BlueRendezvous: Simple Pairing for Mobile Devices

Raman Sarin, Ken Hinckley
Microsoft Research, One Microsoft Way, Redmond, WA 98052
{ramans, kenh}@microsoft.com

ABSTRACT

A major user experience challenge for Smartphones and other mobile devices is to find ways that allow users to dynamically interconnect devices in a quick, simple, and intuitive manner. BlueRendezvous is a prototype application for the SmartPhone that greatly simplifies the Bluetooth discovery process. BlueRendezvous also makes it extremely simple to connect devices in different ways or for different purposes. For example, users can employ the same basic mechanism for mutual sharing of contact information, one-way sharing of images, or establishing a longer-term relationship between devices.

This document presents currently implemented scenarios as well as alternative embodiments, extensions, and design scenarios that we have invented.

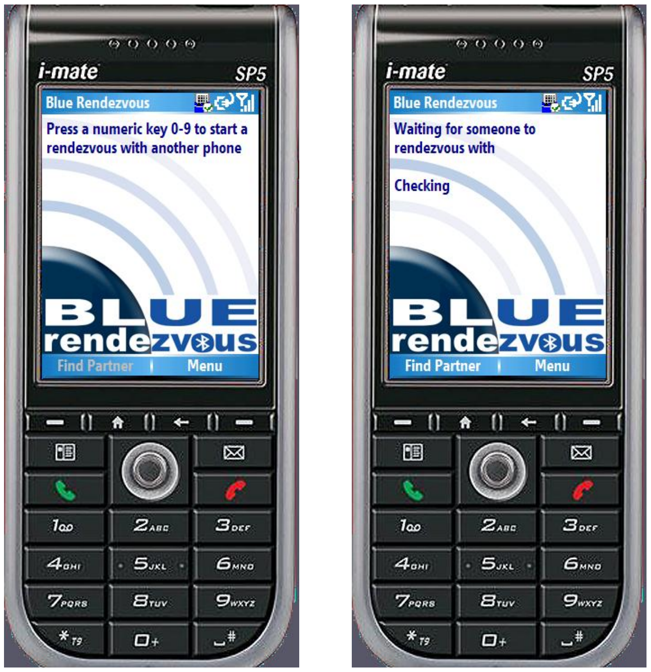
Keywords

Bluetooth, wireless networking, CSCW (computer supported collaborative work), multi-device interaction, networking, communication, file sharing, synchronous gestures, distributed interaction techniques.

INTRODUCTION

BlueRendezvous hides much of the complexity of forming a Bluetooth connection from users, so that users can focus on their face-to-face conversation in the real world, instead of dealing with the details of getting the technology to work. BlueRendezvous is based on *synchronous gestures* [3], which are actions that two or more users take that occur at the same time, or in a particular sequence [4]. Synchronous gestures allow users to pass signals between wireless devices to facilitate dynamic connections. BlueRendezvous extends and adapts these ideas to suit them to the strengths and weaknesses of Bluetooth networking on Smartphones. Some embodiments also consider other networking foundations, such as 802.11, as well as additional features and capabilities that form extensions and alternative implementations. BlueRendezvous contributes capabilities and features suggested by user task scenarios for the ultra-mobile SmartPhone form factor.

Unpublished white paper 1-26-2006



Press a numeric key 0-9 to start a rendezvous with another phone

Waiting for someone to rendezvous with

Checking

Fig. 1. **Left**: BlueRendezvous welcome screen. Users connect by pressing the same numeric key at about the same time. **Right**: After simultaneous key presses, the devices attempt to discover one another.

THE BLUERENDEZVOUS CONNECTION PROCESS

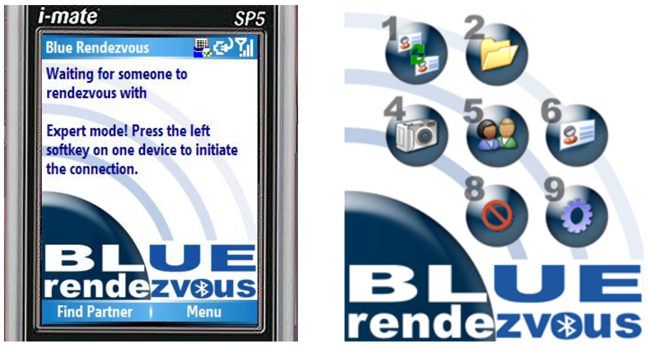
A BlueRendezvous connection encompasses the following general steps:

1. *Start:* Each device must be running the BlueRendezvous application, which activates the Bluetooth radio if necessary.
2. *Synchronize:* One user presses a numeric key from 0-9. The user of the other device must also press the same key at about the same time (Fig. 1, left). Typically, the users would verbally coordinate which key to press.
3. *Connect:* Each user can simply wait for the devices to automatically discover one another (Fig. 1, right). To speed the connection process, one of the users can press the *Find Partner* soft key (Fig. 2). This identifies that user's device as the client.
4. *Choose Action*: Once the devices have established a connection, the action screen (**Error! Reference source not found.**) appears on each device. This screen shows ways that the two connected devices may share information or resources. Either user, or both users, may initiate any of the actions.
5. *Right of Refusal.* If a user initiates an action, typically the other user is offered the opportunity to accept or

refuse the results of that action. For example, the user may be shown a thumbnail of an image offered by the other user, with a Yes/No option to receive it (*Fig. 8*).

6. *Shutdown*. Upon exiting BlueRendezvous, the application automatically closes the connection and reverts the Bluetooth radio to its previous state.

Note that this process is extremely simple and does not require the user to know anything about how Bluetooth works, the names or addresses of the devices involved, or any other technical information. It is also a general purpose mechanism that integrates many different possible cross-device operations into a single interaction model. Thus the method is both easy for users to understand yet also powerful enough to provide many capabilities.


Waiting for someone to rendezvous with

Expert mode! Press the left softkey on one device to initiate the connection.

Fig. 2. **Left**: Expert mode allows one of the users to speed the mutual device discovery process by pressing *Find Partner*. **Right**: Close-up of the Actions screen, a ZoneZoom [10] screen that allows either user to choose a cross-device operation. The available operations depend on the capabilities and user-controlled configuration of each device.

RELATED WORK

We explored synchronous gestures across distributed devices as a way to coordinate activities of wireless devices. Example gestures that we explored include bumping devices together [2,3] or making a pen stroke across multiple pen-operated devices (known as stitching [4]). These were covered in our previous patent filing DISTRIBUTED SENSING TECHNIQUES FOR MOBILE DEVICES.

An early example of a type of synchronous gesture is Smart-Its Friends [5]. A user can hold two smart-its devices together, and shake both of them to associate them (make them 'friends'). A similar shaking pattern is sensed by each device via its embedded accelerometer, and shared with other devices via a low-power radio. The devices then beep whenever such a 'friend' device is sensed entering the range of the wireless radio.

Simultaneous pressing of a button on two different devices is a type of synchronous gesture that was explored in Rekimoto's SyncTap work [8]. Note that Rekimoto relies on precise simultaneous timing of the button press; this means that typically one user will have to press a button on both devices at the same time.

BlueRendezvous uses the button press differently than SyncTap. First, BlueRendezvous accommodates the simple fact that each user has his own personal device and would likely prefer to press a button himself rather than ceding control of their device to the other user. Second, BlueRendezvous only requires the button presses to be close together in time, not precisely synchronized. Bluetooth discovery takes several seconds to occur. During Bluetooth discover one device must transmit that it is 'discoverable' while the other device must listen for discoverable devices. Thus, during our automatic connection process, each device randomly alternates between searching and discovering until the devices find one another. The button presses only must be close enough together so that there is time to for one device to discover the other device before it stops transmitting.

BlueRendezvous also builds on MSR's ZoneZoom [10] technique to integrate the selection of multiple ways to connect the devices into a simple choice mechanism that is well suited to the SmartPhone form factor.

Pick-and-Drop is another early cross-device interaction technique that assumes the existence of a pen with an encoded unique ID that can be read by each device [6,7]. However, pick-and-drop depends on a shared ID code rather than synchronous gestures. A number of other efforts have subsequently explored proximity-based device discovery and identification based on RFID tags [1] or similar technologies [9,12]. The near-field communication standard (NFC forum) is a current commercial realization of these technologies. Note that RFID does not naturally differentiate the receiving device from the sending device; it also typically forces the devices to be touching or in very close proximity, and thus places restrictions on how devices can be connected in a meeting room scenario, for example, where other users may be out of reach beyond arm's length.

EXTENDED DISCUSSION OF CONNECTION STEPS

Start (Step 1)

In some embodiments, BlueRendezvous runs as a background process that responds to particular keypresses. For example, and time that the user hits the # key or a dedicated Rendezvous keys, the device could automatically attempt to discover a partner device that had experienced a nearly simultaneous keypress.

Syncrhonize (Step 2)

Alternatively, users can press different numeric keys or a combination of keys as a more secure connection mechanism. For example, we each could key 7109 on our numeric keypads as a private key to unlock a message encrypted with a well-known public key.

Connect (Step 3)

For the automatic discovery process, the device with the lowest Bluetooth address or unique device ID becomes the server. For the explicit Find Partner process, the device that does not see any keypress defaults to being the ‘server’, that is, it simply continues waiting for another device to discover it. A Bluetooth connection can be established more quickly in the latter case because each device knows which role to take; otherwise the devices must interleave acting both as server and client, until they successfully discover one another. This may take numerous tries because Bluetooth devices can transmit signals that allow other devices to discover them, or they can search for discoverable devices; but they cannot do both at the same time. Note that in some embodiments, buttons or gestures instead of or in addition to the *Find Partner* soft key may be used to distinguish the client device.

Pressing the numeric key twice, or tapping and then holding the key (Fig. 3), offer alternative ways to distinguish the client device. Note that such patterns may be faster for experienced users, since the user does not have to move their fingers between the keys or buttons, and pressing the same key twice (for example) is a fast motion that requires little or no diversion of attention to the keypad.

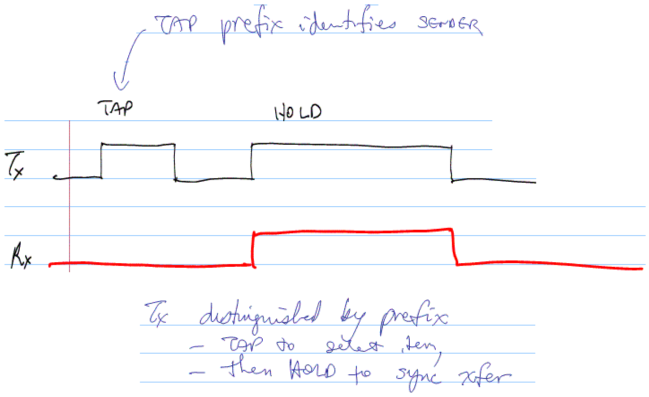


Fig. 3. Double-tap, Tap then hold, or just holding a key for longer than a usual keypress offer other ways to distinguish the client vs. server devices.

A further way to speed connections is to look specifically for device(s) with known Bluetooth addresses. For example, these could be acquired via previous BlueRendezvous connections. The user can choose a specific device, scan through a list of *Favorite Devices* or *Recent Rendezvous Devices*, or even start with these mechanisms but default to the standard process (i.e. the Connect step for unknown device discovery) if the ‘known’ device is not found. Searching for a known device address is particularly advantageous in an environment with many discoverable devices, since otherwise a device may have to query many other devices to determine which is the desired one. Searching by address also offers a way to ease interactions with known or commonly used devices. Note that the user interface need not expose the address itself to the user; it could replace this with a thumbnail photograph of the device’s user, or the name of the device’s owner, which presumably would be exchanged the first time two devices discovered one another as ‘unknown’ devices.

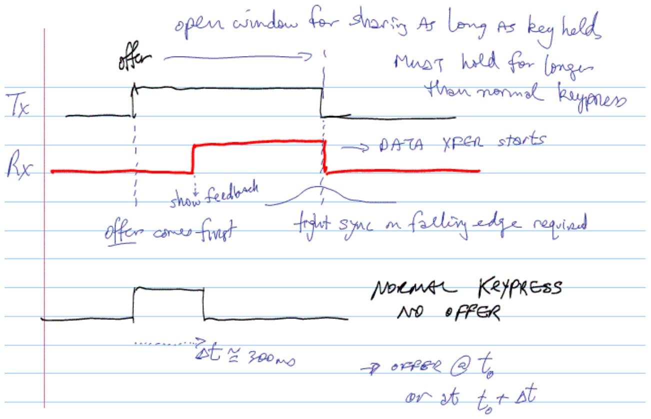
This designates one device as the client, which speeds device discovery: once each device knows its role, one can advertise itself as discoverable, while the other can spend all its time discovering.

In expert mode, one of the users performs an action (e.g., pressing the # key and then the Find Partner soft key, or tapping numeric key associated with the desired operation twice) that specifies his device as the sender (initiating device). Due to current technical limitations of Bluetooth, having the users distinguish the server vs. client device up front may speed the discovery of the desired partner devices.

Multi-Purpose Key Presses and Temporary Connections

In some embodiments, the same keys might be used for both alphanumeric entry and for synchronization between devices. For example, pressing and holding a key could be used to initiate a BlueRendezvous connection.

A connection also might be held open only so long as one or both users hold down a button. This would be particularly useful for transient connections, and would offer a way for users to explicitly control how long the other user had to respond. For example, if a device received a connection after the user lets go of the key, the device could refuse that connection.



Choose Action (Step 4)

The actions may vary depending on the specific devices connected, as well as the context of use (e.g. time of day, where the phones are located, current phone profile, etc.). Multiple screens of actions or hierarchically nested actions may be available.

Once two devices have discovered one another, the screen shows up to 12 icons arranged in a grid pattern. The grid pattern corresponds to the physical keypad of the phone, and activates the corresponding functions in a manner similar to our previous work with the ZoneZoom technique.

Note that the options presented in the grid may depend on the type of the devices that connect, who the person is that you are connecting to, or the user-specified settings on each device. If a zone is empty, hitting the corresponding key may be ignored, bring up a dialog to define a custom cross-device function, or trigger a default action.

Spatial Parameters for Actions

One straightforward extension for some actions is to use the ZoneZoom grid to specify spatial parameters of the connection. For example, one could share an image with another device such that the two devices show the selected image spanning the two screens. If each user simply selects the direction of the other user, this provides enough information for the system to correctly rotate the portion of the image shown on each display to create the illusion of a cross-device tiled display. The design shown below (Fig. 4) mixes keys that select the type of action with keys that specify the relative orientation of the device to act upon. For example if I hit '4' and my friend hits '6', this might share the image across the two screens such that his screen is to my left, but my screen is to his right.

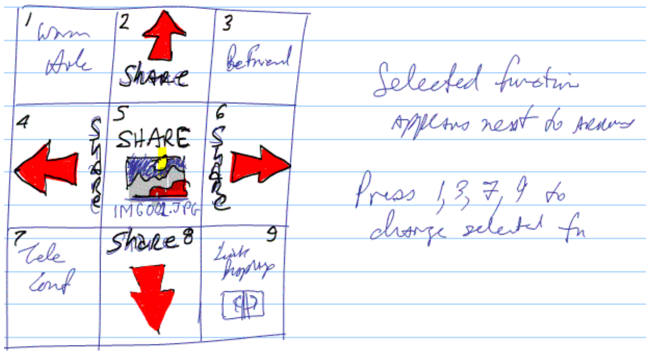


Fig. 4. The combination of keys pressed may indicate spatial operations as well as the type of function. The system does not sense the relative spatial location of the devices: users hit the arrow keys to specify this.

We explored similar ideas in for bumping devices together and stitching [4], and Rekimoto proposed a related idea for SyncTap [8].

In a system that actually does have some capability to sense the relative orientation or proximity of other devices, BlueRendezvous' ZoneZoom connection metaphor may be use to distinguish between various proximal devices. The following figure (Fig. 5) shows a design that distinguishes designs by signal strength (weak, medium, strong) and presents them in a perspective view. This may make it easier to determine which nearby device to connect to or use for a joint operation. Note that in this case, the other devices may not belong to a particular user (e.g., a projector or printer) and thus may not be any way to perform a near-simultaneous key press on the other device.

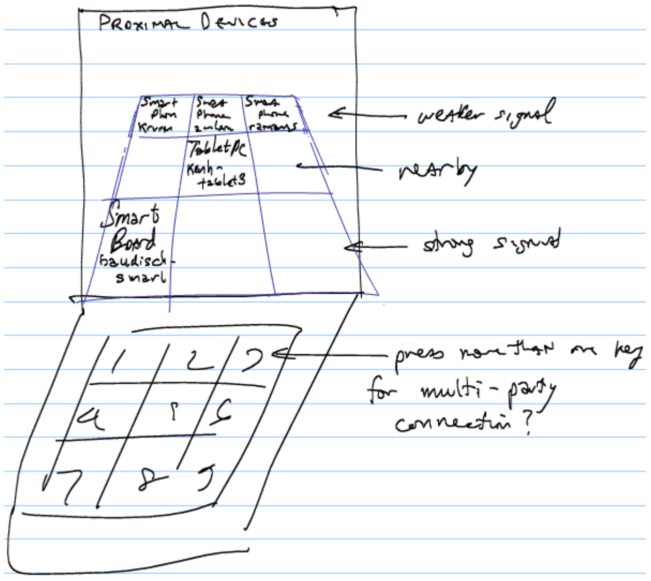
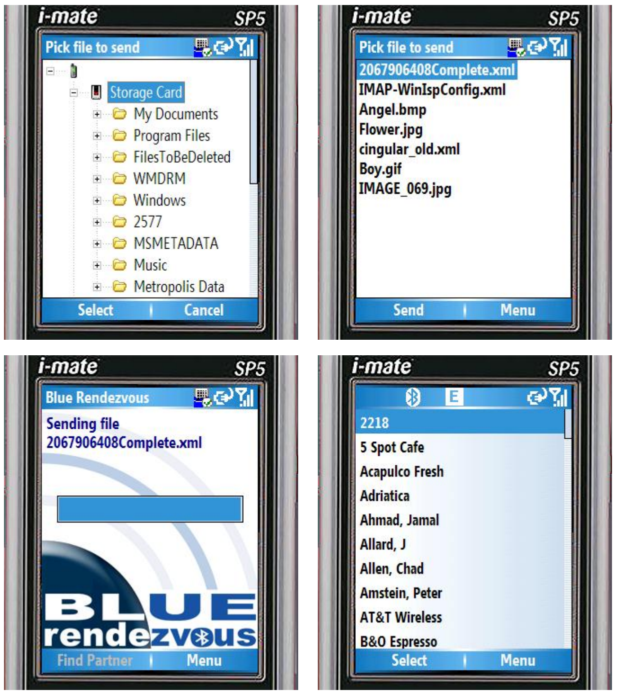


Fig. 5. Selecting among proximal devices ranked by signal strength.

Selection: Specifying the Scope of an Action

Note that users also may specify the scope of an action by forming selections prior to or as a consequence of selecting an action. For example, the user may choose one or more files to transmit to the other device (Fig. 6).



Pick file to send

Storage Card
My Documents
Program Files
FilesToBeDeleted
WMDRM
Windows
2577
MSMETADATA
Music
Metropolis Data

Pick file to send

2067906408Complete.xml
IMAP-WinIspConfig.xml
Angel.bmp
Flower.jpg
cingular_old.xml
Boy.gif
IMAGE_069.jpg

Blue Rendezvous

Sending file
2067906408Complete.xml

2218
5 Spot Cafe
Acapulco Fresh
Adriatica
Ahmad, Jamal
Allard, J
Allen, Chad
Amstein, Peter
AT&T Wireless
B&O Espresso

Fig. 6. **Top Left**: A file control allows users to specify which file(s) to transmit. **Top Right**: Selecting a file to send. **Bottom Left**: When the user hits *Send*, a progress bar gives feedback on the transfer. **Bottom Right**: Similarly, hitting 6 (send contact) brings up a contact chooser.

Choosing a file to send. In this embodiment, the user can navigate a file control to choose any file on their device to transmit. Middle: Choosing the final file to send. In some embodiments, users may start from a default directory or a list of recent files, images, or other content to speed the selection process. Right: Once a file is selected, the sender sees a progress bar that gives feedback on the transfer.

Hitting '6' allows the initiating user to select any contact to send to another device.

Rather than using the standard folder and list controls, content selection controls that build on the ZoneZoom approach can also be envisioned. The following design (Fig. 7) shows one user selecting a contact to send to the other device. This design also helps to emphasize which device is sending and which device is receiving. Simple feedback of this sort could help users avert mistakes or erroneous connections, and is helpful particularly if both users initiate actions at the same time (as supported by BlueRendezvous).

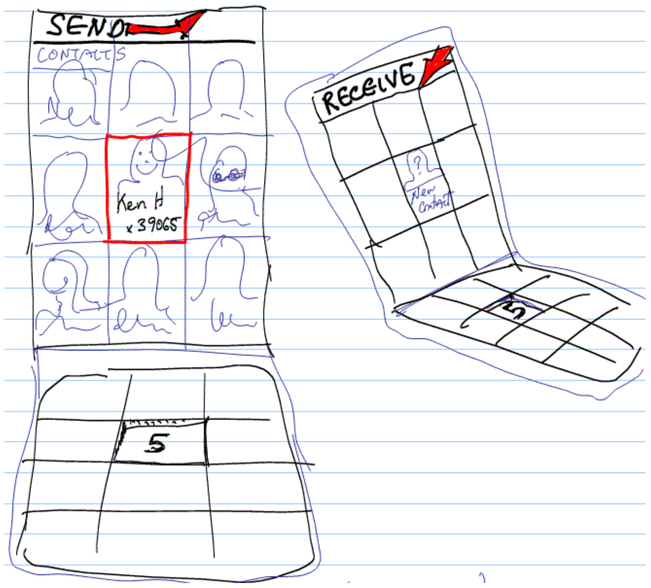
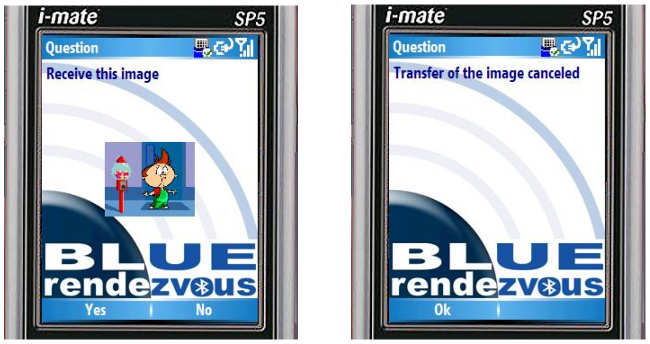


Fig. 7. Using a ZoneZoom based selection metaphor for choosing which contact to send. This design also provides additional feedback to distinguish the sender from the receiver.

Right of Refusal (Step 5)
Some actions may not offer any refusal. Some actions may offer an ongoing option to terminate the connection or undo the results of the operation.



Receive this image

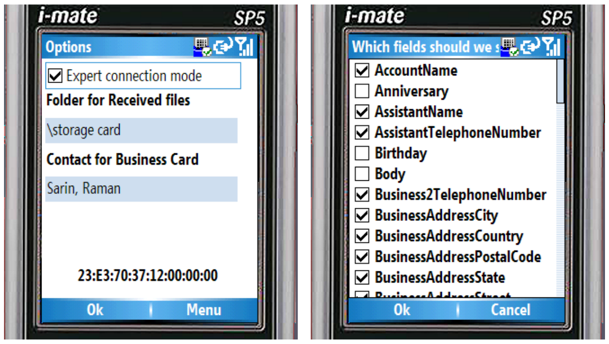
Transfer of the image canceled

Fig. 8. **Left**: A user sees a thumbnail preview of the image that another user is transmitting, and can choose whether or not to accept it (Yes/No). **Right**: Hitting NO allows the receiving user to veto the transfer from the other device.

If the sending user hit 4 (send image), once that user selects an image, the other user will see a small preview of that image. This helps the sender to know exactly what they are agreeing to receive before they actually receive it. The receving user can hit YES (or in some embodiments, simply wait) and the image will complete downloading. In some embodiments, the information is prefetched while the system waits for the user to answer YES or NO. Right: Hitting NO allows the receiving user to veto the transfer from the other device. Note that the sender receives similar tentative feedback and can also choose to cancel the operation prior to its completion.

Shutdown (Step 6)
The connection may be broken, and the radio also may be turned off, after completing an action, after a timeout, after the devices move beyond wireless range of one another, when one device enters a low battery power state, or upon explicit selection of a *Disconnect* command by either user.

Other Options and User-Controlled Settings
BlueRendezvous may expose numerous options and configuration settings to the user (Fig. 9). The ability to shortcut the device discovery using the *Find Partner* soft key is controlled by an option known as Expert Mode, which is enabled by default. The user can also specify the default locations to send or receive files, as well as which contact to use for the business card swap action; another dialog (Fig. 9, right) allows the user to control exactly what fields of the contact should be shared as part of the business card. The bottom of the main options panel (Fig. 9, left) currently displays the device's Bluetooth address, which may be useful for troubleshooting with support staff.



☑ Expert connection mode
Folder for Received files
\storage card
Contact for Business Card
Sarin, Raman
23:E3:70:37:12:00:00:00

☑ AccountName
☐ Anniversary
☑ AssistantName
☑ AssistantTelephoneNumber
☐ Birthday
☐ Body
☑ Business2TelephoneNumber
☑ BusinessAddressCity
☑ BusinessAddressCountry
☑ BusinessAddressPostalCode
☑ BusinessAddressState

Fig. 9. **Left**: Main options panel for BlueRendezvous. **Right**: Options dialog that specifies which contact fields should be included with the user's digital 'business card.'

IMPLEMENTATION DETAILS

How BlueRendezvous makes a connection

The non-expert mode connection of BlueRendezvous is unfortunately not as straightforward as I might have liked it to be. The problems are two fold, number one is that both devices are running the exact same piece of code, now this in of itself is not a huge issue. However the second issue is a much larger technical challenging owing to the design of the Bluetooth chip which is used in most windows mobile devices. This chip it seems is incapable of both advertising it's availability as a useable device and searching for available devices at the same time.

In other words if both devices are shouting trying to find a partner neither one would hear the other. It can be seen that because of this second limitation the first limitation, that both devices are running the same code, becomes a more interesting and difficult bar to pass over.

Essentially what we need to do is guarantee that there is an interval in which one device is looking and the other is listening. The simplest description of my algorithm is that I randomly pick a string of 1's and 0's on each device and iterate through it. On zero I look for other servers, on 1 I patiently sit and wait to be discovered.

First in order to attempt to make sure the strings will be different as a random seed I thought of using the current time, but one must assume that two network enabled phones could possibly be time synchronized to a third server, and that this would not be truly unique. Next I thought about using the address of each device as a random seed, this is guaranteed unique, but is non-variable for the individual device and therefore each device would always use the same string, not super desirable. In the current version of the code I use the current TickCount on the device. In windows the TickCount is the number of milliseconds the device has been up and running for (time since last power cycle), so this is likely to be unique on each device.

I further attempt to optimize this by measuring how long it takes to look for servers, and then using that value as the amount of time I wait if my string has 1. Each string of possible values is 6 digits long. After we've failed six times we will revert back to the "Expert Mode" where the user of one device can press a key, which will force the device into the "Looking for servers" mode, if the other user has not pressed that "Find Partners" button their device will be in listen for clients mode, and will therefore rapidly be found.

Okay so we've now got an understanding of the iteration process works, what exactly is going on when a BR server starts, and what is the client looking for?

Once a number has been pressed on the keyboard BR starts a Bluetooth service using a GUID, the last digit of which is the key pressed by the user. What this means is that when the prospective client is looking for a server it won't find one on which someone has pressed a different number.

The device which is attempting to discover a server basically looks for all discoverable Bluetooth devices in range, and makes a list of what it finds, from this point it queries each one to see if they support a service with this GUID. If they do, it immediately attempts to connect to that device, if it succeeds we move into the main BR screen of asking the users what operations they would like to do. The other devices being in waiting to be a server mode will detect the connection and also enter this mode.

From here it doesn't really matter which device is the client and which is the server they're both just sitting their waiting for data coming in, and sending it out when necessary.

A word on Bluetooth bonding or pairing

All it seems to be (by inspection) is a way of having two devices remember each others address so that the process of discovery is unnecessary the next time a connection is required. There is no requirement that to devices be paired together in order for the communication to occur, if one of the devices knows the other's Bluetooth Address than a connection can occur if the other device will allow it. It is because of this that there is no requirement for two devices to know about each other prior to BlueRendezvous running, and they will not know about each other afterwards either.

Architecture and API's

<Some discussion of the Sink/Source architecture and API's may also be appropriate. However my current inclination is that this might be the basis for a future patent when we have evolved the architecture a bit further…>

EXAMPLE CROSS-DEVICE RELATIONSHIPS

The activities shown in the actions screen of Fig. 2 are only a few representative actions. Additional actions may be offered by having multiple action pages that the user can flip through using the soft keys, miniature joystick, or other keys, or by organizing actions hierarchically as afforded by ZoneZoom [10].

The following sections list various ways that we anticipate BlueRendezvous could be used to link devices and share information between devices, organized by general feature areas. *<TODO: sketches of some of the more interesting scenarios?>*

Communication Channels

- Chat (via SMS or immediately over the wireless)
- Initiate phone call with other device (pairing used to obtain phone number, then auto-dialed)
- Teleconference (add other device to existing call)
- Email (send a new email, forward an existing email)
- Start a chat (text and/or voice and/or ink markup if device has a touch screen).

Content

- Swap business cards (e.g. a designated contact)
- Send a contact
- Send a file
- Send a photo
- Slide show (send series of photos; other user watches as I step through them, but also has option to override me and take control)
- Send a task (todo item)
- Add a task to TODO list based on info from other device (e.g. send action item, exchange action items)
- Send an email (existing message)
- Send my phone number
- Send a calendar entry
- Send a location (e.g. sensed by GPS or wireless/cell signal strengths) to other user
- Send *this* location to other user (e.g. from a GPS phone to one without GPS)
- Send voice memo to other device. This might be an existing audio file or one I generate right now as part of the connection process.
- Send favorites (one or more links) to other device. Other user may be offered option to click on one of them now.
- Send screen capture to other device
- Pass permissions, public/private key, or certificates to other device that will allow it to access some other service at a later time.
- Temporary content: Loan / Offer to share. Other user may have read access only; file self-deletes after a set time period; rights may change or expire after a default or selected time interval.
- Send comments / mark-up on a document. Ink, voice, and textual comments associated with a particular file.

Meetings

- Set up a meeting (new one that is pre-populated with the two users involved)
- Set a meeting with (owner of) other device (add only to my calendar?)
- Invite to a meeting / calendar appointment (an existing one from my calendar; option to formally add other user to the meeting request)
- Forward a meeting request (may not be my meeting; other user not added to meeting request)
- Set up recurring meeting with other person “at this time”: e.g. once a week starting at the hour or half-hour closest to now; once a month; biweekly; daily.

People & Social Networking

- Befriend other device. Sounds cool but what exactly does this mean? Perhaps this is just a catch-all for accessing all the other features in this category.
- Add to quick-connect list (accelerates pairing process when we next attempt to connect these devices).
- Save status of joint task / connection state with other device. When we next pair these devices the same state (applications, documents, etc.) is restored.
- Notify me when other device is nearby (e.g. via Bluetooth discovery or 802.11 multicast strategies). Notification via sound, vibration, and/or pop-up window. Related to concept of Smart-Its Friends [5].
- Refer / introduce other device to a ‘friend’ device
- Collaborative filtering: have a way of rating the other device or person; “don’t link up with this idiot”; “has good content to share” etc.
- Add other device to a preexisting social networking service on the Web or a physical-device-specific one.
- Temporary Friend/Contact. Expires after a given time period, e.g. after a meeting or after a week without contact.
- Invite other device to multi-person game.
- Wormhole between devices. This means the current synchronous connection creates an object on each device that allows them to communicate further at a later time. So I could select a file later and drop it on the wormhole, and the other person would get it. Wormholes may be restricted to certain communication channels (e.g., it might only remain valid while the other device remains within wireless communication range).

Combined Resources

- Dual display, Connect screens of one or more devices together in various relative orientations (Fig. 4). See also synchronous gestures [3] and stitching [4] work.
- Use my location sensor and your location sensor to get a refined location estimate. Could be extremely cool if our two devices have complementary networks, connection media, sensors, signal strengths.

- Combine input capabilities from the two devices, e.g. now you have miniature joysticks for different purposes. Use two-thumb keyboard on my friend's phone to enter comment on my phone (which has only the numeric keys).

Remote control

- Send web browser on other phone to a web address from my favorites list.
- Remotely trigger other device to take a picture. (e.g. we pair our phones, I set my phone down and we stand together, then you trigger my phone to snap the photo). In general remote control various features of other device.
- Remote administrate other device (e.g. fix or change settings for other user)
- Accept input or output from another device. For example, my phone could send a type-in box to the other phone for the other user to fill out. When that user hits enter, the results are sent back to me. Similarly, a portion of my screen output could be mirrored on the other device. See Tan's WinCuts work for related concepts [11].

e-Commerce and gift giving

- Buy something and have it shipped to the owner of the other phone. For example, I find a book on Amazon using my phone. Then I BlueRendezvous my friend and complete the order such that he receives the item. A similar feature would be the equivalent of "collect calling": my friend pays for the item, but it goes to me.
- Share ring tone, music. This might be a "for trial only" and one has to pay the provider / copyright holder at a later time to keep using the object.
- Send gift to other device (e.g. gift of credits / eCash, talk minutes, …)

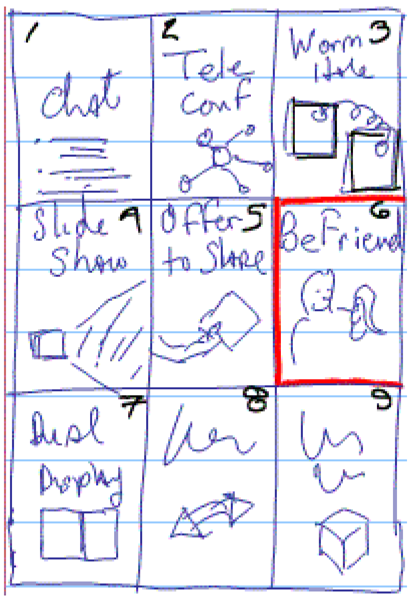
Putting off, non-connection, blocking, ambiguous refusal

Even if one wants to connect to another person's device, one may not actually want to do it right now, so deferring the immediate connection request to an asynchronous channel could be desirable. Furthermore an undesired person may seek to discover or connect to your device. In social settings, it may be rude or embarrassing to outright refuse the person's offering. Having various mechanisms available to block connections, fake failed connections, or send incorrect contact information may be desired to escape these situations without a confrontation.

- Defer synchronous connection with this device, and move over to an asynchronous channel (e.g. send email with contact info or link to information or a shared web site where users may exchange messages, files, etc. at a later time).
- Block this device (similar to adding an email address to blocked sender's list)
- Send 'bogus' contact info to other device, e.g. an intentionally incorrect phone number
- Face-saving connection refusal. Methods to decline connection in a face-saving manner. This allows one or both devices to feign technical errors or lack of connectivity as a way for one of the users to opt out of the connection/transfer without causing the other person to lose face. Feigning low battery, poor signal, connection dropped, etc. are also likely sneaky ways to opt-out.
- Non-ambiguous refusal. 'NO', "Connection denied", "you do not have permission to connect to the other device", "permission refused", "Sorry", "Busy", etc.

Actions in Context

- Offered actions depend on other device that is connected to
- Offered actions depend on multiple other devices that are also detected nearby
- Offered actions depend on our physical location, e.g. at home, there are options to share media; at work, options to share files and data are emphasized; in different cities or countries, socially appropriate features are the default.
- The options may be completely absent, or just emphasized/deephasized depending on the context
- Offered actions depend on the day of the week or the time of day, e.g. weekend (fun, dating, stupid party tricks) vs. weekday (work, calendar, emails, etc.)



CONCLUSION

xxx

REFERENCES

1. Harrison, B., Fishkin, K., Gujar, A., Mochon, C., Want, R. *Squeeze Me, Hold Me, Tilt Me! An Exploration of Manipulative User Interfaces*. Proc. ACM CHI'98 Conf. on Human Factors in Computing Systems, 17-24.
2. Hinckley, K. *Distributed and Local Sensing Techniques for Face-to-Face Collaboration*. ICMI-PUI'03 Fifth International Conference on Multimodal Interfaces, 81-84.
3. Hinckley, K. *Synchronous Gestures for Multiple Users and Computers*. UIST'03 Symposium on User Interface Software & Technology, 149-158.
4. Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., Smith, M. *Stitching: Pen Gestures that Span Multiple Displays*. ACM 7th International Working Conference on Advanced Visual Interfaces (AVI 2004), 23-31.
5. Holmquist, L., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., Gellersen, H. *Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts*. Ubicomp, Springer-Verlag, 116-122.
6. Rekimoto, J. *Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments*. Proc. ACM UIST'97 Symp. on User Interface Software & Technology, 31-39.
7. Rekimoto, J. *A Multiple Device Approach for Supporting Whiteboard-based Interactions*. CHI'98, 344-351.
8. Rekimoto, J., Ayatsuka, Y., Kohno, M. *SyncTap: An Interaction Technique for Mobile Networking*. Mobile HCI 2003, Springer, 104-115.
9. Rekimoto, J., Ayatsuka, Y., Kohno, M., Oba, H. *Proximal Interactions: A Direct Manipulation Technique for Wireless Networking*. INTERACT 2003.
10. Robbins, D. C., Cutrell, E., Sarin, R., Horvitz, E. *ZoneZoom: Map Navigation for Smartphones with Recursive View Segmentation*. ACM Advanced Visual Interfaces (AVI 2004).
11. Tan, D. S., Meyers, B., Czerwinski, M. *WinCuts: Manipulating Arbitrary Window Regions for More Effective Use of Screen Space*. Short paper at CHI 2004.
12. Tandler, P., Prante, T., Müller-Tomfelde, C., Streitz, N. A., Steinmetz, R. *Connectables: dynamic coupling of displays for the flexible creation of shared workspaces*. UIST 2001, 11-20.
13.