Wearable Computing

Editor: Thad E. Starner ■ Georgia Institute of Technology ■ thad@cc.gatech.edu

Powerful Change Part 2: Reducing the Power Demands of Mobile Devices

Rajesh Krishna Balan

Consumers expect their mobile devices to be small and light, have a decent battery lifetime, and still be powerful enough to run all their desired applications. The need to satisfy such consumer demands has driven mobile-device manufacturers to make products (such as PDAs and cell phones) that match these needs. However, this inevitably requires trade-offs. Building smaller, lighter devices requires sacrificing the devices' battery capacity because battery technology is still fairly dense and heavy, and contributes most of a mobile device's size and weight. But, mobile devices with the required resources to run useful desktop applications need additional battery power to run those resources for a reasonable length of time.

Because the tension facing manufacturers appears to be in deciding the battery-weight and battery-performance trade-offs, the solution seems to lie in finding ways to increase a mobile device's battery lifetime. You can do this in two different, orthogonal ways. First, you can address the issue at the supply side: either build better batteries that can supply the necessary energy while still being small and light or develop other ways for a mobile device to acquire energy (such as solar power). Thad Starner discussed these methods in "Powerful Change Part 1: Batteries and Possible Alternatives for the Mobile Market" (*IEEE Pervasive Computing*, Oct.–Dec. 2003).

Alternatively, you can reduce the mobile device's demand on battery power, which will let the device last longer before it runs out of energy. The Aura project[1] at Carnegie Mellon University is investigating several techniques for reducing mobile devices' energy demands. Two of the more promising techniques are *cyber foraging*[2,3] and *fidelity adaptation*.[4]

CYBER FORAGING

The idea of cyber foraging ("living off the land") is to dynamically augment a mobile device's computing resources by exploiting nearby servers. A user might discover these servers, known as *surrogates*, and use them opportunistically at different locations in the course of his or her movements. For example, John is in a foreign country and would like to be able to communicate with people there. He realizes that his PDA can perform language translation, but it doesn't have the computational ability or the necessary language files. Fortunately, help is at hand. The coffee shop nearby has a few desktop computers available that John can use as surrogates (see the related sidebar). Discovering this, John instructs these surrogates to retrieve the language files and perform the computation required for the language translation. The surrogates then ship the results back to John's PDA for his use. This scenario highlights two different ways to use surrogates: as *data-staging* or as *compute* surrogates.

Data-staging surrogates

When a mobile device attempts to retrieve files from either a distributed file system or the Web, fetching them can incur long round-trip-time (RTT) latencies if those files are situated on servers far from the mobile client. This high latency requires the handheld client to spend significant time retrieving the distant files, which increases both the mobile user's response time and, more importantly, the mobile device's power consumption.

In these cases, data staging is particularly effective in reducing both the file-transfer latency and overall power consumption. It works as follows: the mobile client discovers that a data-staging surrogate is present and tells the surrogate what files it requires. The surrogate then retrieves the files from the distant file server. This lets the mobile client retrieve them from the nearby surrogate without having to incur any large RTT latencies. To prevent malicious data access, all data stored on the surrogates is encrypted with keys that only the files' owner can decrypt. Research has shown that using data-staging surrogates can reduce a handheld device's file access latency by up to 54 percent.[5] This translates into a substantial power savings because the handheld device will need to be operational for less time to retrieve the necessary files.

Data staging is most effective when coupled with software that can accurately predict which files the user will require in the near future. This lets nearby surrogates precache files that the user will shortly need, greatly reducing

1536-1268/04/$20.00 © 2004 IEEE ■ *Published by the IEEE CS and IEEE ComSoc*

the time the user must wait for the files to become available.

For example, John's PDA realizes that John always checks his email when he goes for coffee at the nearby coffee shop. So, when the PDA detects (using location-aware systems) that John is going for coffee, it instructs the coffee shop's data-staging surrogate to start downloading John's email. When John arrives, his PDA can quickly retrieve his email from the surrogate. John saves time, and his PDA saves power. The Aura project is developing this sort of predictive software, which has shown great promise in being able to accurately predict a user's behavior.[1]

Compute surrogates

Another technique a mobile device can use to conserve energy is to perform remote execution using compute surrogates. This occurs when the device asks a nearby compute server to perform computation on its behalf. For example, John wants to perform language translation, as in the first scenario. However, his PDA is low on battery power, and performing the translation would quickly exhaust what's left. So, John instructs his PDA to ask a nearby compute surrogate to perform the translation for him. John's PDA ships a file containing the language text to be translated to the compute surrogate, which performs the translation and returns a file containing the translated text to John's PDA.

By using a compute surrogate, John can both reduce his PDA's power consumption and improve the translation's response time (the surrogate probably has more resources than the PDA and thus can perform the translation faster). Even though the mobile device consumes a little energy communicating with the compute surrogate, it saves the energy required to actually perform the computation. Research has shown that using remote execution can be highly effective in reducing a mobile device's energy usage.[6]

FIDELITY ADAPTATION

Cyber foraging is only effective when the mobile device has access to surrogates, so what happens when no surrogates are available? Can we do anything to reduce the mobile device's battery demands in this case? The answer is yes, and the solution is fidelity adaptation. Fidelity refers to an application-specific metric of quality that you can adjust by modifying the application's runtime parameters. For example, a speech-recognition application has higher fidelity when using a large speech vocabulary rather than a small speech vocabulary. However, using the large vocabulary requires more energy because the mobile device must perform more disk I/O along with more computation to handle the large vocabulary. Fidelity thus represents a trade-off between power consumption and the application's quality (in general, fidelity settings affect many other resources such as CPU, memory, and network usage as well). You can exploit this trade-off to reduce energy usage, albeit at the loss of application quality.

For example, Fred, an architect, wishes to view a 3D representation of his current project on his PDA. However, the 3D model's frame rate is unacceptably low because his PDA can't cope with the computational requirements of displaying the model. Unfortunately for Fred, no compute surrogates are available to handle the computation for him. He thus has two choices: he can accept the horrible

SURROGATES FOR CYBER FORAGING

You can use data-staging surrogates as data repositories for a mobile device and use compute surrogates to perform remote computation on behalf of an application running on a mobile device. But who will provide the surrogate infrastructure for cyber foraging? Isn't creating this infrastructure expensive? The answer is no. These days, desktop computers sell for only a few hundred dollars, with prices continuing to drop. As such, providing a surrogate costs the same as (or even less than) providing wireless network connectivity. So, we can envision that in the foreseeable future, public spaces will also provide surrogates for mobile devices to use.

These public spaces could include airport lounges, schools, and coffee shops, all of which already provide wireless network connectivity today. The key challenge is thus not the surrogates' cost but how to convince public space owners that providing surrogates is useful for attracting customers (much like wireless access connectivity attracts customers today).

To address this, Aura project researchers have shown that using surrogates is effective in reducing a mobile device's power demands.[1] This is because even though the mobile device consumes energy to establish communications with the surrogates, it still conserves energy overall because it doesn't have to perform the disk I/O or computation the surrogates perform on its behalf. This research, coupled with complementary research[2] demonstrating surrogates' usefulness in improving the response time for mobile users, should encourage manufacturers to build software that actively uses surrogates where possible. This in turn will encourage public-space owners to provide surrogates to attract mobile users.

REFERENCE

1. D. Garlan et al., "Project Aura: Toward Distraction-Free Pervasive Computing," *IEEE Pervasive Computing*, vol. 1, no. 2, 2002, pp. 22–31.

2. R.K. Balan et al., "Tactics-Based Remote Execution for Mobile Computing," *Proc. 1st Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys), Usenix, 2003, pp. 273–286.

frame rate or reduce the model's quality to improve the frame rate at the cost of some detail. Fred chooses to reduce the quality because his goal was to get a project overview, which the lower-quality model still provides. Fred can also extend the PDA's battery life by reducing the quality because his PDA will now spend less energy processing the model for display. The Odyssey project[4] has demonstrated the viability of using fidelity adaptation to improve system performance.

PUTTING IT ALL TOGETHER

In practice, it's not often that a mobile device uses only cyber foraging or fidelity adaptation. More likely, it would use these techniques in some combination. For example, to display Fred's 3D model, the PDA can choose to both reduce the model's quality and then ask a compute surrogate to perform the computation. This lets the mobile device obtain the optimal combination of power savings, latency, and application quality. For instance, if the device chooses to only use a compute surrogate, the power savings and application quality might be high. However, the latency might still be unacceptably high because the model is too big for even a surrogate to compute in a reasonable time period. In this case, the PDA should decrease the model's fidelity and then send it to the surrogate for processing. Systems such as Chroma,[7] Spectra,[8] and Puppeteer[9] have demonstrated that fidelity adaptation coupled with cyber foraging is extremely powerful in saving energy (up to a 30 percent improvement[8]) and improving latency.[7]

Part 1 of this series highlighted different methods for improving a mobile device's energy supply. This included efforts to build better batteries and initiatives using alternative energy sources (such as solar power and pedal power) to run mobile devices. Part 2 focused on how to reduce a mobile device's energy demand using surrogates in the environment and by reducing application fidelity. It's foreseeable that in the near future, we'll use a combination of these techniques to improve a mobile device's lifetime, as this final example demonstrates.

John is in another country and meeting clients at a local café. His PDA uses the surrogates available in the café, coupled with fidelity adaptation to run the language translation software he needs for the meeting. Some time later, John's PDA starts to run out of energy. John thus decides to temporarily step out of the cafe to recharge his PDA using solar power (because he lacks the required power cables to use the local power sockets). When the PDA has recharged sufficiently, John reenters the cafe and resumes his discussion with his clients.

REFERENCES

1. D. Garlan et al., "Project Aura: Toward Distraction-Free Pervasive Computing," *IEEE Pervasive Computing*, vol. 1, no. 2, 2002, pp. 22–31.
2. R.K. Balan et al., "The Case for Cyber Foraging," *Proc. 10th ACM Special Interest Group on Operating Systems European Workshop* (SIGOPS), ACM Press, 2002, pp. 87–92.
3. M. Satyanarayanan, "Pervasive Computing: Vision and Challenges," *IEEE Personal Comm.*, vol. 8, no. 4, 2001, pp. 10–17.
4. B.D. Noble, "Agile Application-Aware Adaptation for Mobility," *Proc. 16th ACM Symp. Operating Systems Principles* (SOSP), ACM Press, 1997, pp. 276–287.
5. J. Flinn et al., "Data Staging on Untrusted Surrogates," *Proc. 2nd* USENIX *Conf. File and Storage Technology* (FAST), USENIX, 2003, pp. 15–28.
6. J. Flinn, S. Park, and M. Satyanarayanan, "Balancing Performance, Energy, and Quality in Pervasive Computing," *Proc. 22nd Int'l Conf. Distributed Computing Systems*, IEEE CS Press, 2002, pp. 217–226.
7. R.K. Balan et al., "Tactics-Based Remote Execution for Mobile Computing," *Proc. 1st Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys), USENIX, 2003, pp. 273–286.
8. J. Flinn and M. Satyanarayanan, "Energy-Aware Adaptation for Mobile Applications," *Proc. 17th ACM Symp. Operating Systems Principles* (SOSP), ACM Press, 1999, pp. 48–63.
9. E. de Lara, D.S. Wallach, and W. Zwaenepoel, "Puppeteer: Component-Based Adaptation for Mobile Computing," *Proc. 3rd USENIX Symp. Internet Technologies and Systems* (USITS 01), USENIX, pp. 159–170.

Rajesh Krishna Balan is a graduate student in the Computer Science Department at Carnegie Mellon University. Contact him at rajesh@cs.cmu.edu.