USER AND APPLICATION INTERFACE FOR A DISTRIBUTED E-LEARNING SYSTEM

Takao Kawamura, Kazuo Kuramochi
Faculty of Engineering, Tottori University, 4-101 Koyama-Minami, Tottori, Japan
kawamura@ike.tottori-u.ac.jp, s032021@ike.tottori-u.ac.jp

Junya Kishida, Shinichi Motomura
Graduate School of Engineering, Tottori University, 4-101 Koyama-Minami, Tottori, Japan
s022022@ike.tottori-u.ac.jp, motomoura@tottori-u.ac.jp

Kazunori Sugahara
Faculty of Engineering, Tottori University, 4-101 Koyama-Minami, Tottori, Japan
sugahara@ike.tottori-u.ac.jp

Keywords: Distributed System, P2P, Mobile Agent, Web-based training, e-Learning.

Abstract: In this paper, the user and application interface of our previously proposed e-Learning system is presented to demonstrate how user interfaces for distributed systems should be composed. The proposed e-Learning system has two distinguishing features. Firstly, it is based on P2P architecture for scalability and robustness. Secondly, all exercises in the system are not only data but also agents so that they have functions, such as scoring user's answers, telling the correct answers, and showing some related information without human instruction. The interface provides three kinds of communication: communication between a user and an exercise agent, communication between an exercise agent and support applications, and communication between a user and other users. The interface consists of two parts: an interface agent and a plugin for Firefox web browser. An interface agent hides the existence of the P2P network and other agents from the Firefox plugin and users. The developed e-Learning system including the interface is examined by experiments in classroom.

1 INTRODUCTION

Nowadays, e-Learning systems are very popular everywhere, such as college education, corporate education, and community education. The term e-Learning covers a wide set of applications and processes, such as Web-based training (hereafter we abbreviate as WBT), computer-based training, virtual classrooms, and digital collaboration. We are concerned with asynchronous WBT that allows the learner to complete the WBT on his own time and schedule, without live interaction with the instructor.

Although a large number of studies have been made on asynchronous WBT (Nishita et al., 2002; Homma and Aoki, 2003; Helic et al., 2005), all of them are based on the client/server model. The client/server systems generally lack scalability and robustness. In the recent years, P2P research has grown exponentially. Although the current P2P systems are famous for file sharing, and the consequent legal problems, P2P systems are gradually proving to be a very promising area of research because they have potential for offering a decentralized, self-sustained, scalable, fault tolerant and symmetric network of computers providing an effective balancing of storage and bandwidth resources.

We have proposed and implemented a distributed e-Learning system based on P2P architecture (Kawamura and Sugahara, 2005; Kawamura et al., 2005a; Motomura et al., 2006c) using Maglog that is a Prolog-based framework for building mobile multi-agent systems we have also developed (Kawamura et al., 2005b; Motomura et al., 2006a; Motomura et al., 2006b). The proposed e-Learning system has two distinguishing features. Firstly, it is based on P2P architecture and every user's computer plays the role of a client and a server. Namely, while a user uses the proposed e-Learning system, his/her computer (hereafter we refer to such a computer as a node) is a part of the system. It receives some number of contents from another node when it joins the system and has responsibility to send appropriate contents to requesting nodes. Secondly, each exercise in the system is not only data but also an agent so that it has functions,

such as scoring user's answers, telling the correct answers, and showing some related information without human instruction.

In this paper, we present the user and application interface (hereafter we abbreviate as UAI) of the proposed e-Learning system to demonstrate how user interfaces for distributed systems should be composed.

This paper is organized in 6 sections. The proposed e-Learning system is described in Section 2. We describe three functions of the UAI in Section 3 and an implementation of the UAI in Section 4, respectively. In Section 5 experimental results are presented. Finally, some concluding remarks are drawn in Section 6.

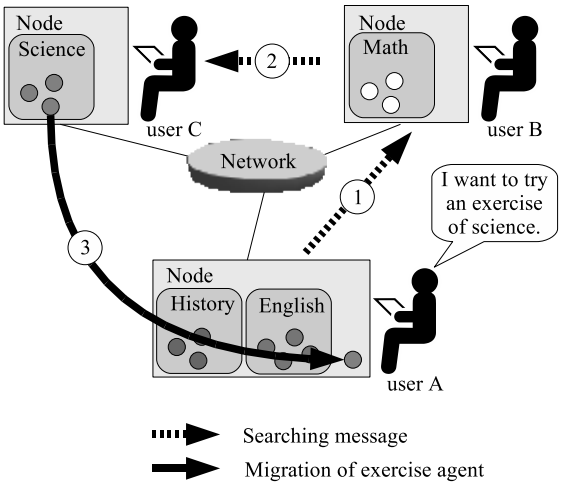


Figure 1: Proposed e-Learning system.

2 PROPOSED E-LEARNING SYSTEM

2.1 Overview

As mentioned in the previous section, we focus on asynchronous WBT, that is to say, a user can connect to the proposed e-Learning system anytime and anywhere he/she wants. Once connection is established, the user can obtain exercises one after another through specifying categories of the required exercises. User's answers for each exercise are scored as correct or incorrect right away. Related information may be provided for each answer, which can be viewed when the correct answer is shown.

While a user uses the proposed e-Learning system, his/her computer is a part of the system. Namely, it receives some number of categories and exercises in them from another node when it joins the system and has responsibility to send appropriate exercises to requesting nodes.

The important point to note is that the categories a node has are independent of the categories in which the node's user are interested as shown in Figure 1. Figure 1 illustrates that user A's request is forwarded at first to the neighbor node, next forwarded to the node which has the requested category.

2.2 P2p Network

All exercises in the proposed system are classified into categories, such as "English/Grammar", "Math/Statistics", and "History/Rome", etc.

When the proposed system bootstraps, one initial node has all categories in the system. When another node joins the system, it is received certain number of categories from the initial node. The categories are distributed among all nodes in the system according as nodes join the system or leave the system.

We would like to emphasize that in existing P2P-based file sharing systems, such as Napster (Napster, 2003), Gnutella (Gnutella, 2001), and Freenet (Clarke et al., 1999) each shared file is owned by a particular node. Accordingly, files are originally distributed among all nodes. On the other hand, the categories in the proposed system are originally concentrated. Consequently, when a new node joins the system, not only location information of a category but the category itself must be handed to the new node. Considering that, the P2P network of the proposed system can be constructed as a CAN (Ratnasamy et al., 2001).

The CAN has a virtual coordinate space that is used to store $(key, value)$ pairs. To store a pair (K_1, V_1), key K_1 is deterministically mapped onto a point P in the coordinate space using a uniform hash function. The corresponding $(key, value)$ pair is then stored at the node that owns the zone within which the point P lies. In the proposed system, we let each category be a key and let a set of exercises belonging to the category be the corresponding value.

Our P2P network is constructed with 2-dimensional coordinate space [0,1] $\times$ [0,1] to store exercise categories, as shown in Figure 2. The figure shows the situation that node C has just joined the system as the third node. Before node C joins, node A and node B shared the whole coordinate space half and half. At that moment, node A managed "Math/Geometry" ,"Math/Statistics", and "History/Rome" categories and node B managed "English/Grammar", "English/Reader and "History/Japan" categories, respectively. When node C

joins the system, it is mapped on a certain coordinate space according to a random number and takes on corresponding categories from another node. For example, in the case of Figure 2, node C takes on the "History/Japan" category from node B and exercises of the category move to node C.

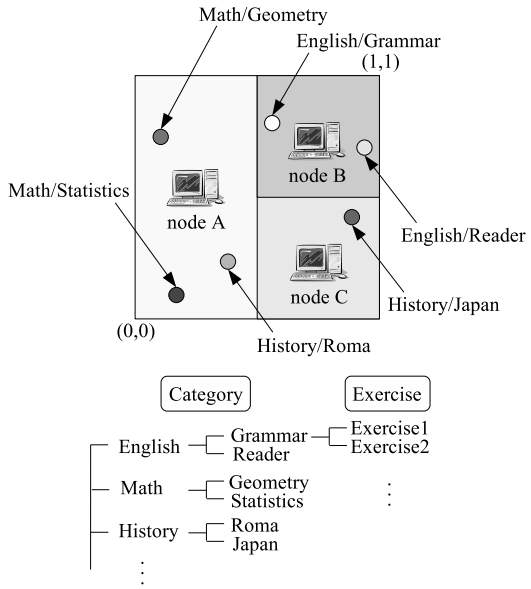


Figure 2: P2P network.

2.3 Mobile Agents

Generally, in addition to service to show an exercise, a WBT server provides services to score user's answers, to tell the correct answers, and show to some related information about the exercise. Therefore, for the proposed system that can be considered a distributed WBT system, it is not enough that only exercises are distributed among all nodes. Functions to provide the above services also must be distributed among all nodes. We adopt mobile agent technology to achieve this goal. Namely, an exercise is not only data but also an agent so that it has functions, such as scoring user's answers, telling the correct answers, and showing some related information about the exercise.

In addition, mobile agent technology is applied to realize the migration of categories, that is, each category is also an agent in the proposed system.

3 USER AND APPLICATION INTERFACE

There are three types of possible communications between for using the distributed e-Learning system. In this section, three functions of the UAI to realize these communications are described.

3.1 Function for Communication Between a User and an Exercise Agent

Although each exercise agent has a question and functions, such as scoring user's answers, telling the correct answers, and showing some related information about the exercise, it cannot communicate with users directly. All agents in the proposed system communicate with other agents or applications through 'field's provided by Maglog framework. A field is a kind of preemptive queue. When agent-A intends to send a message to agent-B, agent-A queues the message onto the field owned by agent-B. Since agent-B executes a message dispatch loop like all other agents, it reads the message written by agent-A sometime. The UAI therefore has a function to convert user's request into messages and to write them onto the fields of exercise agents, and vice versa.

3.2 Function for Communication Between an Exercise Agent and Applications

An exercise agent may need some support applications to score user's answer. For example, when the correct answer is "x^2+x+y", other forms of answers, such as "$y+x+x^2$" or "$x+x^2+y$" answered in a text box should be scored as correct. However, it is impossible for exercise agents to score all possible forms of the correct answer as correct without a computer algebra system. An exercise requiring C programming is another example. An exercise agent must compile and execute a user's answer to compare the output of it with the contents of the correct answer. Since exercise agents migrate between nodes, it is an infeasible idea that each exercise agent includes a computer algebra system or a C compiler. Instead, in the proposed system, exercise agents utilize support applications, such as a computer algebra system or a C compiler prepared on each node. The UAI therefore has a function for communication between exercise agents and support applications.

3.3 Function for Communication Between a User and Other Users

Users may need to be advised to understand correct answers. Existing e-Learning systems often provide BBS (Bulletin Board System) to deal such situation. However, we don't adopt BBS because it is not suitable for distributed systems. In addition, questions on BBS are neglectable. Instead, we let the UAI have a function for users to help each other by means of chat.

4 IMPLEMENTATION OF THE USER AND APPLICATION INTERFACE

We have implemented the UAI consisting of two parts: an interface agent and a plugin for Firefox web browser (Mozilla Corporation, 2005). Users try exercises by using Firefox with the plugin that does not know the existence of the P2P network and any agents except an interface agent. In this section, we describe the architecture of a node at first. Next we describe the two parts of the UAI, respectively.

4.1 Node Architecture

As shown in Figure 3, the following agents exist on each node. An agent server in this figure is a runtime environment for agents.

Node Agent There is one node agent on each node. It manages the zone information of a CAN and forwards messages to the category agents in the node.

Exercise Agent Each exercise agent has a question and functions to score user's answers, tell the correct answers, and show some related information about the exercise. These data are formatted in HTML.

Category Agent Each category agent stands for a unit of a particular subject. It manages exercise agents in itself and sends them to the requesting node.

Interface Agent There is one interface agent on each node. It is an agent part of the UAI.

User Agent There is one user agent for each user. A user agent manages user information that include login name, password, IP address of the user's computer, online/offline status, and study log.

Group Agent A group agent exists for grouping user agents as a category agent exists for grouping exercise agents. Currently, users are grouped by the first letter of their login name, so that there are 26 group agents corresponding to one letter in the English alphabet.

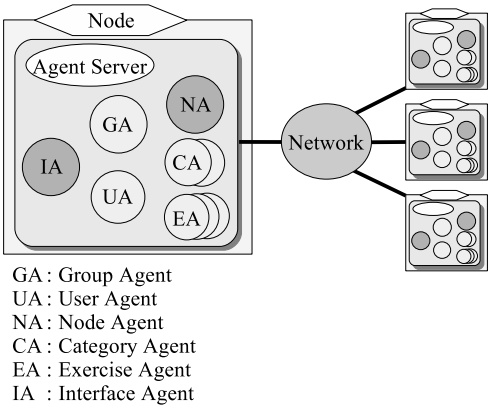


Figure 3: Agents in a node.

4.2 Firefox Plugin

Users try to answer exercises by using Firefox with the plugin. Figure 4 is a screenshot of the main user interface window generated by the plugin. Categories are classified into a hierarchical tree structure, as shown in the category pane of Figure 4. By clicking the left button of a mouse on a category, Users can select the category. An exercise belonging to the category is obtained from some node and is shown in the exercise pane by clicking the exercise button on the menu bar. Users can require to score their answer anytime by clicking the score button. Correct answers and related information about the exercise are shown by clicking the answer button. In addition to them, users can request some advise about the exercise by clicking the chat button.

4.3 Interface Agent

When a user tries an exercise, in addition to the agent server, two types of processes run as shown in Figure 5. One type is for Firefox with the plugin for the UAI. The other type is for support applications. They help exercise agents to score students' answers. There is a wrapping Java class for each support application to be utilized by an interface agent.

An interface agent enables three kinds of communication as follows:

1. Communication between a user and an exercise agent.

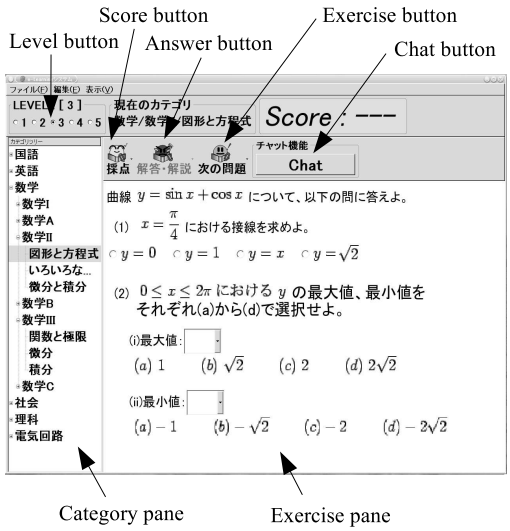
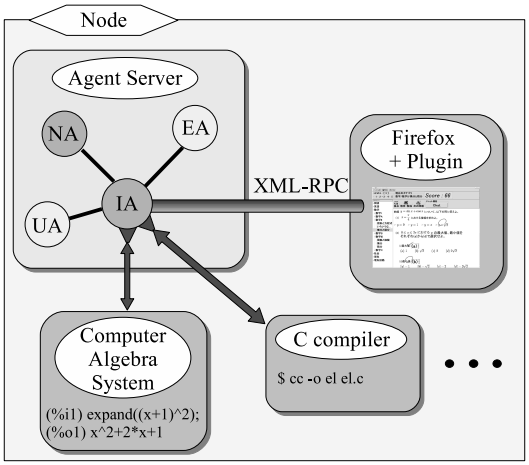


Figure 4: Main user interface window of the proposed e-Learning system.



UA: User Agent
NA: Node Agent
EA : Exercise Agent
IA : Interface Agent

Figure 5: User and application interface and other components in a user studying node.

The Firefox plugin converts user's request into messages and writes them onto a filed of an interface agent via XML-RPC (Winer, 1998) using jsolait (Kollhof, 2004).

After that, the plugin waits until the result of a request is written on the same field by the interface agent. An interface agent forwards the received messages to an appropriate agent: A request for obtaining an exercise is forwarded to the node agent on the user's computer, and requests for being scored and for obtaining the correct answer and related information are forwarded to the exercise agent.

2. Communication between an exercise agent and support applications.

   All agents on Maglog, which are written in a Prolog-like language, are translated into Java source codes by our Maglog translator, and then compiled into Java classes by the Java compiler to being executed. An agent runs as a thread in an agent server which is also implemented using Java. Thereby agents can utilize any Java object, i.e., an interface agent, which works as a proxy of an exercise agent, can utilize a support application for which a corresponding wrapping Java class is prepared. A wrapping Java class utilizes the standard Java class `Process` to invocate a support application and to communicate with it using standard input/output.

3. Communication between a user and other users.

   When a user wants to be advised about an exercise, the following steps are performed.

   (a) The interface agent asks the exercise agent the potential advisers.

   (b) The exercise agent tells a list of login names of the users who gained a good score on itself to the interface agent.

   (c) The interface agent asks the appropriate group agent whether or not each user in the list is logging-in the system. If a user is logging-in, the IP address of the user's computer is also asked.

   (d) The interface agent asks each user who is logging-in the system and gained a good score on the exercise in random order until one user offers his/her service.

   (e) The interface agent establishes a chat channel between the two users.

5 EXPERIMENTS

The developed e-Learning system including the user and application interface was examined by experiments in classroom. For the experiments, 60 PCs with Intel Pentium4 3.0GHz processor and 1GB of RAM are connected via 1000Base-T network. All the PCs were running on GNU/Linux (kernel version is 2.6.0) operating system. The version of Java language runtime environment was 1.4.2.

Figures 6 and 7 show, respectively, the result of scoring, and the correct answer and related information of the exercise.

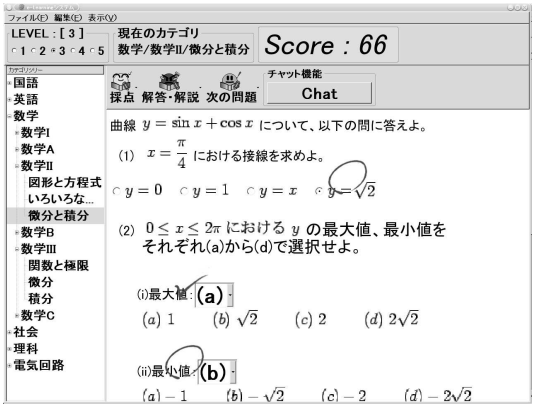


Figure 6: User's answers are scored as correct or incorrect by clicking the score button.

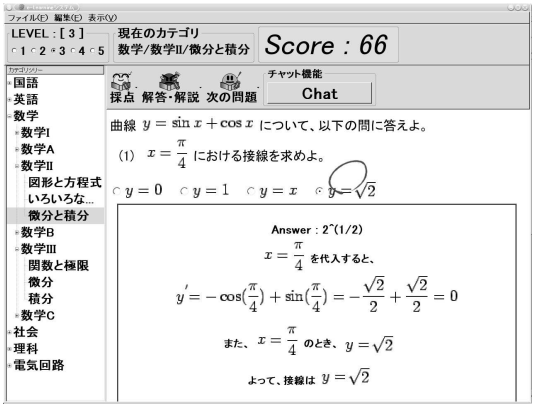


Figure 7: Correct answers and related information are shown by clicking the answer button.

Figure 8 shows an example result of scoring using a computer algebra system, Maxima (Maxima Project, 2000). In this example, the exercise agent requests Maxima through an interface agent to compare its correct answer with a user's answer. Maxima returns a subtraction of two expressions. The exercise agent scores the user's answer as correct or incorrect whether the result of the subtraction equals 0 or not. Consequently, as shown in Figure 8, both "$\cos x - \sin x$" and "$-\sin x + \cos x$" are scored as correct.

As shown in Figure 9, while a chat session is in progress, same windows are opened on the both two users' screen.

6 CONCLUSION

Since existing asynchronous WBT systems are based on the client/server model, they have problems of

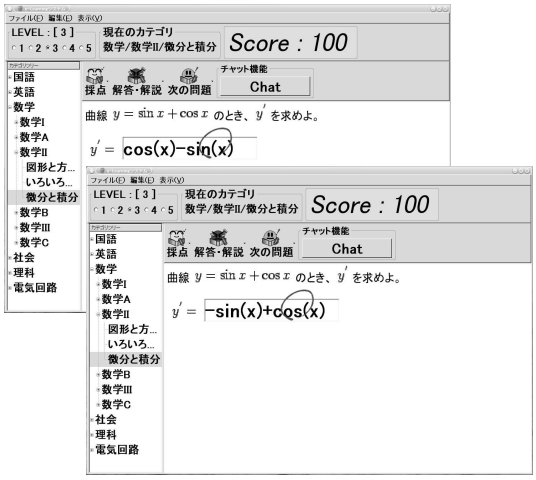


Figure 8: Example of scoring using a computer algebra system.

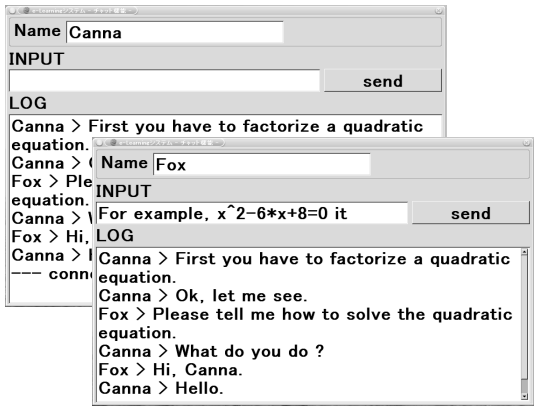


Figure 9: Chat windows.

scalability and robustness. The proposed e-Learning system solves these problems in decentralized manner through both P2P technology and mobile agent technology.

In this paper, the user and application interface of the proposed e-Learning system are described to demonstrate how user interfaces for distributed systems should be composed. It realizes three kinds of communication: communication between a user and an exercise agent, communication between an exercise agent and support applications, and communication between a user and other users. The user and application interface consists of two parts: an interface agent and a plugin for Firefox web browser. An interface agent hides the existence of the P2P network and other agents from the Firefox plugin and users.

The developed e-Learning system including the user and application interface was examined by experiments. More practical experiments are now planned

at our university to confirm the effectiveness of the proposed e-Learning system.

REFERENCES

Clarke, I., Sandberg, O., Wiley, B., and Hong, T. W. (1999). Freenet: A distributed anonymous information storage and retrieval system. http://freenetproject.org/freenet.pdf.

Gnutella (2001). Gnutella.com. http://welcome.to/gnutella/.

Helic, D., Krottmaier, H., Maurer, H., and Scerbakov, N. (2005). Enabling project-based learning in wbt systems. *International Journal on E-Learning (IJEL)*, 4(4):445–461.

Homma, H. and Aoki, Y. (2003). Creation of wbt server on digital signal processing. In *Proceedings of 4th International Conference on Information Technology Based Higher Education and Training*. Marrakech, Morocco.

Kawamura, T., Motomura, S., Nakatani, R., and Sugahara, K. (2005a). Multi agent-based approach for asynchronous web-based training system. In *Proceedings of International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pages 537–542. Waltham, Massachusetts, USA.

Kawamura, T., Motomura, S., and Sugahara, K. (2005b). Implementation of a logic-based multi agent framework on java environment. In *Proceedings of International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, pages 486–491. Waltham, Massachusetts, USA.

Kawamura, T. and Sugahara, K. (2005). A mobile agent-based p2p e-learning system. *IPSJ Journal*, 46(1):222–225.

Kollhof, J.-K. (2004). jsolait –trac–. http://jsolait.net/.

Maxima Project (2000). Maxima - a gpl cas based on doemacsyma. http://maxima.sourceforge.net/.

Motomura, S., Kawamura, T., and Sugahara, K. (2006a). A logic-based mobile agent framework for web applications. In *Proceedings of the 2nd International Conference on Web Information Systems and Technologies*, pages 121–126. Setubal, Portugal.

Motomura, S., Kawamura, T., and Sugahara, K. (2006b). Logic-based mobile agent framework with a concept of “field”. *IPSJ Journal*, 47(4):1230–1238.

Motomura, S., Nakatani, R., Kawamura, T., and Sugahara, K. (2006c). Distributed e-learning system using p2p technology. In *Proceedings of the 2nd International Conference on Web Information Systems and Technologies*, pages 250–255. Setubal, Portugal.

Mozilla Corporation (2005). Rediscover the web. http://www.mozilla.com/firefox/.

Napster (2003). napster. http://www.napster.com/.

Nishita, T., Kondo, K., Ohno, Y., Takai, Y., Miura, K. T., Dobashi, Y., Ishiuchi, S., Takahashi, T., Kimura, A., and Miyai, A. (2002). Development of a web based training system and courseware for advanced computer graphics courses enhanced by interactive java applets. In *Proceedings of International Conference on Geometry and Graphics*, volume 2, pages 123–128.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. (2001). A scalable content-addressable network. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 161–172. ACM Press.

Winer, D. (1998). Xml-rpc specification. http://www.xmlrpc.com/spec.