Ad-hoc Endpoint Pairing based on Simultaneous Clicking for Ubiquitous Communications

Takashi Oshiba, Kazuaki Nakajima and Hideaki Nebayashi
Service Platforms Research Laboratories, NEC Corporation, Japan
{oshiba@cp, nakajima@ah, nebayashi@ap}.jp.nec.com

1. INTRODUCTION

The spread of ubiquitous communications is dramatically changing the way of daily work. Office workers can now deepen mutual understanding by using a telephone and a PC concurrently [1]. In this paper, we present an ad-hoc endpoint pairing method that enables a user to easily establish a pair of endpoints, such as an IP-hard phone (fixed or mobile IP-phone) and a PC-based soft phone, simply by simultaneously clicking a GUI button on the display screen of each endpoint. Using the endpoint pairing information, the user can easily launch a PC-to-PC web/data conferencing application during a phone-to-phone audio conversation. It can thus promote deeper mutual understanding between users.

2. CONVENTIONAL APPROACHES

In conventional approaches, there are two types of endpoint pairing; static one and dynamic one.

2.1 Static Endpoint Pairing

The example shown in Figure 1 illustrates the conventional static endpoint pairing approach.

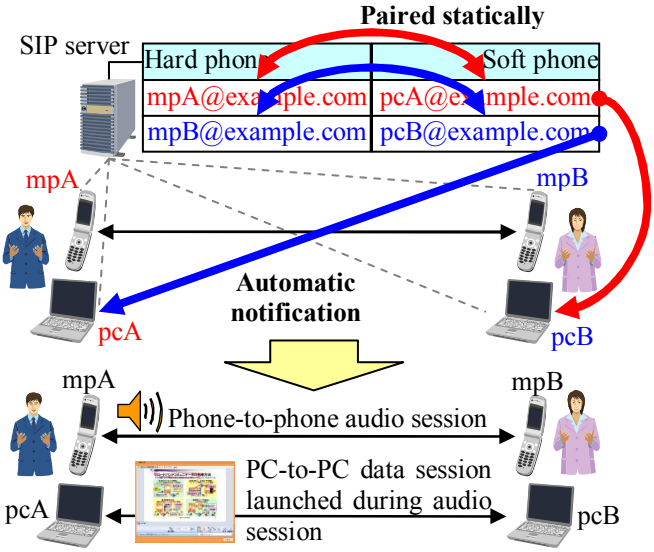

Figure 1: Static endpoint pairing and automatic SIP address notification.

When A and B begin to talk using IP-hard phones, the soft phone of user A's (B's) PC automatically obtains the SIP address of user B's (A's) PC from the SIP server. This automatic SIP address notification can be realized by associating (i.e., pairing) the SIP addresses of the IP-hard and soft phones on the server. Note that a system administrator has to statically pair up the SIP addresses of the two endpoints beforehand. Using the received SIP address of B's PC, user A can easily launch a web/data conferencing application to share a document, image, spreadsheet, etc. Users can thereby add a PC-to-PC data session to a phone-to-phone audio session, enabling them to visually indicate what they want to explain. This facilitates their discussion, so they can achieve a mutual understanding more rapidly.

2.2 Dynamic Endpoint Pairing

If a user could establish a pair of endpoints dynamically, the dynamic endpoint pairing would enable a user to use both endpoints concurrently in various business situations. Let's consider potential scenarios.

(a) **User A uses own hard phone and shared-use PC**: User A is talking to user B using his or her VoWLAN (Voice over Wireless LAN) mobile phone at a remote office. User A's PC, which was previously paired with the mobile phone, is at user A's office. A shared-use PC is available at the remote site. With dynamic pairing, user A can pair up the mobile phone with the shared-use PC dynamically and launch a PC-to-PC data session immediately.

(b) **User A uses own hard phone, own laptop PC, and own desktop PC**: User A is talking to user B using his or her VoWLAN mobile phone at user A's office. His or her laptop PC has already been paired with the mobile phone, but he or she wants to share a document that is only on his or her desktop PC. With dynamic pairing, user A can pair up the mobile phone with the desktop PC dynamically and launch a PC-to-PC data session immediately.

3. PROBLEMS WITH CONVENTIONAL APPROACHES

3.1 Troublesome

In conventional static endpoint pairing approaches like that illustrated in Figure 1, the concurrent use of both endpoints is troublesome in these scenarios. In scenario (a), the shared-use PC will not be automatically notified of the SIP address because the endpoints pair is statically defined beforehand. User A thus has to perform a tedious task in order to launch a PC-to-PC data session. He or she has to input his or her ID/password into the soft phone on the shared-use PC manually in order to login and then input the SIP address of user B's soft phone manually in order to call. If user A does not remember the SIP address, he or she has to search through the organizational tree of the directory system to find the SIP address or ask user B for the SIP address through the audio session. Similarly, scenarios (b) also entail tedious tasks.

3.2 Uncertainty

If multiple users execute conventional dynamic endpoint pairings simultaneously, mismatched pairings with other user's endpoints (*mismatched pairings*) can occur. For example, in the industry standard protocol for dynamic

wireless device pairing [2], a Wi-Fi access points (AP) and a laptop PC can be paired dynamically. In the pairing negotiation phase, the laptop PC sends a pairing request to the Wi-Fi AP, which accepts the request in a FIFO manner. This FIFO policy leads to *mismatched pairings*. Consider the case in which user A has WA1 and PC1 and user B has WA2 and PC2, and user A is near to user B. If they begin the pairing negotiation simultaneously, PC1 sends a pairing request to WA1 and PC2 sends a pairing request to WA2, but WA1 also receives the request from PC2 and WA2 also receives the one from PC1. If the reception order on WA1 is PC2 and then PC2, and the one on WA2 is PC1 and then PC2 a *mismatched pairing* (a WA1-PC2 pair or a WA2-PC1 pair) could occur, even if the desired pairs are WA1-PC1 and WA2-PC2.

4. REQUIREMENTS

Given the problems with the conventional approaches, we identified two requirements for a dynamic and flexible endpoint pairing method.

(1) **Simple dynamic endpoint pairing and PC-to-PC data session launching**: A user should be able to choose and establish an endpoints pair easily without having to manually input an ID/password or SIP address and be able to communicate using both a phone-to-phone audio session and a PC-to-PC data session in an ad-hoc manner.

(2) **Positive dynamic endpoint pairing**: *Mismatched pairings* should not be possible. The created endpoints pair should meet the user's expectation even if there are concurrent pairing requests from other users.

5. AD-HOC ENDPOINT PAIRING

We have developed an ad-hoc endpoint pairing method as a new dynamic endpoint pairing that satisfies the two requirements described above. A user can establish an endpoints pair dynamically, simply by simultaneously clicking a GUI button on the display screen of both endpoints. The user can avoid *mismatched pairings* in an intuitive manner.

5.1 Overview

In our method, the core component is an *endpoint pairing server* (EPS), which manages the endpoint pairings. The EPS determines whether two endpoints can be paired or not on the basis of the time lag between the user's two clicks. Pair establishment is done in two phases: temporal pair creation and formal pair creation. The use of these two phases prevents *mismatched pairings* even if the EPS receives pairing requests from other users at the same time. In the first phase, the EPS assigns a different still image to each temporal pair and sends the image to the two corresponding endpoints. Users confirm whether the temporal pair matches the user's expected pair or not by comparing the received images on two endpoints. If it does, the EPS promotes the temporal pair to a formal pair.

If this promotion is executed during the user's phone-to-phone conversation with the other party, and if the other party has already finished his or her endpoint pairing, the EPS immediately sends SIP address notifications (mentioned in Section 2.1) to the PCs of the two users. Note that the EPS not only accepts pairing requests during a phone-to-phone conversation but also accepts them when a user is not talking. Users can establish their endpoints pairs beforehand, so when they begin a phone-to-phone conversation, the automatic SIP address notifications are sent to their PCs immediately.

5.2 System Architecture and Method Flowchart

Figure 2 shows the architecture of a system using our method.

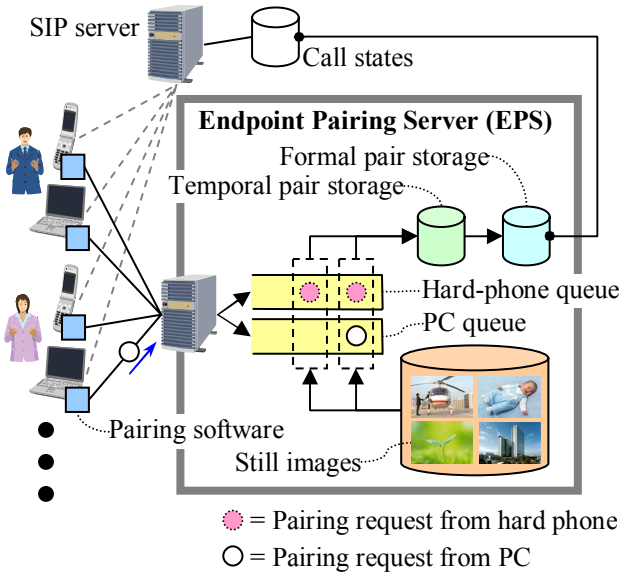

Figure 2: Architecture of system using our method.

The EPS has two queues, a hard-phone queue for pairing requests from hard phones and a PC queue for pairing requests from PCs. It has storage for the temporal pairs and formal pairs. It also has a number of still images which is used to be assigned to each temporal pair. The images enable a user to prevent *mismatched pairings* by using an image comparison. Although the EPS is logically separated from the SIP server, it can reside in the same box. The SIP server tracks the call state for each endpoint; the call states indicate the endpoints active during a conversation. Each endpoint has pairing software to communicate with the EPS.

Figure 3 shows the steps in our method. In steps S1–S2, the temporal pair is created. In steps S3–S5, the user confirms that the pairing is correct and the formal pairing is created. In step S6, the endpoint pairing is completed.

5.3 Operation

The user launches pairing software on the hard phone and on the PC to be paired, which brings up a user interface, for example, a GUI button, on each for pairing. Note that, even if the endpoint is a shared-use endpoint, the user does not have to input his or her ID/password into the pairing software. When the user simultaneously clicks the GUI buttons on the user interfaces using both hands (Figure 4), pairing requests are sent from the pairing software of each endpoint to the EPS (S1). Each one includes the type of endpoint, i.e., hard phone or PC, and its IP address.

The EPS determines whether the two endpoints can be paired or not (S2) by comparing the time lag between the reception of the two requests and a server-side predefined value, the *pairing timeout*. When a pairing request first arrives, the EPS creates a pairing slot, and subsequent pairing requests are queued in either the hard phone or PC queue until the *pairing timeout* is reached. In the example shown in Figure 5, there are three pairing slots.

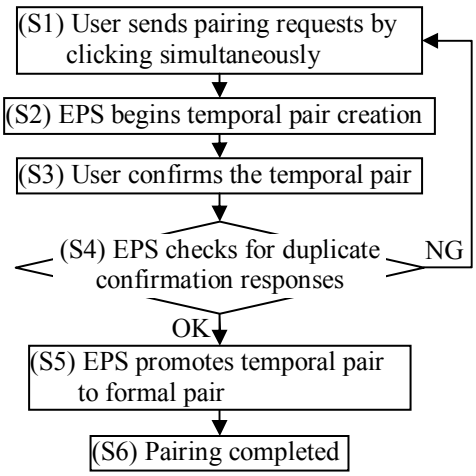

Figure 3: Flowchart of our method.

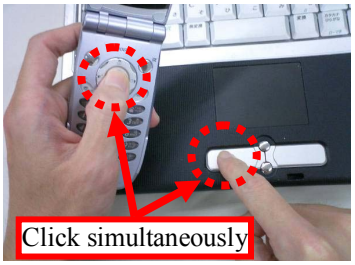

Figure 4: Simultaneous clicking of hard phone and PC.

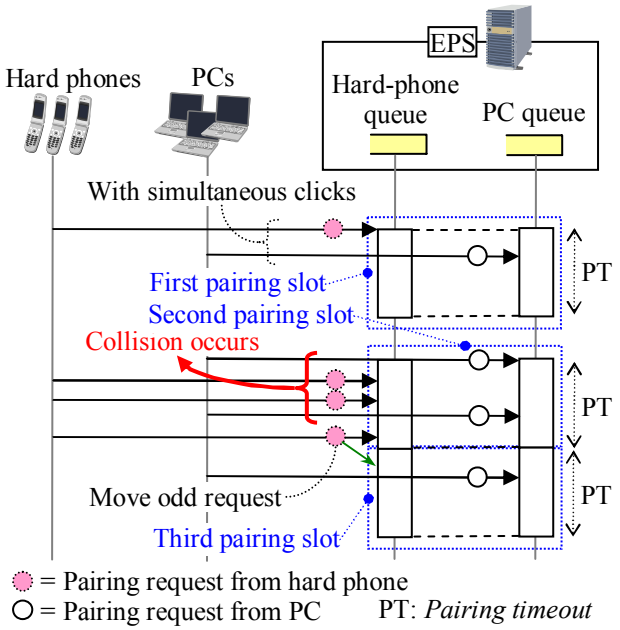

Figure 5: Pairing requests with simultaneous clicking.

The first pairing slot receives a pairing request from a hard phone and one from a PC. The second pairing slot receives three from hard phones and two from PCs. When the second paring slot times out, a third pairing slot is immediately created. The last request from a hard phone in the second slot is moved to the third slot because there is no request from a PC with which to pair it. As a result, the third pairing slot ends up with one request from a hard phone and one from a PC. After each timeout of a pairing slot, the EPS selects two pairing requests, one from each queue, and makes them a temporal pair. The two requests must have the same deepness in their respective queue, as illustrated in Figure 2. The EPS continues this selection process until the two queues are empty.

5.4 Positive Intuitive Protection against Mismatched Pairings

With every timeout of a pairing slot, the EPS assigns a different still image to each temporal pair, and stores the temporal pair in the temporal pair storage area (Figure 2). The EPS sends the image to the two corresponding endpoints. Note that a collision with the two users occurs in the second slot in Figure 5, and the collision may cause *mismatched pairings*. If a collision occurs and multiple temporal pairs are created, the EPS additionally sends the images which are assigned with other temporal pairs to the PCs; the hard phones receive only a single image (Figure 6).

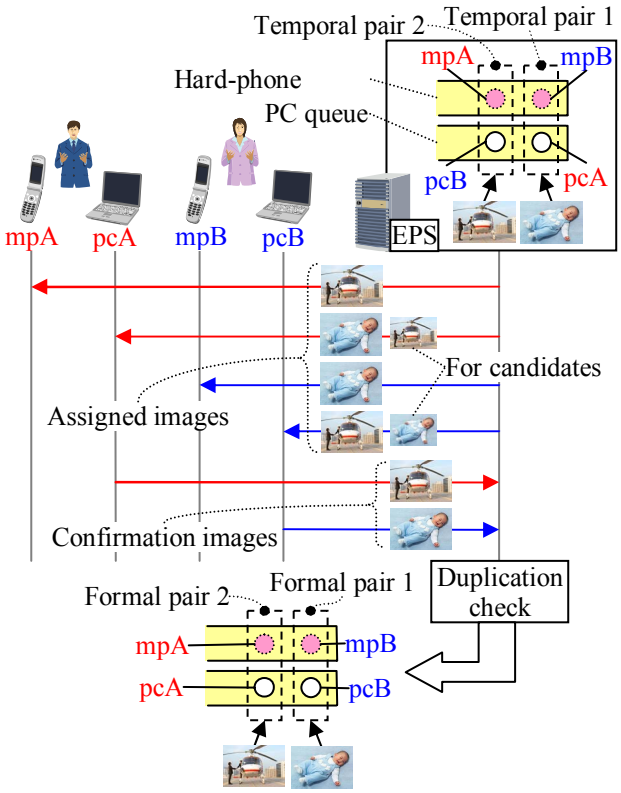

Figure 6: Image assignment and confirmation.

Each endpoint draws the image it receives on its display screen. If a collision occurs, the image drawn on the hard phone may not be the one drawn on the PC (Figure 7).

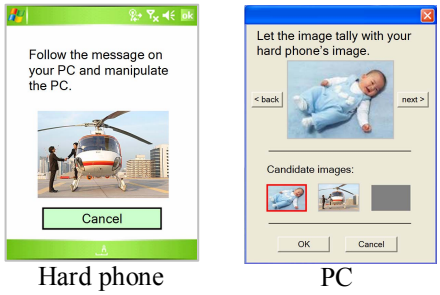

Figure 7: Positive intuitive protection against *mismatched pairings* by using image comparison.

If they differ, the temporal pair does not match the expected pair. The user can prevent their promotion to a formal pair by changing the image on the PC (in Figure 7, from the baby to the helicopter) and clicking OK (S3). The PC transmits this

"confirmation image" (the helicopter) to the EPS. If the images are the same (both are the helicopter), the temporal pair matches the expected pair. In this case, the user simply clicks OK on the PC screen. This image-comparison procedure makes the confirmation process intuitive, simple, and quick because people have highly developed image recognition ability.

The EPS uses another server-side predefined value, *confirmation timeout*. After the EPS sends the assigned images to the endpoints, it accepts responses from users with confirmation images until the *confirmation timeout* is reached. If the EPS has received all the responses from the users within the *confirmation timeout*, the EPS checks for duplicate confirmation images (S4). This duplication check prevents a malicious user from hijacking another user's endpoint by sending the same confirmation image as the other user. After the duplication check, the EPS exchanges the PC part of each temporal pair as needed in accordance with the confirmation images and promotes the temporal pairs to formal pairs (S5). This concludes the endpoint pairing process (S6). The EPS then sends the automatic SIP address notifications.

In short, a user can create an endpoints pair easily and dynamically without having to manually input his or her ID/password or SIP address of the other user in order to begin a PC-to-PC data session. Moreover, *mismatched pairings* are prevented. Our method thus satisfies all the requirements mentioned in Section 4.

6. EVALUATION

We evaluate the practicability of ad-hoc endpoint pairing method in office use by analyzing the relationship among the probability of collision occurrence, the *pairing timeout* and the number of endpoints. We assume the collision probability should be less than 2.0% in order to ensure the practicability in office use. We confirm that about 1,720 endpoints can be accommodated where the *pairing timeout* is 1.0 second. Thus, our method can work well in practical office use.

In a system using our method, the collision probability is an important performance indicator because a collision can cause a different image representation between the hard phone and PC, as shown in Figure 7. If this happens too often, users will become annoyed, making our method less advantageous.

In our method, there are two types of endpoint pairing, as mentioned in Section 5.1: the pairing during a telephone conversation and the pairing when a user is not in a conversation. We assume that collisions do not occur for the former because the two users are in conversation and thus can coordinate the timing of their pairing requests. We thus limited our analysis to the latter.

The latter type of pairing can be adapted to the M/D/1(0) model in queuing theory. We assume that the arrival process of pairing requests is Poisson distribution. The service time is always *pairing timeout (PT)*, i.e. a degenerate distribution. A pairing slot corresponds to the single server of the M/D/1(0) model. When a pairing slot is started by a user's pairing request, if a pairing request is arrived from another user during the pairing slot is being active, then, a collision occurs. This corresponds that when the single server of the M/D/1(0) model is occupied i.e. in service, if another arrival occurs during the single server is in service, then, a blocking occurs. Thus, the collision probability corresponds to blocking probability B. In this analysis, we assume each endpoint sends one pairing request per 24 hours. Therefore, the traffic intensity $a = n \cdot PT/(24 \times 3600) = n \cdot PT/86400$ where n is the number of endpoints. We used the Erlang-B formula to calculate the probability.

$$B = \frac{\frac{a^1}{1!}}{\sum_{i=0}^{1} \frac{a^i}{i!}} = \frac{a}{1+a} = \frac{\frac{n \cdot PT}{86400}}{1+\frac{n \cdot PT}{86400}} = \frac{n \cdot PT}{86400 + n \cdot PT}.$$

Figure 8 shows collision probability B for various numbers of endpoints and a PT of 3.0, 2.0, or 1.0 seconds. If the PT is set to 1.0 second, our method can accommodate about 1,720 endpoints. Thus, we confirm our method can ensure the practicability in office use.

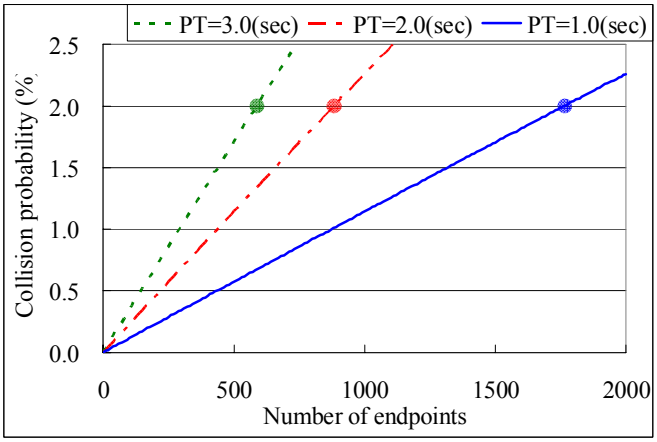

Figure 8: Collision probability for different numbers of endpoints.

7. CONCLUSION AND FUTURE WORK

We have developed a method for implementing ad-hoc endpoint pairing. It enables a user to create a pair of endpoints dynamically and easily, simply by simultaneously clicking a GUI button on the display screen of each endpoint. This method prevents mismatched pairings, i.e., pairings with another user's endpoints, in an intuitive manner.

We plan to adapt our developed method to multiple endpoints communications for security augmentation. Authorizing a PC-to-PC session using a phone-to-phone session ensures the security of multiple endpoints communications.

8. REFERENCES

[1] S. Shirasawa et al., "IP-Telephony Technology and Solution Families Covering One Box Startup to Carrier Grade Service," NEC Journal of Advanced Technology, Vol. 1, No. 2, pp. 133–142, 2004.

[2] C. Kuo et al., "Low-cost Manufacturing, Usability, and Security: An Analysis of Bluetooth Simple Pairing and Wi-Fi Protected Setup," Usable Security (USEC'07), pp. 325–340, 2007.