Opportunistic Awareness: Annexing Peripheral Devices for Information Monitoring

Heather E. Mahaney & Jeffrey S. Pierce
College of Computing
Georgia Institute of Technology
810 Atlantic Ave
Atlanta, GA 30332 USA
+1 404 385 4361
{mahaneyh, jpierce}@cc.gatech.edu

ABSTRACT
Opportunistic annexing enables users to improve the I/O capabilities of their handheld devices by annexing resources in their environment. By developing an awareness client over opportunistic annexing, we can allow mobile users to take advantage of pervasive computing infrastructure to monitor information in the periphery of their attention. Important considerations when designing architecture for the awareness client include automating discovery, authentication and connections. Developing interfaces to split across multiple, undetermined devices also raises challenges for design. By building an awareness client over opportunistic annexing, we can improve mobile users' capabilities to monitor information in the periphery without directly interaction with their handheld devices.

Keywords
Handheld devices, opportunistic annexing, peripheral interaction, multi-device interfaces

INTRODUCTION
As computing infrastructure becomes more pervasive in our environments, users increasingly have access to information anytime and anywhere. PDAs and cellular phones enable users to carry computing power in their pockets. Wireless technologies including WiFi and Bluetooth extend the capabilities of these devices by allowing users to remain connected to other people and sources of information while mobile. This connectivity encourages richer interactions and experiences, however it also increases the information overflow already bombarding many users.

Our research focuses on improving the I/O capabilities of handheld devices through *opportunistic annexing*. Opportunistic annexing is the process of temporarily attaching a device to a computational environment to enhance its capabilities. Many of the obvious applications of opportunistic annexing enhance traditional uses of handhelds. Users might annex a keyboard to type a memo into their PDA or a monitor to see their schedule for the month without scrolling. While we believe these applications will be useful, they only represent a subset of the interesting interactions enabled by opportunistic annexing.

In addition to improving the I/O capabilities of handheld devices, opportunistic annexing can allow users to take advantage of more convenient resources through their PDAs and cellular phones. At her desk, a user may use her handheld devices to monitor information in the periphery, relying on her PDA to alert her of appointments or her cell phone to signal an incoming call. Opportunistic annexing enables the user to view details of her appointment by annexing her desktop monitor. She could annex speakers and a microphone to have a conversation on her cell phone. While mobile, a user may hear alerts from her handheld devices but must retrieve the device and focus her full attention on it to interact. Opportunistic annexing could allow the user to take advantage of mobile connectivity to monitor interesting information without getting her handheld out of her bag.

We propose an *awareness client* running on top of the opportunistic annexing architecture to enable users to monitor information while mobile. Our awareness client would collect information a user was interested in monitoring such as incoming email and phone calls, changes in the stock market and sports scores. The awareness client would annex a device at the periphery of the user's attention to signal a change in monitored information. Creating an awareness client involves many challenges. The awareness client must discover available devices and determine which devices should display an alert. The awareness client must also manage authentication and establishing connections. Interfaces for awareness must be designed to divide across multiple, unspecified devices. When designing interfaces we must also consider how to display alerts with varying degrees of urgency and privacy. An effective awareness client would help the user manage information in the periphery of their attention without requiring direct interaction with a device.

ANNEXING ARCHITECTURE

Users may annex devices either directly or through an intermediary. We anticipate that in the short term handhelds will annex most devices indirectly using the computer they are attached to as an intermediary. As more I/O devices become wireless, through Bluetooth, WiFi or IR, handhelds will establish direct connections to annex resources. A primary issue for annexing devices is whether the user annexes devices using his handheld or using the devices he wants to annex. The former can be thought of as pushing the interface out to other devices from the handheld and the latter as pulling the handheld interface out using other devices.

Annexing via pushing. Annexing via pushing is better for both privacy and security. Pushing provides more privacy because the handheld does not need to keep other devices aware of its presence, reducing the risk of people tracking a user by monitoring the presence of his handheld. Pushing eliminates the risk of unauthorized connections and removes the processing and power burden of distinguishing valid and spurious connection requests. The primary disadvantage of pushing is that the user needs direct access to the handheld; this approach is not possible if the handheld is in the user's backpack.

Annexing via pulling. Annexing via pulling allows users to annex devices without interacting with their handhelds. The primary drawback of annexing via pulling is that it forces users to authenticate themselves to their handhelds when annexing a device. While annexing via pushing should be equally straightforward for both directly and indirectly annexed devices, annexing via pulling is likely to be more difficult for directly annexed devices, especially nontraditional devices with impoverished I/O.

Annexing for Mobile Awareness

To extend opportunistic annexing to mobile awareness applications, we must construct an awareness client over the annexing architecture. For a device to be annexable, it would need to support opportunistic annexing but would not need to run a specific client for awareness applications. Opportunistic annexing supports general interface passing, hence an interface for information monitoring could be sent to the annexed device like any other. Typically when annexing, the user decides which devices to annex and is responsible for authentication. Annexing for mobile awareness raises different challenges. Since the user will not be directly interacting with the handheld device, the awareness client must locate devices to annex, decide which devices are appropriate to annex, establish connections and determine the methods of authentication.

When locating devices to annex, the awareness client must find available devices at the periphery of the user's attention. These could be wearable devices or other available I/O resources located near the user. The Join and Capture system [5] allows a user to *capture* I/O devices she encounters in the environment. Once a device is captured, it subscribes to the user's session and its behavior is synchronized with other captured devices in that session. The user connects to devices by plugging her JAVA ring in to an iButton connector on the device she wishes to capture or by using the People Watcher that allows the user to simply touch a device to capture it. However, since our focus is on periphery interactions, we do not want to burden the user with manually discovering devices to monitor information.

Because mobile users' computing environments may constantly change, we need a way to automate the connection process. Technologies such as Smart-its [1] monitor movement, connecting devices that move with the same rhythm. This could be an option for connecting handhelds to wearable devices that would have similar movements while the user was mobile. The awareness client can also discover devices through polling. Bluetooth devices may advertise both their address and available services, allowing them to be discovered by an awareness client running over Bluetooth. Because handhelds have limited battery power, constant polling for devices is not an option. We could discover devices only when a message needs to be sent to the user. For this method to be effective, the speed of discovery and connection must be quick enough to notify the user of a phone call in time for her to answer. Handhelds enabled with GPS or other position detecting hardware could look for available devices when the user moved to a new location and cache the addresses for later use. This could provide quicker connections but the location service may also drain battery power.

As with other types of annexing, it may be most practical for the awareness client to establish connections using the push model. Assuming that the user has initiated the awareness client before placing the handheld in a pocket or bag, we can consider the user already authenticated to her own device. When there is a message for the user, the awareness client would push the alert onto the devices it chose to annex. Since peripheral devices are in close range by definition, these connections would ideally be established over Bluetooth, using WiFi as a back up if Bluetooth was unavailable.

A user might wish to annex additional devices to receive more information about an alert. This could be done using the pull model to bring up an interface containing more information, onto a nearby screen. In this case, the pull model would require nearby devices to poll for the address of the handheld. An identified handheld could be represented as an icon on the screen which users would click to interact with the handheld through the annexed device. The user would need to authenticate to her handheld to prove the annexed device was allowed to pull

data from the handheld. In most instances, typing a password on the annexed device would be sufficient means for authentication.

In the push model, the awareness client must authenticate to the device it wishes to annex before it may establish a connection. As mentioned above, authentication may be difficult on nontraditional devices without a display or keyboard. In addition, continuously authenticating to multiple devices would distract the user. Using a certification authority could circumvent this problem. A certification authority would receive a signed certificate from the handheld stating its credentials to annex the devices. The authority would then verify the credentials and suggest to the device whether it should allow itself to be annexed. This solution could also be used to authenticate annexable resources to the handheld, proving that they are secure to annex. Because we cannot assume all devices to have access to infrastructure supporting certification, we must explore other authentication methods, especially for wearable devices. Movement monitoring technologies [1] could provide implicit authentication for wearable devices. The fact that a user is in possession of a device may be good enough to determine that she has the right to annex many of its resources. Another option for authentication with wearable devices would be to manually configure the device to give particular handhelds privileges to connect. This can be done using Bluetooth by exchanging names and passkeys between the devices to form a bond.

INTERFACES

Designing user interfaces when users can opportunistically annex devices presents two particular challenges: learning how to effectively divide interfaces across annexed devices and learning how to design interfaces given uncertainty about the devices users will employ. Designing interfaces for peripheral awareness also brings challenges of determining how to appropriately alert the user in a given situation.

Designing Under Uncertainty

Without knowing which combination of devices a user may annex, developers must come up with new ways of designing the user experience. Designers may choose to handcraft the desired interface, creating different implementations for a variety of possible devices. This approach can work well when designers can accurately forecast what device combinations users are likely to employ, but it has problems when the set of likely combinations is very large. Some researchers have explored automatically generating interfaces (e.g. XWeb [4]). They provide a semantic description that avoids specifying layout or types of input. Devices can use this description to create an interface at run-time that fits their abilities. While this approach allows interfaces to adapt to a variety of I/O devices, it does not guarantee that the interfaces will be aesthetically pleasing or easy to use. Another possible method would be to allow designers to craft sub-components of interfaces but leave their assembly to the handheld at run-time.

We expect that in practice users will primarily annex desktop displays, keyboards, and mice. Rather than choosing one approach or another, a more effective method might be to craft interfaces for common cases and rely on generated interfaces when users annex unanticipated devices. The iCrafter system uses this method [6]. For mobile awareness applications, the commonly annexed devices may differ greatly from standard desktop systems. Specific awareness interfaces for watches, headsets or wearable displays would be designed for the awareness client, giving the system support for multi-modal interactions. Generated interfaces could support awareness on unanticipated devices.

Dividing Across Multiple Devices

Because opportunistic annexing allows users to employ multiple I/O devices, we must explore how to spread the interface across these devices to utilize them most effectively. When designing for mobile awareness, an alert consisting of an alarm and a message may be split between a headset and a watch display. The user may also wish to employ a larger display to view details of the message. This requires the awareness client to determine which I/O elements are available and how the interface elements should be divided and displayed between the devices

Once the awareness client discovers available devices, it must determine which are appropriate to annex. In addition to the availability of devices, the urgency and privacy of the message would help determine where the system should route alerts. User preferences could be used to narrow the options. For example, a user may specify that she would always like to monitor information on her watch except urgent messages should always sound an audio alert through a headset or the handheld itself. Since we cannot anticipate available devices a user could not design rules for each situation. However, we assume that a set of general heuristics could be applied to determine which devices to annex.

To support interface division, we must also have architectural support to tie the interface elements together. We can use the Model, View, Controller model of user interfaces as a framework for discussing where the components may lie in different situations. Deciding where the Controller lives should be straightforward. When the user directly annexes devices, the Controller will live solely on the handheld and devices should transmit their event data directly to the Controller. The devices should communicate with the Controller

according to a standard protocol allowing the Controller to identify the type of I/O stream it is receiving and handle the events appropriately.

Deciding where the other components should live is more complex because there are several valid possibilities. One approach is to send the View to the display by sending the actual pixels; an annexed display can copy them directly into its display buffer. A second approach is to send semantically described data and rely on the display to generate the View similarly to the method used by Xweb[4]. With the former two approaches, the Model remains on the handheld. Another approach is to send the data and a semantic description of the desired View. Cooltown [2] and the Personal Server [8] use this approach, employing HTML to describe the interface. A final approach is to send a Model and View in the form of code along with the data. This approach provides as much flexibility as sending the pixels to display, but it introduces the risk of malicious code. Both Jini [7] and SpeakEasy [3] use code to transfer interfaces between devices.

In mobile awareness applications, handhelds will often annex devices with smaller amounts processing power. We also cannot assume that these devices will have a browser to decode HTML. This suggests an architecture where most of the processing is done on the handheld. Most awareness interfaces would be extremely simple. Displaying an envelope on a watch could signal a new email. A headset could play sounds to alert the user of changes in information they were monitoring. Users may want to define their own alerts to signal changes in monitored information or use defaults defined by the awareness client. These simple signals could be made with the Model and Controller residing on the handheld which would pass the preprocessed signal to annexed device. The complexity in designing the interfaces comes from the many possible combinations of input and output. The annexed device must send a description of its I/O to the awareness client for it to know how to communicate with the device. The awareness client must also know whether a device has both input and output to determine if a combination of devices is required to make alerts interactive. In the case of a watch, users could press buttons to view more details on a message or dismiss the alert. When input is not available, the user could walk to the closest monitor, pulling the awareness application to the screen.

We will also need to learn how users will want to divide an interface to protect their privacy. We need an awareness client that can learn what a user feels is private and can incorporate that information into the alert. An envelope icon may have a seal on it to signal private information. This would help prevent the user from displaying a sensitive message on a large public display. Special alerts could also signal urgency and importance of a message. These values could be set by the sender or predetermined by the user.

CONCLUSION

Opportunistic annexing allows users to augment their handheld devices with I/O resources they encounter in their environments. By adding an awareness client over the annexing architecture, we can extend opportunistic annexing to provide peripheral awareness, enabling mobile users to monitor information without directly interacting with their handheld devices. While annexing for mobile awareness comes with many challenges, we hope that combined research efforts will make this application a reality.

REFERENCES

1. Holmquist, L. E., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., Gellersen, H-W. (2001) Smart-its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts. *Proceedings of UbiComp '01*, 273-291.

2. Kindberg, T. and Barton, J. (2001) A Web-Based Nomadic Computing System. *Computer Networks*, 35(4), 443-456.

3. Newman, M.W., Izadi, S., Edwards, K., Sedivy, J.Z., Smith, T.F. (2002) User Interfaces When and Where They are Needed: An infrastructure for Recombinant Computing. P*roceedings of UIST '02*, 171-180.

4. Olsen, D. R., Jefferies, S., Nielsen, T., Moyes, W., Fredrickson, P. (2000) Cross-modal Interaction using XWeb. *Proceedings of UIST '00*, 191-200.

5. Olsen, D. R., Nielsen, S. T., Parslow, D. (2001) Join and Capture: A Model for Nomadic Interaction. *Proceedings of UIST '01*, 131-140.

6. Ponnekanti, S.R., Lee, B., Fox, A., Hanarahan, P., Winograd, T. (2001) ICrafter: A service framework for ubiquitous computing environments. *Proceedings of Ubicomp '01*.

7. Waldo, J. (1999) The Jini Architecture for Network-Centric Computing. *Communications of the ACM*, 42 (7).

8. Want, R., Pering, T., Danneels, G., Kuman, M., Sundar, M., Light, J. (2002) The Personal Server: Changing the way we thing about ubiquitous computing. *Proceedings of UbiComp '02.*