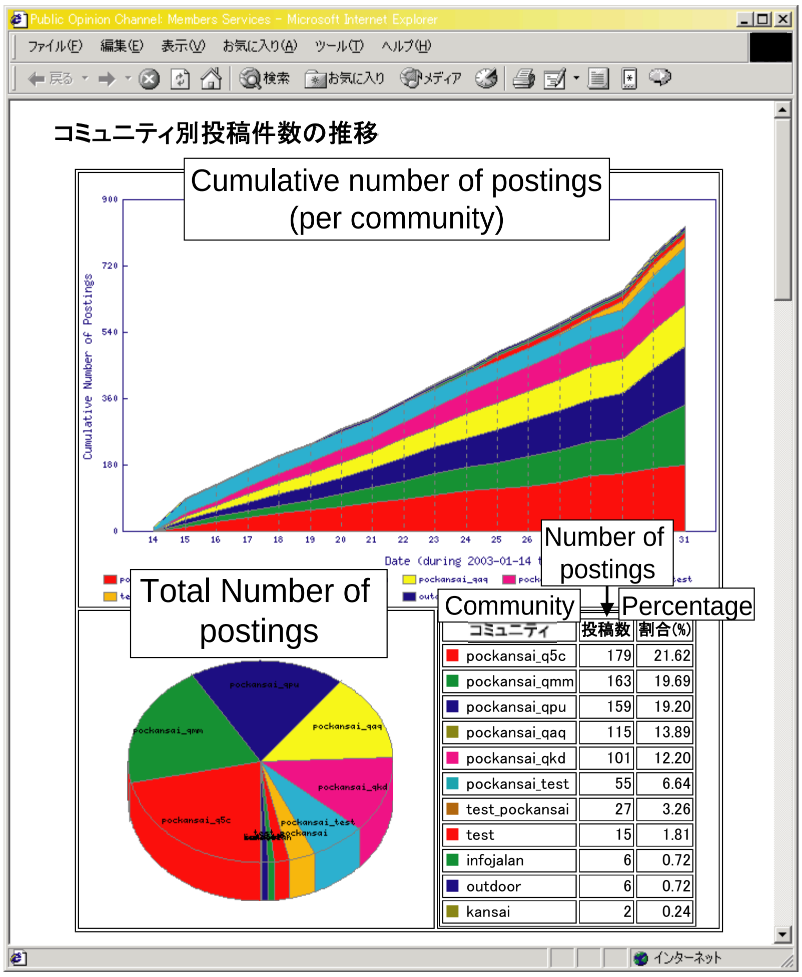
(1) *Number of access*. This report provides a summary of access to the POC system (see Figure 4). The figure shows total number of access / IP / postings in a specified term. The figure also shows number of access per IP as a ranking list. By clicking a button in the list, an analyst can browse a *browsing behavior report* (described later) for each IP.

(2) *Postings per community*. This report analyzes number of postings per community (see Figure 5). The figure shows cumulative number of postings, and total number of postings per community.

(3) *Posting behavior*. This report analyzes posting actions for each account (see Figure 6). The figure shows posting actions taken by an account ‘q5c552’. This report shows number of postings per day, daily trend of postings, and a list of messages and stories posted by this account.

(4) *Browsing behavior*. This report analyzes browsing actions of each account. The report shows number of messages and stories browsed by an account.

(5) *Contents ranking*. This report shows frequently accessed contents.



コミュニティ	投稿数	割合(%)
🟥 pockansai_q5c	179	21.62
🟩 pockansai_qmm	163	19.69
🟦 pockansai_qpu	159	19.20
🟫 pockansai_qaq	115	13.89
🟪 pockansai_qkd	101	12.20
🟦 pockansai_test	55	6.64
🟫 test_pockansai	27	3.26
🟥 test	15	1.81
🟩 infojalan	6	0.72
🟦 outdoor	6	0.72
🟫 kansai	2	0.24

Figure 5. Report on postings per community.

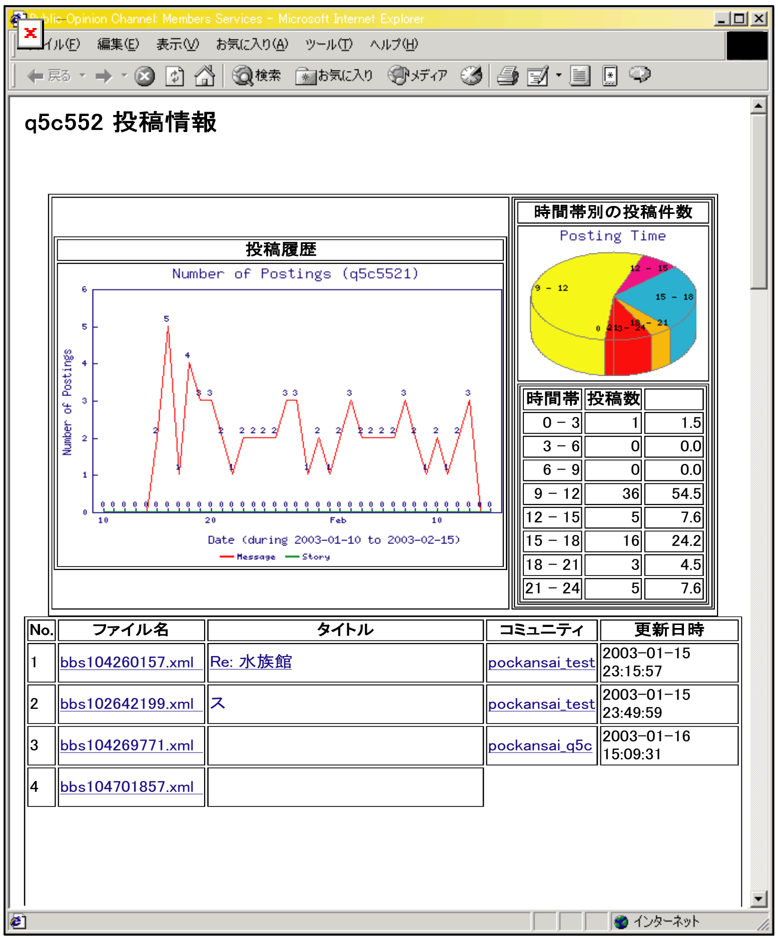
(6) *Session*. This report analyzes duration of each session per an account. A session can be identified by ‘ACK LOGIN’ and ‘LOGOUT’ tokens.

(7) *Recent postings*. This report analyzes messages and stories posted in the last three days. One can browse contents of messages and stories.

3.2.2 POC ANALYZER

POC ANALYZER is a tool for analyzing relations between messages. This tool is written in C, and runs as a Linux application. Figure 7 shows a screen image of POC ANALYZER. Relations between messages are represented as a *network* where messages are represented as *nodes* and relations are represented as *directed links*. POC ANALYZER shows a network graphically. By looking at a network graph, an analyst can find key messages in the network.

POC ANALYZER consists of (1) *parameter field*, (2) *list field*, (3) *graph field*, and (4) *information field*.



ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る
検索
お気に入り
メディア
投稿履歴
Number of Postings (q5c5521)
Number of Postings
Date (during 2003-01-10 to 2003-02-15)
Message
Story
Posting Time
9 - 12
12 - 15
15 - 18
18 - 21
21 - 24
インターネット

q5c552 投稿情報

時間帯別の投稿件数

時間帯	投稿数	
0 − 3	1	1.5
3 − 6	0	0.0
6 − 9	0	0.0
9 − 12	36	54.5
12 − 15	5	7.6
15 − 18	16	24.2
18 − 21	3	4.5
21 − 24	5	7.6

No.	ファイル名	タイトル	コミュニティ	更新日時
1	bbs104260157.xml	Re: 水族館	pockansai_test	2003−01−15 23:15:57
2	bbs102642199.xml	ス	pockansai_test	2003−01−15 23:49:59
3	bbs104269771.xml		pockansai_q5c	2003−01−16 15:09:31
4	bbs104701857.xml			

Figure 6. Report on posting behavior.

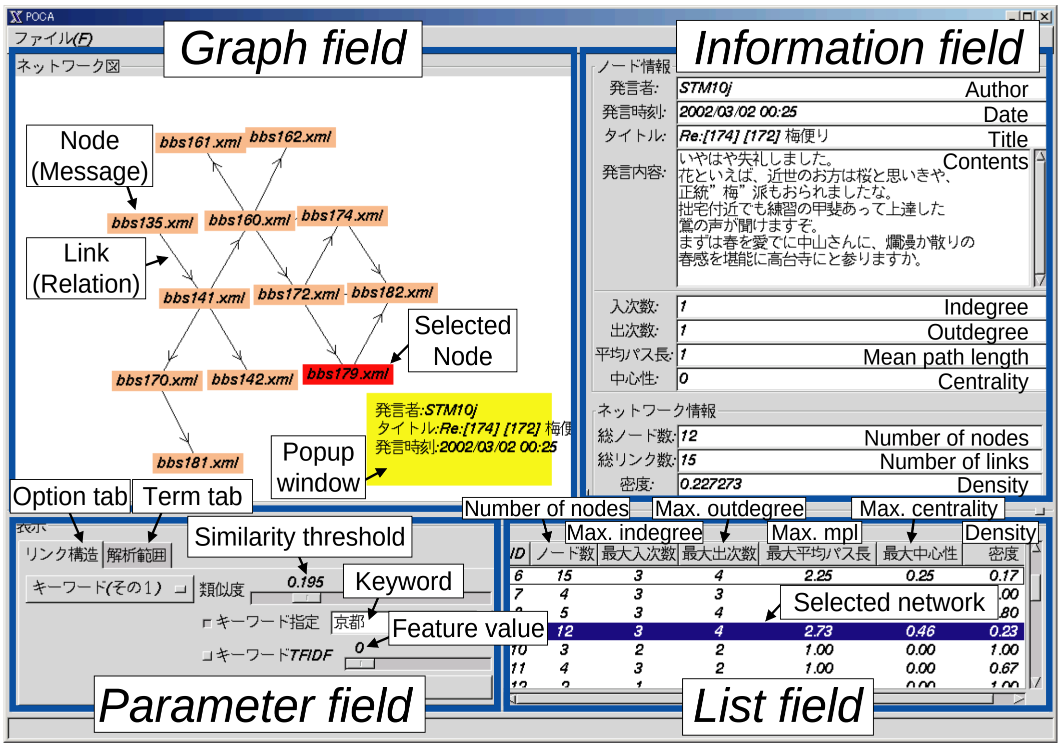
(1) *Parameter field*. In this field, an analyst can create networks by specifying parameters. S/he can specify similarity threshold between messages, and keyword and its feature value in the option tab. S/he can also specify the period of time in the term tab.

(2) *List field*. This filed shows a list of networks that meet specified parameters in the parameter field. An analyst can select a network

according to feature values of a network. Feature values include number of nodes, maximum number of *indegree* and *outdegree* that indicate number of incoming and outgoing links of a node (Wasserman and Faust, 1994, p.126), and maximum number of *mean path length* (*mpl*) / *centrality / density* (meanings of these values are described later).

(3) *Graph field*. This field displays a graph of the selected network in the list field. A popup window, which shows an overview of a node, appears when a mouse cursor is over the node.

(4) *Information field*. This field shows (a) information on the selected node in the graph field (such as author and date), and (b) information on the selected network in the list field (such as number of nodes/links).



発言者:STM10j
タイトル:Re:[174] [172] 梅便り
発言時刻:2002/03/02 00:25

ノード情報

発言者:	STM10j	Author
発言時刻:	2002/03/02 00:25	Date
タイトル:	Re:[174] [172] 梅便り	Title
発言内容:	いやはや失礼しました。 花といえば、近世のお方は桜と思いきや、 正統”梅”派もおられましたな。 拙宅付近でも練習の甲斐あって上達した 鶯の声が聞けますぞ。 まずは春を愛でに中山さんに、爛漫か散りの 春感を堪能に高台寺にと参りますか。	Contents
入次数:	1	Indegree
出次数:	1	Outdegree
平均パス長:	1	Mean path length
中心性:	0	Centrality

ネットワーク情報

総ノード数:	12	Number of nodes
総リンク数:	15	Number of links
密度:	0.227273	Density

キーワード(その1)
類似度 0.195
キーワード指定 京都
キーワードTFIDF 0

ID	ノード数	最大入次数	最大出次数	最大平均パス長	最大中心性	密度
6	15	3	4	2.25	0.25	0.17
7	4	3	3	[illegible]	[illegible]	.00
[illegible]	5	3	4	[illegible]	[illegible]	.80
[illegible]	12	3	4	2.73	0.46	0.23
10	3	2	2	1.00	0.00	1.00
11	4	3	2	1.00	0.00	0.67
12	2	1	[illegible]	[illegible]	0.00	1.00

Figure 7. POC ANALYZER.

POC ANALYZER analyzes relations according to the following steps.

(1) Create a network by finding links between two messages.

(a) Make a link when either one has a reference to the other.

(b) Make a link when similarity is larger than threshold τ .

(2) Calculate feature values for the network.

Firstly, POC ANALYZER creates a network according to the reference structure and similarity among messages. In case of the reference structure, a link is created between messages. A link begins from a message that has a reference attribute, and ends with the referred message. In case of similarity, POC ANALYZER calculates similarity between two messages based on the vector space model (Salton, 1989).

In the vector space model, a message is represented as a vector. A vector consists of feature values for terms that appeared in the message. A feature value (w_{ij}) for the j-th term ($term_{ij}$) of the i-th message ($\boldsymbol{D}_i$) is calculated by

$$w_{ij} = tf_{ij} \cdot \log(\frac{N}{df_j}),$$

where tf_{ij} is the word frequency of $term_{ij}$ in $\boldsymbol{D}_i$, N is the total number of terms, and df_j is the number of messages that contain $term_{ij}$.

$\boldsymbol{D}_i$ is a vector that consists of feature values (w_{ij}) for each term in the message.

$$\boldsymbol{D}_i = (w_{i1}, w_{i2}, ..., w_{iN})$$

Similarity between messages p and q ($sim(\boldsymbol{D}_p, \boldsymbol{D}_q)$) is calculated by the following formula.

$$sim(\boldsymbol{D}_p, \boldsymbol{D}_q) = \frac{\sum_{i=1}^{N} w_{pi} \cdot w_{qi}}{\sqrt{\sum_{i=1}^{N} w_{pi}^2} \cdot \sqrt{\sum_{i=1}^{N} w_{qi}^2}}$$

The direction of a link is from an old message to the new one.

After creating a network, POC ANALYZER calculates feature values of the social network analysis. POC ANALYZER calculates (1) *density*, (2) *centrality* (we use *betweenness centrality*), (3) *mean path length*.

Density (D) of a network is a feature value that represents the quantity of links in the network (Wasserman and Faust, 1994, pp.101-102). D is calculated by

$$D = \frac{l}{L_{\max}},$$

where l is the actual number of links in the network, and L_{max} is the maximum number of possible links for the network, i.e., a node has links with any other nodes in the network. A network that has high density value can be regarded as a homogeneous network that contain same or similar topics.

Betweenness centrality of a node N_i ($Cb(\mathrm{N_i})$) is a feature value that represents the influence of a node N_i in a network (Wasserman and Faust, 1994, pp.188-191). $Cb(\mathrm{N_i})$ is calculated by

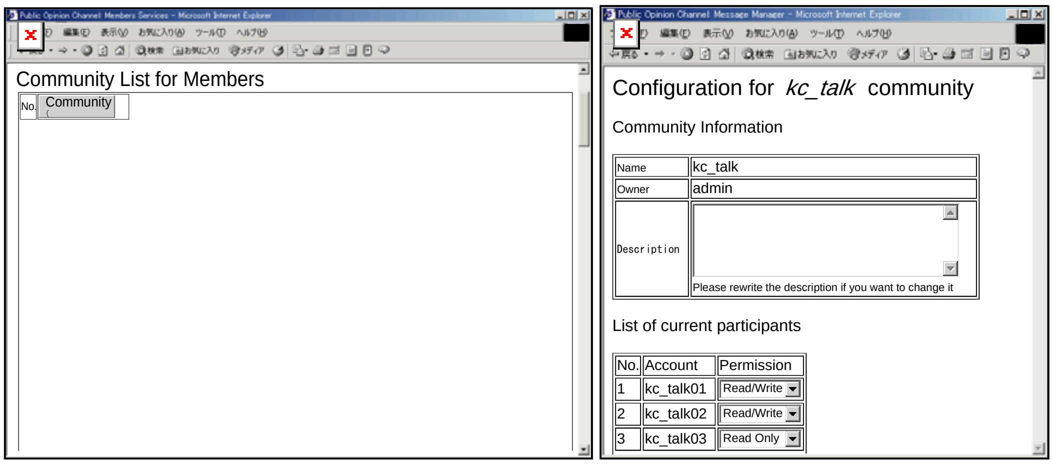
$$Cb(N_i) = \frac{L_{jk}(N_i)}{L_{jk}},$$

where $L_{jk}(N_i)$ is the number of *geodesics* between any nodes N_j and N_k that pass through N_i. A geodesic is the shortest sequence of links between two nodes (Wasserman and Faust, 1994, p.110). There might be several geodesics between two nodes. L_{jk} is the total number of geodesics between N_j and N_k. A node that has high betweenness centrality value can be regarded as a node that summarized previous topics, and stimulated community members to post further messages.

Mean path length of a node N_i ($mpl(N_i)$) is the average of path length between nodes N_i and its descendant nodes that can be found by following outgoing links. POC ANALYZER calculates $mpl(N_i)$ as

$$mpl(N_i) = \frac{\sum_{j \in Desc(N_i)} g(N_i, N_j)}{count(Desc(N_i))},$$

where $Desc(N_i)$ is a set of descendant nodes of N_i, $count(Desc(N_i))$ is the number of nodes in $Desc(N_i)$, and $g(N_i, N_j)$ is the *geodesic distance* between N_i and N_j. N_j is a descendant node of N_i. A geodesic distance is the length of a geodesic (Wasserman and Faust, 1994, p.110). A node that has high mean path length value can be regarded as a message whose topic affects many messages in a community.


Public Opinion Channel: Members Services - Microsoft Internet Explorer
Public Opinion Channel: Message Manager - Microsoft Internet Explorer

Community List for Members

No.	Community

Configuration for *kc_talk* community

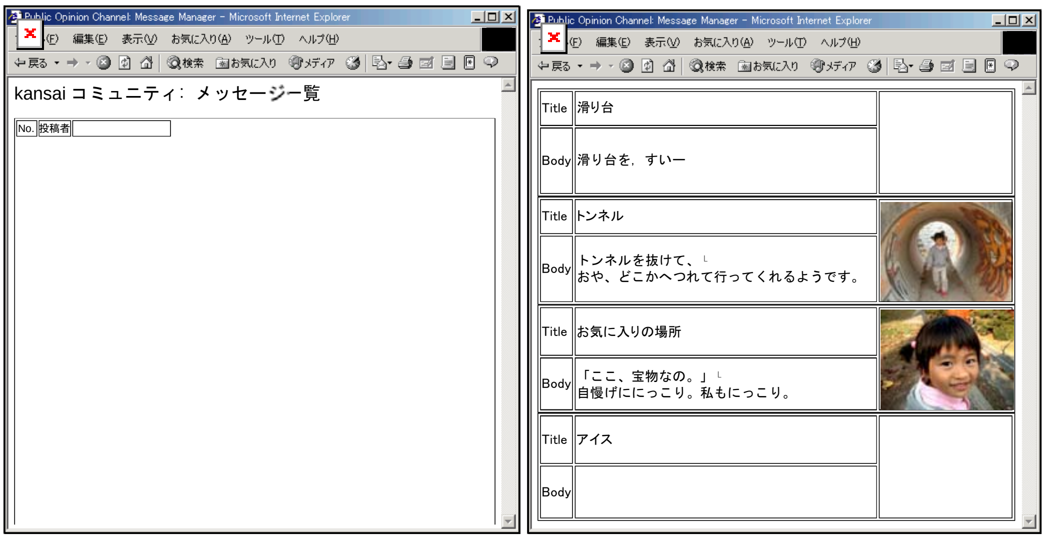
Community Information

Name	kc_talk
Owner	admin
Description	Please rewrite the description if you want to change it

List of current participants

No.	Account	Permission
1	kc_talk01	Read/Write
2	kc_talk02	Read/Write
3	kc_talk03	Read Only

Figure 8. Web pages for browsing and configuring settings of communities.


Public Opinion Channel: Message Manager - Microsoft Internet Explorer
(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る 検索 お気に入り メディア

kansai コミュニティ: メッセージ一覧

No.	投稿者	

Title	滑り台	
Body	滑り台を，すいー	
Title	トンネル	
Body	トンネルを抜けて、 おや、どこかへつれて行ってくれるようです。	
Title	お気に入りの場所	
Body	「ここ、宝物なの。」 自慢げににっこり。私もにっこり。	
Title	アイス	
Body		

Figure 9. Web pages for browsing messages and stories.

3.3 Community maintenance function

The community maintenance module provides Web pages for browsing and configuring settings of a community. A community organizer can create a community, add accounts for newcomers of a community, and change permissions of accounts. Figure 8 shows web pages for browsing and configuring settings of communities. An organizer can find number of messages and stories posted in a community, and s/he can configure settings of a community. Figure 9 shows Web pages for browsing messages and stories. An organizer and an analyst can check contents of a message and a story. S/he can also remove unnecessary messages and stories.

Figure 10. Technical staff of the helpdesk.

4. Evaluation

We describe experience in a long-term field-test called KDDI FTTH trial, and a social psychological experiment using the POC system.

4.1 KDDI FTTH trial

We had a field-test of the POC system in the KDDI FTTH (Fiber to the Home) trial. The trial was held for one year (from March 25 in 2002 to February 28 in 2003). 443 households in Shinjuku and Bunkyo areas in Tokyo participated in the trial for evaluating broadband services such as VOD (Video on Demand) and videoconference. We provided the POC system as one of services.

Our aim in this field-test was to find tacit knowledge on Shinjuku and Bunkyo areas, and circulate the knowledge among participants. Therefore, we hosted communities that treat topics on local information on Shinjuku and Bunkyo areas.

For supporting participants and observing activities of them, we set up (1) a *helpdesk*, and (2) an *analysis team*. The helpdesk, which consists of one technical staff (see Figure 10), played the role of a community organizer (see Figure 1). Her tasks include (a) answering questions from participants, (b) checking contents of postings, and (c) maintaining postings and communities.

The analysis team, which consists of a cognitive psychologist and a social psychologist, played the role of an analyst (see Figure 1). The team estimated system usage by analyzing server logs. They provided feedback for the system designers for improving the system.

We found following merits of the proposed functions.

(1) Understanding current state of communities.
(2) Analyzing server logs instantly.
(3) Maintaining communities easily.

4.1.1 Understanding current state of communities

The log analysis module enabled the analysis team to find current state of communities. Figure 11 shows number of access to the log analysis module. Total number of access was 2,258. Most of access was made by the number of access report that shows the latest access to the system (see Figure 4). During the trial, the team was able to follow up-to-date state of communities objectively by browsing this report. The team checked this report frequently by reloading its Web pages regularly (e.g., in the morning, after having a lunch, and before leaving the office).

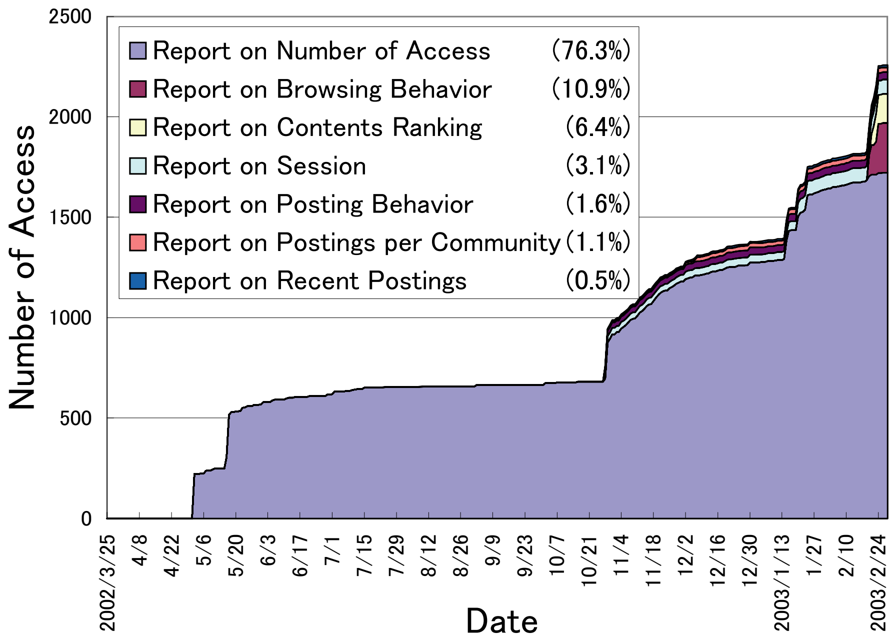


Figure 11. Usage of the log analysis module in the FTTH trial.

4.1.2 Analyzing server logs instantly

The log analysis module enabled the analysis team to analyze server logs instantly. The module had not been implemented until early May (see Figure 11). It was hard for the team to analyze server logs manually. They had to write scripts for parsing and extracting data from logs. It took much time to analyze data. Although they were able to reuse the scripts, it was inconvenient to log in the POC server, and runs scripts on a UNIX terminal.

By using the log analysis module, the team was able to analyze server logs easily. Although we acquired 9.8MByte (118,713 events) of server logs in the trial, the module was able to analyze them around 10 seconds (tested under a server machine that has Pentium III 800MHz dual processors and 1GB memory). The time required for analyzing logs was not a serious issue during the trial. The log analysis module was extended in the latter term of the trial (around November 4 in Figure 11) according to the suggestions of the analysis team.

4.1.3 Maintaining communities easily

The technical staff was able to maintain communities easily. Before the community maintenance module was implemented, the staff had to ask a system designer to configure settings of communities. It was inconvenient for both of staff and the designer. After implementing the module, she was able to configure settings of communities, and maintain messages and stories by herself. She was able to engage in her works on her Web browser easily.

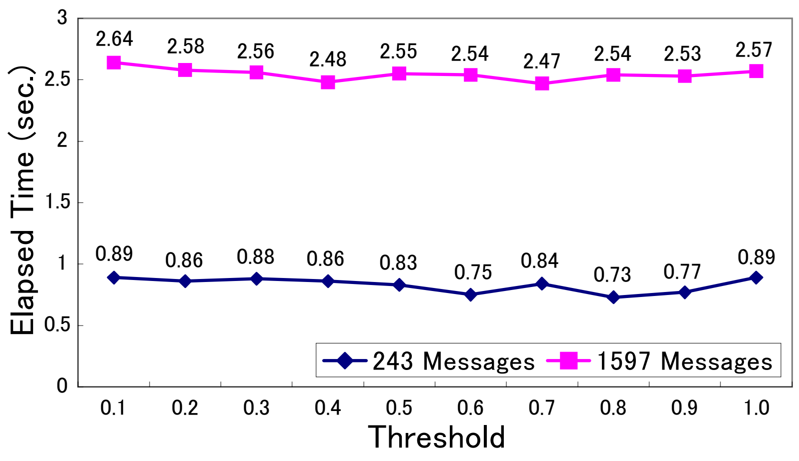


Figure 12. The time required for analyzing messages.

4.2 Social psychological experiment

We had a social psychological experiment using the POC system (Matsumura et al., 2002). In this experiment, 20 subjects were asked to exchange messages using the system with other subjects for four weeks. 243 messages were exchanged in this experiment. We found followings merits of the proposed functions.

(1) Analyzing messages efficiently and objectively.
(2) Understanding activities in a community totally.

4.2.1 Analyzing messages efficiently and objectively

POC ANALYZER enabled a psychologist to analyze relations between messages efficiently and objectively. Figure 12 shows the time required for analyzing messages. The average time for analyzing 243 messages was 0.83 seconds (tested under a workstation that has Xeon 2.4GHz single processor and 512MB memory). For comparison, we also measured time using another set of messages that contain 1,597 messages. The average was 2.55 seconds for 1,597 messages. A psychologist was able to analyze messages efficiently. Furthermore, because POC ANALYZER analyzes messages based on objective criteria, several psychologists were able to share and discuss the same analysis results.

4.2.2 Understanding activities in a community totally

The combination of results from log analysis module and POC ANALYZER was useful for analyzing activities in a community totally. By using the log analysis module, the psychologist was able to check recent activities in a community. Once he found interesting activities, he analyzed messages by using POC ANALYZER. By combining results from both tools, the psychologist was able to analyze activities in a community by complementing each result.

5. Related work

Hilbert proposed an environment for testing usability of software (Hilbert and Redmiles, 1998). This environment monitors and collects user interface (UI) events remotely. This approach is suitable for understanding microscopic activities of the user of software. Meanwhile, it is not suitable for understanding macroscopic activities of the user such as results in the log analysis module.

With respect to a communication tool that has an analysis function, there are CollabLogger and LumberJack (Morse and Steves, 2000; Chi et al., 2002). Focuses of these tools are on the log analysis. Meanwhile, our focus is on total support of an experiment. We focus on not only the log analysis, but also message analysis and community maintenance. Implemented functions supported community maintenance and analysis works.

With respect to the message analysis, several message analysis tools are proposed (Sack, 2000; Smith and Fiore, 2001). With respect to the log analysis tools, there are tools for analyzing server logs such as Analog (www.analog.cx), Report Magic (www.reportmagic.org), and Lire (www.logreport.org/lire/). If we look at either POC ANALYZER or log analysis module only, we hardly find differences between those tools and implemented functions. An important point, however, is that the combined use of analysis tools is important for analyzing activities in a community totally. In the social psychological experiment, the psychologist was able to analyze activities in a community by using both of analysis tools.

6. Summary and future work

We described community analysis and maintenance functions in the POC system. Having an experiment of a community support system smoothly is important for investigating a KC community. We proposed requirements for community analysis and maintenance functions, and implemented those requirements in the POC system. We tested implemented functions in the FTTH trial and a psychological experiment. The implemented functions supported community maintenance and analysis works in those experiments.

Our proposal is a basic support for an analyst and a community organizer during an experiment of a community support system. We believe that our proposal is one of first steps for investigating a KC community. Our future work is to continue to have experiments using the POC system, and gain feedback from analysts and community organizers.

Acknowledgements

The authors would like to thank KDDI Co., Ltd. for giving us a chance for joining the FTTH trial.

References

Bruckman, A., Erickson, T., Fisher, D., and Lueg, C. (2001): 'Dealing with Community Data: A Report on the CSCW 2000 Workshop', *SIGCHI Bulletin*, Vol. 33, No. 4, p.13. (http://www.acm.org/sigchi/bulletin/2001.4/comm_data.pdf)

Chi, E., Rosien, A., and Heer, J. (2002): 'LumberJack: Intelligent Discovery and Analysis of Web User Traffic Composition', *Proc. WEBKDD 2002 - ACM-SIGKDD Workshop on Web Mining for Usage Patterns and User Profiles*, Edmonton, Canada, Jul. 2002.

Fukuhara, T., Nishida, T., and Uemura, S. (2002): 'POC Communicator: a System for Collaborative Story Building'. *Proc. Knowledge-Based Intelligence Engineering Systems and Allied Technologies (KES2002)*, Podere d'Ombriano, Italy, Sep. 2002, pp.1336-1340 (in Vol.4).

Fukuhara, T., Fujihara, N., Azechi, S., Kubota, H., and Nishida, T. (2003): 'Public Opinion Channel: a Network-Based Interactive Broadcasting System for Supporting a Knowledge-Creating Community'. In Howlett, R., Ichalkaranje, N., Jain, L., and Tonfoni, G. (eds.): *Internet-Based Intelligent Information Processing Systems*, chapter 7, World Scientific Publishing.

Hilbert, D. and Redmiles, D. (1998): 'Agents for Collecting Application Usage Data over the Internet'. *Proc. Autonomous Agents – 2nd Intl. Conf. on Autonomous Agents*, Minneapolis, MN, USA, May 1998, pp.149-156.

Kamada, K., Kurohashi, S., and Nishida, T. (2002): 'Story Generation Method from Disordered Messages'. *Proc. 8th Annual Conference of the Association for Natural Language Processing*, Kyoto, Japan, Mar. 2002, pp.363-366 (in Japanese).

Koch, M. and Lacher, M. (2000): 'Integrating Community Services: A Common Infrastructure Proposal'. *Proc. Knowledge-Based Intelligence Engineering Systems and Allied Technologies (KES'2000)*, Brighton, UK, Sep. 2000, pp.56-59.

Kubota, H. and Nishida, T. (2001): 'Maintaining Knowledge of Conversational Agents'. *Proc. Knowledge-Based Intelligence Engineering Systems and Allied Technologies*, Osaka, Japan, Sep. 2001, pp.329-333.

Matsuda, K., Miyake, T., and Kawai, H. (2002): 'Culture Formation and its Issues in Personal Agent-oriented Virtual Society: PAW^2'. *Proc. Collaborative Virtual Environments (CVE2002)*, Bonn, Germany, Sep. 2002, pp.17-24.

Matsumura, K., Azechi, S., Yamashita, K., and Fukuhara, T. (2002): 'Psychological Effects of Participants on the Networked Community'. *Proc. KMN'02 – IEEE Intl. Workshop on Knowledge Media Networking*, Kyoto, Japan, Jul. 2002, pp.73-77.

Morse, E. and Steves, M. (2000): ‘CollabLogger: A Tool for Visualizing Groups at Work’, *Proc. WET ICE’00 – IEEE 9th Intl. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Gaithersburg, MD, USA, Mar. 2000, pp.104-109.

Nishibe, Y., Morihara, I., Hattori, F., Nishimura, T., Yamaki, H., Ishida, T., Maeda, H., and Nishida, T. (1998): ‘Mobile Digital Assistants for International Conferences’, In Ishida, T. (ed.): *Community Computing*, chapter 8, John Wiley & Sons.

Nishida, T., Bowker, G., Mason, J., Miyashita, T., Nagao, K., Nishimura, T., Ohguro, T., Sidhu, C., Sumi, Y., Van den Besselaar, P., and Yokozawa, M. (1998): ‘Methodology for Large Scale Experimentation: A Discussion Report’, In Ishida, T. (ed.): *Community Computing and Support Systems (Lecture Notes in Computer Science; 1519)*, pp.11-15, Springer.

Noelle-Neumann, E. (1984): ‘The Spiral of Silence: Public Opinion? Our Social Skin’, University of Chicago Press.

Sack, W. (2000): ‘Conversation Map: A Content-Based Usenet Newsgroup Browser’. *Proc. IUI - 5th Intl. Conf. on Intelligent User Interfaces*, New Orleans, LA, USA, Jan. 2000, pp.233-240.

Salton, G. (1989): ‘Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer’. Addison-Wesley.

Smith, M. and Fiore, A. (2001): ‘Visualization Components for Persistent Conversations’. *Proc. ACM CHI2001 – Conf. on Human Factors in Computing Systems*, Seattle, WA, USA, Mar. 2001, pp.136-143.

Wasserman, S. and Faust, K. (1994): ‘Social Network Analysis: Methods and Applications’, Cambridge University Press.

Yamashita, K. and Nishida, T. (2002): ‘SIQ (Social Intelligence Quantity): Evaluation Package for Network Communication Tools’, In Dai, G. (ed.): *Proc. APCHI2002 - 5th Asia Pacific Conference on Computer Human Interaction*, Beijing, China, Nov. 2002, pp. 271-280, Beijing: Science Press.

Yoshida, S., Kamei, K., Ohguro, T., Kuwabara, K., and Funakoshi, K. (2000): ‘Building a Network Community Support System on the Multi-Agent Platform SHINE’. *Proc. PRIMA2000 - Third Pacific Rim Intl. Workshop on Multi-Agents*, Melbourne, Australia, Aug. 2000, pp.88-100.