Shine: a Cyber-community Application Platform — A Proposal —

Sen Yoshida, Takeshi Ohguro, Koji Kamei, Kaname Funakoshi, and Kazuhiro Kuwabara

NTT Communication Science Laboratories
{yoshida,ohguro,kamei,kf,kuwabara}@cslab.kecl.ntt.co.jp

Abstract. We are developing an environment in a cyber-society that will support people in the cyber-society to communicate with members in communities, and to form, maintain, and evolve communities. This paper proposes a common platform that provides the foundation for the various applications to be used in this environment. The platform, named Shine, is designed as a multi-agent architecture. We will discuss requirements of the design such as the organization of modules, communication protocols for agents, and user modeling.

1 The necessity of socialware and its platform

As public communication networks such as the Internet continue to grow, societies outside of real society, i.e., cyber-societies, are going to continue to appear on the networks. A cyber-society has the advantage of providing communication capabilities to real society that are free from geographical or temporal constraints. Because people are unable to deal with too much information or too many other humans, however, they will tend to lose proper perspective for acting or will hesitate to act and end up isolated in such a society.

In real society, although we only meet others in a physically limited sense, all of us are successful in creating communities of various people. In a community, both people with common interests and goals, and similar knowledge, and people with differing interests and goals, and different levels of knowledge, exist. We can enjoy conversations with those sharing our interests and collaborate with those sharing our goals. Moreover, we can deepen our knowledge in discussions with those having different opinions. In this way, we can actively communicate within relatively small but satisfactorily heterogeneous groups. Moreover, community feelings develop from such continual communications, and these community feelings help further smooth the communications that takes place.

Communities in a cyber-society must also develop in a similar way for improved communications. Actually, some active communities already exist in cyber-societies. At present, however, many people are worried about the overabundance or the lack of information and people, and so it is difficult to form and maintain communities. To overcome this difficulty, a new communication environment is needed that can provide an appropriate range of activities for

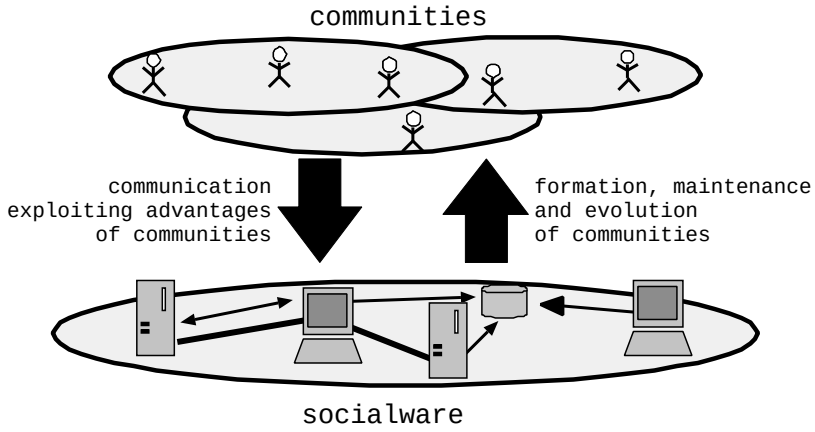

Fig. 1. Socialware

each person, can understand the diversity of knowledge and situations among community members, and can help achieve the potential of the community.

With this motivation, research on communication systems for communities on networks has been growing recently [3]. We call such systems "socialware" [2]. Socialware supports

- communications exploiting advantages of communities, and
- community formation, maintenance, and evolution

as shown in Fig. 1.

It is easy to think of many applications of socialware. Several applications for communications exploiting advantages of communities are follows:

- Circulation of information via human networks, such as word-of-mouth communications, recommendations, and collaborative filtering [12].
- Arrangement of informal, suggestive or creative conversations, such as brainstorming sessions [1, 5].
- Establishment of efficient collaborations for developing resources such as software and databases.
- Arrangement of localized and informal economic activities, such as bazaars and flea markets.

On the other hand, several applications supporting community formation, maintenance, and evolution are as follows:

- Arrangement of effective meetings and notification of community feelings [8, 13].
- Coordination of collaborations and conversations in a group [6, 7].
- Planning for establishing user identities in communities.

While we can think of a variety of socialware applications, some of the functions of these applications are commonly required. These functions include:

1. The dynamic adaptation of acquaintance relations or group formation,
2. An analysis of each person's features, role, and situation within a community,
3. An interface that links a user's community feeling and a system's logical information, and
4. Flexible and intuitive communication utilities.

At the present, however, there is no software environment able to integrate these common functions. Consequently, there exists no cooperation among application programs, or the sharing and reuse of program parts in application development. For this reason, we propose a platform named Shine[1] as a common base for various socialware applications and discuss how the platform is achieved. Specifically, Shine is designed as a multi-agent system.

2 The functions required by Shine

In this section, we examine the functions required by Shine because they are common to various socialware applications.

2.1 Dynamic adaptation of acquaintance relations and group formation

Most of the socialware application programs available today were designed as client-server (CS) systems, and so their control mechanisms and data are centralized. Such conventional CS architectures, however, isolate the activities of the clients of each server, and consequently, users are unable to expand their human relations with users of other servers. On the other hand, providing a huge unique server to overcome this limitation is far from realistic from the viewpoint of scalability.

In Shine, we adopt a decentralized, multi-agent architecture to tackle this problem. In our approach, the control mechanisms and data for human relations are distributed to each person. Namely, the Shine system provides each user an agent that is dedicated to the person's social interactions.

In a decentralized multi-agent system, no agent knows every other agent. It is therefore necessary to consider how agents will encounter other agents. In real society, people meet many unknown others but can flexibly reorganize groups to match situations. Our goal is to bring the group dynamism of real society to cyber-societies. In Shine, an agent exchanges personal information with other agents and they all collaboratively perform the necessary actions to change group formation dynamically (e.g., mediation or introduction).

[1] The name "Shine" is derived from the Japanese word Hikaridai (meaning the tableland of shine), which is the name of the location where Shine is being developed.

2.2 Analysis of each person's feature, role and situation within a community

To analyze a person's features, role, or situation in a community systematically, a socialware application has to adopt a kind of user model. By following a specific user model, the application can store data about the attributes and status of others into a database and can use the data for analysis.

One of a number of user models can be used depending on how it regards a human. For example, when a user model considers a human to be a knowledge source, it can describe the person in a knowledge representation language [9]. To roughly represent a person's interests, feature vectors of the vector space model are suitable [13]. In addition, there is a method that prescribes communication processes as protocols and describes communication entities as state transition systems [4] to model each person's situation and conversation flow.

Each current socialware application uses a suitable user model. For Shine to achieve cooperation among application programs and the sharing of parts in application development, however, it requires fundamental data types and operations to abstract all of the user models. Accordingly, Shine adopts object-oriented programming with the notion of inheritance, and provides a library of such fundamental data types.

2.3 An interface that links a user's community feeling and a system's logical information

A socialware application program needs a user interface that allows users to relate their feelings about acquaintances to the system and to understand intuitively the meaning of the logical information owned by the system following a user model. Therefore, Shine not only provides popular user models but also interface libraries for those models. Such interface libraries of user models may include visualization mechanisms and other mechanisms that use available human-computer interaction media.

2.4 Flexible and intuitive communication utilities

Complex communications takes place in communities in that it is impossible to describe when who is talking about what where. Besides, when we try to determine whom to communicate with, especially when we want to broadcast a message to community members, it is important that the communications must adapt to all dynamic changes in the community formation and be linked to the community feelings of users. Shine provides a function that can flexibly determine the range of communications based on the data including the features and status of others.

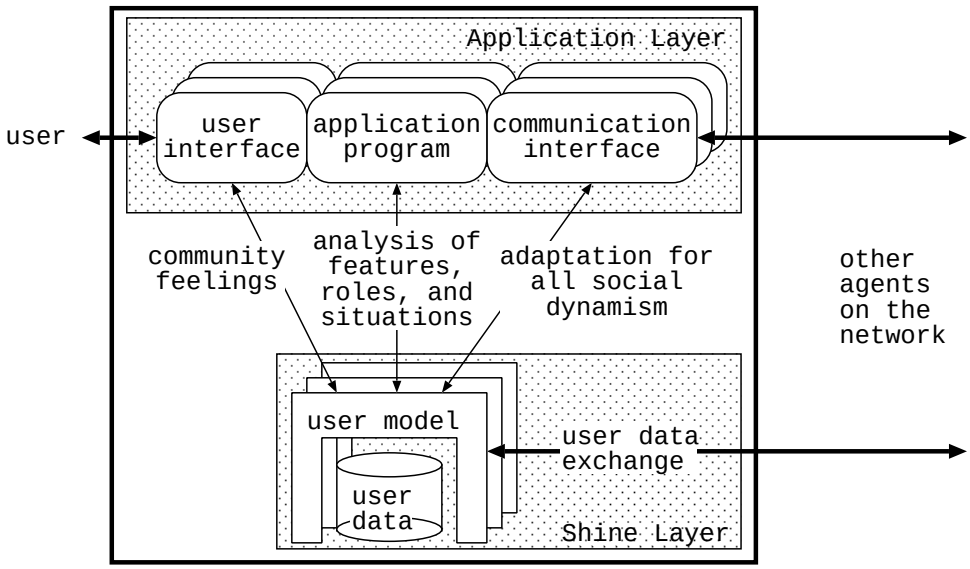

Fig. 2. Internal structure of a Shine agent

3 Design and implementation of Shine

In the previous section, we examined functions that are common to various socialware applications and therefore required to be implemented by Shine. As discussed, it is suitable for Shine to be designed as a multi-agent system.

Figure 2 shows the design of a Shine agent's internal structure. The agent is divided into two layers: one layer depends on the domain of applications (Application Layer) and the other layer is common to Shine applications (Shine Layer). The Shine Layer supports each module of the Application Layer as follows.

- For the user interface, Shine helps to link a user's community feeling and a system's logical information.
- For the body of an application program, Shine enables a person to take advantage of being in a community by analyzing the features, roles, and situations of people.
- For the interface enabling communications with others, Shine provides the ability to adapt to the dynamic changes in acquaintance relations and group formation.

In addition, Shine Layers themselves communicate with each other to exchange user data for mediation and/or member-introduction purposes when other functions require them. Accordingly, the network of Shine agents is also layered as shown in Fig. 3.

Incidentally, the following conditions are required for a programming language to implement Shine applications.

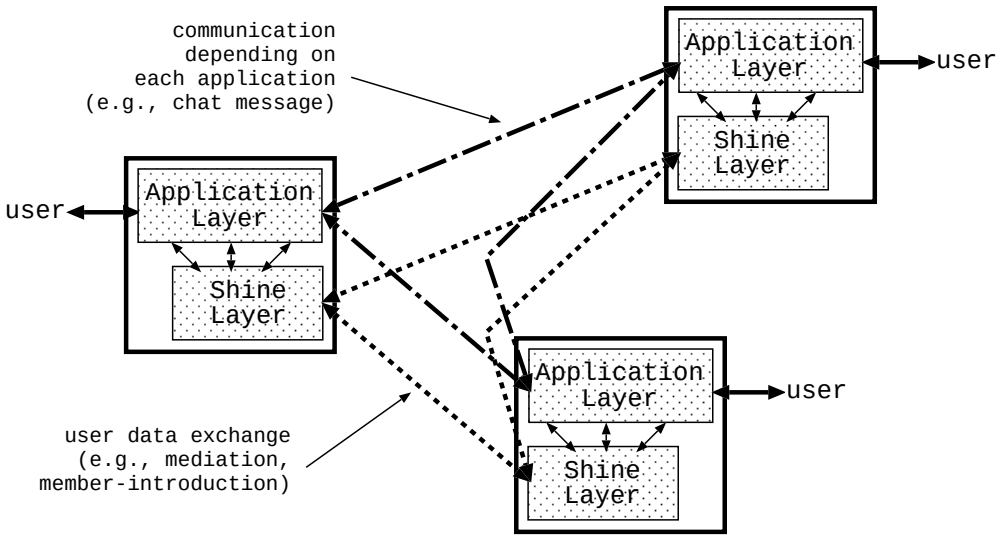

Fig. 3. Layered network of agents

- It should be suitable for distributed environments.
- It should be object-oriented for the purpose of abstracting user models.
- It should allow programmers to easily build user interfaces able to handle community feelings.

Java is one of the languages that satisfies these conditions and that has been widely popularized. We will therefore provide Shine as a Java class library.

4 Example: Community Organizer

This section describes an example of a Shine application. The Community Organizer [2, 13] is a socialware application that the authors are developing. This system supports the formation of new cyber-society communities as well as the communications among community members by visualizing potential communities in accordance with the degree of common interest of users. Figure 4 is a sample image of the Community Organizer.

This system calculates the degree of common interest (i.e., a relevance value) for each user pair, and then places user icons in a plane to reflect the relevance values between the pairs. An icon that corresponds to a user is placed in the center of his/her view. All of the users can see information about other users who share common interests, and can broadcast messages to them. This system is expected to encourage all users to meet and exchange ideas by providing such information.

Although the present version of the Community Organizer is an independent client/server system, it can be revised for porting to the Shine platform.

Fig. 4. Sample image of the Community Organizer (Japanese version)

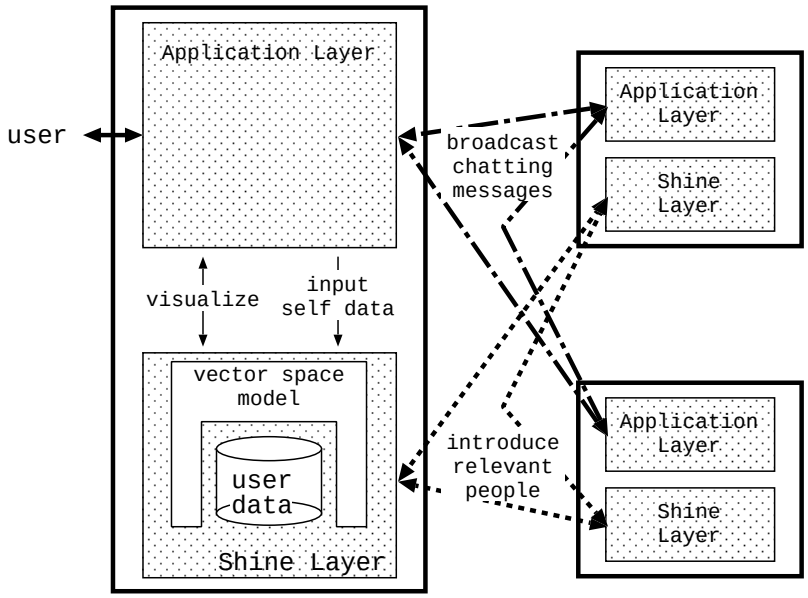

Fig. 5. The Community Organizer based on Shine

The Community Organizer uses a vector space model of an information retrieval technique as the user model. Because Shine provides a library for this model, the program of the Community Organizer can make use of prepared methods such as the cosine measure [11]. Furthermore, the user interface of the Community Organizer can choose its visualization mechanism from Shine's vector space visualization library, which includes not only the spring model [13] but also other mechanisms such as the dual-scaling method [5].

With the Community Organizer as a client/server system, those users represented on the display as icons are only in groups whose members are registered on the server. Unfortunately, this situation lacks flexibility and makes the society of users closed. Using Shine, however, the Community Organizer can become a distributed multi-agent system because the Shine layer of each Community Organizer agent is responsible for communications with other agents by mediation and/or/the introduction of other members.

The architecture of the Community Organizer based on Shine is shown in Fig. 5.

5 Conclusions

In this paper, we proposed a platform named Shine which can provide common functions for various socialware applications and discussed how to achieve the platform.

Much work remains to be done, however. At first, we are going to study the architecture of Shine more precisely, and implement it. We plan to do this work in parallel with the development of socialware applications such as the Community Organizer [2, 13] and Gleams of people [10]. There are also various other matters to deal with, for example, privacy issues.

References

[1] Kunihiko Fujita and Susumu Kunifuji. A realization of a reflection of personal information on distributed brainstorming environment. In Takashi Masuda, Yoshifumi Masunaga, and Michiharu Tsukamoto, editors, *Proceedings of the International Conference on Worldwide Computing and Its Applications '97*, volume 1274 of *Lecture Notes in Computer Science*, pages 166–181. Springer-Verlag, 1997.

[2] Fumio Hattori, Takeshi Ohguro, Makoto Yokoo, Shigeo Matsubara, and Sen Yoshida. Socialware: Multiagent systems for supporting network communities. *Communications of the ACM*, 42(3):55–61, 1999.

[3] Toru Ishida, editor. *Community Computing — Collaboration over Global Information Networks.* John Wiley & Sons, 1998.

[4] Kazuhiro Kuwabara, Toru Ishida, and Nobuyasu Osato. Agentalk: Describing multiagent coordination protocols with inheritance. In *Proceedings of the 7th International Conference on Tools with Artificial Intelligence*, pages 460–465, 1995.

[5] Kenji Mase, Yasuyuki Sumi, and Kazushi Nishimoto. Informal conversation environment for collaborative concept formation. In Ishida [3], chapter 6, pages 165–205.

[6] Shigeo Matsubara, Takeshi Ohguro, and Fumio Hattori. CommunityBoard: Social meeting system able to visualize the structure of discussions. In *Proceedings of the Second International Conference on Knowledge-based Intelligent Electronic Systems (KES'98)*, pages 423–428. IEEE, 1998.

[7] Shigeo Matsubara, Takeshi Ohguro, and Fumio Hattori. CommunityBoard 2: Mediating between speakers and an audience in computer network discussions. In Oren Etzioni, Jörg P. Müller, and Jeffrey M. Bradshaw, editors, *Proceedings of the Third Annual Conference on Autonomous Agents*, pages 370–371. ACM Press, 1999.

[8] Yoshiyasu Nishibe, Ichiro Morihara, Fumio Hattori, Toshikazu Nishimura, Hirofumi Yamaki, Toru Ishida, Harumi Maeda, and Toyoaki Nishida. Mobile digital assistants for international conferences. In Ishida [3], pages 245–284.

[9] Toyoaki Nishida and Hideaki Takeda. Towards the knowledgeable community. In Kazuhiro Fuchi and Toshio Yokoi, editors, *Knowledge Building and Knowledge Sharing*, pages 155–164. Ohmsha and IOS Press, 1994.

[10] Takeshi Ohguro, Sen Yoshida, and Kazuhiro Kuwabara. Gleams of people: Monitoring the presence of people with multi-agent architecture. In *Proceedings of the Second Pacific Rim International Workshop on Multi-Agents.* In this volume.

[11] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[12] Upendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *CHI '95 Proceedings: Conference on Human Factors in Computing Systems: Mosaic of Creativity*, pages 210–217. ACM SIGCHI, 1995.

[13] Sen Yoshida, Koji Kamei, Makoto Yokoo, Takeshi Ohguro, Kaname Funakoshi, and Fumio Hattori. Visualizing potential communities: A multiagent approach. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS-98)*, pages 477–478. IEEE Computer Society, 1998.