Mobile Enterprise Act Two: The Props

Jonathan W. Lowe

This column covers the role of emerging technologies in the exchange of spatial information.

Last month, we began a series exploring mobile enterprises. Part two of this three-part series caters to the toy-lovers in our industry. Among the available computing devices, PDAs may be the most intriguing spatial toys. Deceptively slim and light, they have become powerful mini-computers uniting GPS receivers, wireless communications, digital cameras, MPEG players, movie viewers, and, finally, good old workaday maps. Clearly, any industry with field workers stands to increase efficiency and reduce costs by introducing PDA technology into their workforce — don't they? Maybe. Before rushing off to the nearest retailer, stop to consider just how powerful these devices really are and how the entire enterprise must change to accommodate these occasionally connected new members of the spatial club.

The pipedream

Already there are dozens of PDAs to choose from in today's market, and new ones continue to appear — each with a subtle variation in target audience. Sharp's (www.developer.sharpsec.com) Zaurus, for instance, seduces hard-core developers with a Linux-based PDA. Palm (www.palm.com) and Handspring (www.handspring.com) continue to enable user-friendly personal information management on PDAs running their Palm OS. Also using the Palm-OS, Sony's (www.sony.com) Clie offers the highest resolution screen in an effort to corner the entertainment market. Compaq's (www.compaq.com) iPAQ and HP's (www.hp.com) Jornada use the Windows-CE operating system, offering Windows users a look and feel similar to their desktop machines — a siren call to the business user.

These are just a few of the many devices on the U.S. market. Abroad, there are even more, some of which are heading to America, so keep an eye out for the new Toshiba (www.toshiba.com), Sanyo (www.sanyo.com) and Fujitsu-Siemens (www.fujitsu-siemens.com) devices. With so many choices, figuring out which device works best for your enterprise can be challenging.

Selecting by operating system, though, narrows the search immediately. Currently, within the United States, PDAs use only one of three possible operating systems: Linux, Palm OS, and Windows-CE (that is, PocketPC). For simplicity's sake I'll focus on the Compaq iPAQ in this column, in part because all the mapping vendors build software for the iPAQ's Pocket PC operating system, and also because I was able to acquire one for hands-on testing.

Programmer's fantasy. As a programmer, I fantasized wildly about owning a handheld version of the familiar Windows environment on what I naively imagined would be a PDA as functional and powerful as my three-year-old desktop PC. One fateful day, an iPAQ appeared in my office. Poised to rule the world, my new iPAQ docked and ready, I downloaded eMbedded VB (for free) from the Web and promptly programmed my first PDA application — a button that opened a message box that said "Hello World!" My translation of pre-existing programming experience to the new device was a success.

VB, however, is a programming language for Windows 95, 98, NT, and 2000, not Windows CE. As such, VB has similarities to eMbedded VB, but is by no means equally functional. Many familiar VB commands simply don't exist in eMbedded VB. This isn't a cruel joke by Microsoft; it's simply a reflection of the limited hardware environment of a PDA. The specific variations in the two programming languages are critical to programmers, but not appropriate material for a "Net Results" column. What is relevant to any spatial professional, though, is an understanding of the PDA's limited hardware environment because this influences our use of PDAs in the enterprise.

The reality

Although manufacturers tout the advances in PDA processor speeds (in the 200 MHz range), memory (system RAM of 64 MB, system ROM of 16

Glossary

TCP/IP: Transmission Control Protocol/ Internet Protocol

UDP: User Datagram Protocol

VB: Visual Basic

Net Results columnist Jonathan W. Lowe is the owner of Local Knowledge Consulting (Berkeley, California), where he designs and implements spatial Web sites. Lowe can be contacted at info@giswebsite.com.

MB), flash or microdrive storage capacity (2 GB or more), and color displays of 16 million colors, the linchpin of any lightweight and portable system turns out to be its battery power.

When describing the Power PC environment, Microsoft's documentation explains power as follows:

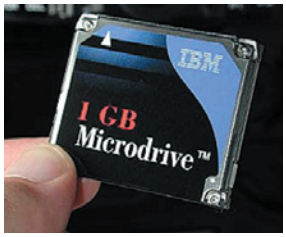
> Because a Pocket PC is portable, battery life is very important. A Pocket PC can run at least 15 hours on its standard battery source.

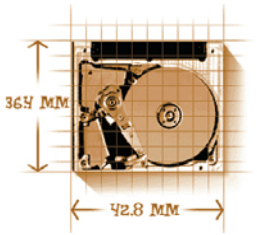
Fifteen hours from a miniscule lithium-ion battery? Amazing! Imagine how much one could rely on a device with such an extended mobile life. Deeper within the programming section, however, we learn more about that impressive 15 hours:

> Power management for a Pocket PC is based on the following assumptions:
>
> - The Pocket PC is used less than two hours per day in bursts from five minutes to one hour.
> - The display is powered 100 percent of the time during use.
> - The device runs less than 10 percent of the time during typical use.
> - The device uses both main batteries and a backup battery.
> - The device has no nonvolatile, writable memory.
> - PC card storage devices that draw appreciable power from internal batteries significantly reduce battery life. To obtain maximum battery life, avoid using PC cards in favor of storage devices that draw little power.
> - Audio playback also consumes power at a relatively fast rate and will greatly reduce battery life.

These specific power management assumptions help explain the optimism of the misleading initial "at least 15 hours" claim. A wily marketer enables the false assumptions of his audience without lying outright — 10 percent of 15 hours is 1.5 hours! No wonder that, when testing mapping software on my iPAQ throughout an afternoon, I ran out of battery power in less than two hours — much sooner than expected. (To be fair, Microsoft's "two hours per day" predictions are probably conservative. Using headphones, I listened continuously to audio files stored in the main memory of my iPAQ until the battery ran out, three hours later.)

Fortunately, users in the field are not benchmarking their PDA's mapping software throughout the afternoon. They have a job to do. So, if real field users only turn on their PDAs for five or ten minutes now and then, the batteries should last all day, even on a brutal 15-hour day, or more than two regular eight-hour days without benefit of a recharge. The question for the mobile spatial enterprise is whether field users will actually limit their PDA use to brief bursts amidst longer stretches of quiescence. For instance, what if they want to listen to digital music on their PDAs while driving between job sites, or





FIGURES 1a and 1b Most popular with photographers, but applicable to storing spatial data, IBM's MicroDrive expands a PDA's storage by a gigabyte of data.

watch a PocketTV video during lunch? Hello battery drain.

Using a corporate PDA as an entertainment center is only one competing draw on mobile battery power. Storage expansion options are also power hogs. When marveling at the ability to store one or 2 GB of data on a flash memory card (what Microsoft refers to above as a "PC card") or microdrive, bear the tradeoffs in mind. Although the complete Tele Atlas North America (www.na.teleatlas.com) dataset for the entire nine county San Francisco Bay Area fit onto my iPAQ's IBM's (www.ibm.com) 1-GB microdrive (See Figures 1a and 1b) with room to spare, the power to view, pan, and zoom its map drained the iPAQ's battery in less than two hours. When main memory is available, using it instead of cards or microdrives will extend battery life.

Convergence. The convergence of technologies on handheld devices is downright incredible, but again, is not without cost to the availability of the overall system. For instance, if the enterprise's field workers require GPS technology, tiny receivers from various vendors clip into most PDAs and enable real-time positioning on the PDA's map display. Some operations may also need to capture digital photos of a job site. To the rescue again, PDAs support microcameras that store digital snapshots in the PDA's memory and use its screen to preview the images. Cameras and GPS receivers both require either their own batteries or a share of the PDA's power. There are workarounds to limited power — such as a recharger cable that plugs into the car's cigarette lighter — but there's always a tradeoff in true mobility and system simplicity.

Weight of wireless. One of the more recent technological additions to the PDA is wireless communications. Though slow (9,600 baud or worse), wireless modems do support wireless data transfer between the PDA and an Internet site. Mapping software vendors address the bandwidth problem by limiting the spatial exchanges to just the edited data. Whether this solution succeeds or not is a topic for part three of this series. Potentially more important to the entire enterprise, however, is the fact that all wireless signals — cell phones, radios, and pagers — occasionally drop their connections. We drive through tunnels, walk between tall buildings, and venture deep into the wilderness beyond the limits of coverage. And, as described earlier, limitations of battery power virtually require that the wireless device stays turned off most of the time, unable to send or receive messages until reactivated. Once an organization decides to commit to mobile wireless devices, the repercussions of the fact that PDAs are only occasionally connected will ripple through the entire system, all the way back to the central server.

Occasionally connected enterprise

Working with distributed mobile devices of limited power and occasional connection changes the whole enterprise's system because it simultaneously extends and erodes messaging capabilities. In the enterprise computing context, messaging refers to the way various parts of an application communicate with each other. Mobile devices extend message delivery to formerly isolated field workers, no matter where they happen to be roaming. However, because of their occasionally connected nature, mobile devices erode the reliability of a successful message delivery. What if the receiving device is turned off during the attempted message transmission? Where does the message go? Is it lost, or does the sending device automatically keep trying? What if the sending device is turned off while still trying to transmit? When successful messaging is mission critical, then guaranteed message delivery and priority-based messaging become key promises of software integrators and additional considerations of enterprise architects.

Messaging. Sometimes, guaranteed message transmission doesn't matter. In some real-time spatial applications, GPS-enabled devices send messages about their position to a central source. For instance, a trucking company may track the locations of its trucks along their delivery routes. If a truck goes through a tunnel while trying to send or even capture its position, the signal may fail — but who cares? Even if only some of the signals transmit successfully, the truck's path can probably be reconstructed with acceptable accuracy, especially if the trucks broadcast their positions every minute or even every five minutes.

On the other hand, what if the message absolutely must go through, and the job isn't over until the sender has absolute certainty of that fact? Military operations and their associated life-and-death urgency offer a good illustration of the need for guaranteed message delivery. What if, after penetrating enemy lines with a laser-powered pointing device, the soldier's job is to transmit the coordinates of his reconnaissance target to a plane flying high overhead. Once the message is delivered, the soldier has a short time to evacuate the area before the plane drops its bombs — the target itself may be semimobile too, so time is of the essence. The soldier needs to know that the message from his or her occasionally connected laser-pointer device has reached the destination — another occasionally connected battlefield mapping device in the plane, and maybe other destinations in the overall military enterprise.

The queue. To achieve guaranteed message delivery between applications running at different times and communicating across networks and systems that may be temporarily offline, developers use message queues. To guarantee message delivery in the military example, the soldier would send messages not directly to the plane, but to a message queue (essen-

tially a holding area) on another machine that is always connected to the network. Likewise, if the plane temporarily loses connectivity, it reads the message from the same message queue when its network connection is reestablished. This way, the message will always eventually arrive at its final destination as long as it first reaches the message queue.

Recognizing that occasionally connected devices introduce challenges to messaging, Microsoft provides a Microsoft Message Queuing service to developers of Windows-CE device applications. Other vendors have parallel support, such as IBM's MQ-series software.

Priority messaging. In the simple example of messaging between a soldier, a plane, and a shared, permanently connected message queue, there is no provision for competition from other participants. In reality, though, hundreds or thousands of machines and people may be beaming messages throughout the enterprise. At the lowest level, messages are composed of bytes that must flow across a network. No matter how high the bandwidth, at some point, there is a limit to the number of simultaneous messages a system can transmit. During periods of high activity, messages may accumulate in one queue before being sent to another. If some messages are more important than others, they can be marked as higher priority and sent first rather than in their order of arrival in the line. Despite the seeming democracy of plain old zeros and ones, some packets get box seats and others only standing room. Marking an email "urgent" uses the same concept. The ranking of simultaneous transmissions relative to one another, called priority-based messaging, is another key consideration of large enterprises, the military again being a good example.

Protocols. Yet another consideration that mobile wireless devices and guaranteed message delivery impose on the design of an enterprise system is protocol selection. Transmission protocols are predefined steps for breaking up, identifying, and sending information across a network and then reconstituting and confirming reception on the receiving end. Both sender and receiver know the rules, and so can make sense of the transmission. A commonly used protocol pairing, TCP/IP involves complex validation and confirmation steps. As a result, transmitting a GPS location from a truck back to a central office using TCP/IP can take as long as 45 seconds to complete, but the received message is guaranteed to be identical to its source. For a truck, TCP/IP transmission might be a fine choice, but would a 45-second transmission interval adequately track a supersonic jet?

Another protocol, UDP, doesn't confirm reception. It merely fires off the message, indiscriminate of listener availability. Unlike TCP/IP, UDP messages take only five seconds to completely send. To track a jet fighter whose position changes radically in a short time, UDP transmissions every five seconds may be a better choice than TCP/IP. What we lose in reliability of delivery balances out in the sheer number of attempted transmissions. Even if only half its positional messages get through, the picture of where that jet is going is more accurate with the shorter transmission interval of the UDP protocol.

The mouse that roared

It's hard to believe that a sleek little toy like a PDA could change the architecture of an enterprise so thoroughly, but integrated applications increasingly involve small mobile devices such as PDAs. When envisioning how your enterprise's field workers will become empowered by mobile PDAs, don't forget that the hardware itself, though powerful, has its limits. And those limits in turn may change the underlying system architecture of the whole enterprise, particularly of the messaging between the ever more distributed universe of occasionally connected computing devices.