A Roadmap to Wireless: The State of the Technology

By J. Dale Gonzalez
VP, Wireless Devices
Air2Web

Mobile commerce is anticipated to be the next business revolution. In an effort to take their businesses wireless, organizations face a number of critical, strategic decisions and a host of confusing options. The right strategy for any organization depends on its response to the following questions:

1. How much market reach do I want and which technologies must I support as a result?
2. What is my objective for offering wireless access? Are interactivity and/or m-commerce important to my business?
3. Is there a benefit to going wireless without my brand attached?
4. How much control do I need over my wireless application's features and functions?
5. How suitable is my existing application for wireless use? How much redesign will be required?
6. How well prepared am I internally to develop and host wireless applications?

A Roadmap to Wireless: The State of the Technology provides the context to determine which wireless strategy best suits an organization's needs. Included are a summary of the industry forces shaping wireless technology, guidelines for creating successful wireless applications internally, detailed explanations of the chief device technologies available, and an analysis of deployment alternatives in the market.

A Roadmap to Wireless: The State of the Technology

Bringing computing power to a wireless device opens an extensive new market for commerce, particularly as cellular telephones become equipped with browsers as a standard feature. Industry sources estimate that next year more wireless devices will be sold than PCs and landline phones combined, and within two years there will be 329 million wireless Internet subscribers worldwide.[1]

"Revenues for wireless e-commerce are anticipated to reach $83 billion by 2003, an increase of 222 percent over this year."

The Industry Standard

Taking an application wireless involves much more than simply porting an existing Internet site to a browser-enabled phone. Mobile applications differ greatly from ones created for PCs and laptops because their users have a different set of needs and expectations. Users want wireless transactions that are available at any time, are easy-to-use, tailored specifically for their requirements, and executable in minutes. In order to deliver on user expectations,

[1] *International Data Corporation*

developers must create a comprehensive application specifically targeted for wireless devices, rather than just adding wireless accessibility to an existing Web site.

"Wireless deployment and support can be so complex that it makes solving Rubik's Cube blindfolded look easy. Companies that want to roll out wireless technology … need to stay up with changing standards, as well as the different technologies used in carrier networks."

InformationWeek

Compounding the challenges of wireless development is the lack of an industry standard. Several industry groups—positioning themselves as the de facto standard—are battling for control: the Wireless Application Protocol (WAP) Forum, created by Phone.com and other high-end market players; the World Wide Web Consortium (W3C); and Japan's NTT DoCoMo. Because wireless is a global phenomenon, the emergence of a single dominant standard that supports all markets seems unlikely in the near term.

In addition, there is an ongoing debate over technology development, including whether the acceptance of WAP will surpass that of Personal Digital Assistant (PDA) devices. Industry leaders also are trying to predict how long Short Message Service (SMS) will continue to lead the wireless marketplace.

"The list of potential uses for wireless communications in the future is as endless as we dare to dream. The fantastic range of possibilities will require far reaching technological innovation to make them a reality: innovation that is already well underway."

The GSM Association

The reality of wireless today is that the easiest way to deploy applications—via wireless modem technology—represents only a fraction of the market. In terms of deployment cost, the most expensive applications are centric to the largest market segment, SMS phones—which, at 64 percent share of the US market, cannot be ignored despite the complexities involved in supporting the devices.

Best Practices for Wireless Implementation

The most effective wireless applications are those that marry users' demands with the technology options available on a particular device. The guidelines below will help developers deliver a reliable and secure solution that is easy to use and optimizes the functionality of a wireless device.

■ **Create new applications** – Mobile applications should be designed specifically for wireless devices, rather than forced to fit an existing Internet application. The best ones are those that extend, not replace, the Internet applications with which they are associated. Wireless devices should not be considered alternatives to a PC but rather a completely new means of interacting with an application. The developer should expect that certain tasks will only be performed on the PC and some will only be performed using the wireless device. It will be the very rare feature that is available from either access mechanism.

■ **Develop interactive applications** – One-way, or "paging style" models don't create high-value mobile applications. Interactive applications allow end-users to ask for a service when the service is

most valuable to them. A truly valuable wireless solution not only must incorporate notifications sent by the server; it also must supply users with functionality on demand. Supporting both server- and user-initiated transactions ensures that users have access to the features they want when they need them, instead of according to a predefined schedule.

■ **Deliver user-friendly operations** – Although it can be extremely challenging given the nature of wireless devices, it is critical that developers provide an easy user experience. Wireless devices are the ultimate constrained computing devices with limited CPU, memory, and battery life plus awkward screens and keypads. Wireless networks are strangled by low bandwidth and high latency, as well as unpredictable availability and stability. Developers should look for style guides, such as those provided by Phone.com, to get a head start.

■ **Remember your audience** – Users of wireless devices may or may not be the Web-savvy group that developers are accustomed to. There are more cellular phone users today than there are people with Internet access. Developers cannot assume that Web conventions such as hyperlinks will be immediately intuitive to an end-user with only occasional exposure to a Web brower. Applications must be as simple to use as possible because if the learning curve is too steep, users will not continue to use them.

■ **Allow user personalization** – Given the limitations of device screen real estate and the costs associated with wireless access, functionality must be targeted to each specific person using the application. The greatest uptake stems from giving end-users the

ability to personalize application features to their tastes, lifestyles and comfort levels.

■ **Leverage device characteristics and be aware of device limitations** – The wireless landscape is populated with a large variety of devices, and the great differences between them make supporting the least common denominator impractical. Developers must tailor applications to accomodate the specific devices that will access those applications. In addition, given the limitations of the environment, developers should strive to make the best use of device capabilities. For example, many wireless devices are first and foremost, audio devices; making effective use of audio can significantly enhance a wireless application. Also, developers must consider that wireless device manufacturers support the wireless markup standards differently, and screen sizes vary widely from one device to the next. Finally, applications must be thoroughly tested on every device supported.

■ **Consider location sensitivity** – One primary advantage of wireless devices over PCs is that users can access applications no matter where they are. Truly compelling wireless applications incorporate location awareness where it makes sense. Of course, developers should be aware that connectivity to the application servers may or may not be available depending on the user's current location. In order to develop applications that are as dynamic as possible, developers must consider how the transient nature of network availability impacts device usage, including the potential for break down outside of available cellular network territories or in remote locations.

■ **Eliminate hierarchical navigation** – Applications should be flat. Studies suggested that as many as 50 percent of users are lost with every click.

■ **Be sensitive to security** – Applications that allow users to conduct transactions must reflect careful consideration of security implications. Wireless applications place greater demands on security than traditional applications because the devices are more easily stolen than PCs. The differences in direct support for encryption from technology to technology, and the lack of client-side support for security add to the security burden. Furthermore, since the link between client and server can be severed at any time, extra steps must be taken to ensure transaction integrity.

■ **Avoid screen scraping** – Applications based on screen scraping can be very fragile. Changes to the underlying Web pages can have an unintended and unpredictable impact on the wireless application overlay. Techniques such as comment tags and tag IDs, designed to minimize the fragility of the solution, place added burden on the Web development group. The techniques often aren't directly supported by Web development tools, and they certainly are not foolproof. Furthermore, the interesting aspects of a Web site from a wireless perspective are its dynamic elements, which do not originate as HTML. Developers should consider going directly at the source of the content rather than its rendered form. The most compelling applications are those that have been designed specifically for wireless deployment. Beginning with a Web site may constrain the design of the wireless application in an unintended way. Instead, developers should view the Web site and the wireless application as distinct elements of the overall solution. One enhances and extends the other, but neither is built directly on top of the other.

Wireless Device Techniques

There are primarily four techniques a developer can use to get information to wireless device users. Each has its technological advantages and challenges, but more importantly, each has a market payoff directly proportional to the size of the investment required to support it. Of course, wireless applications intended to reach the widest addressable market must support all viable technologies.

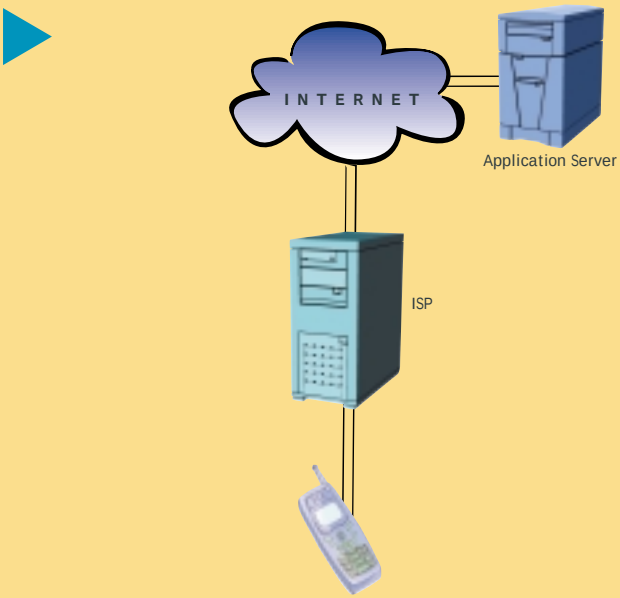
Number of U.S. Users[2]	Wireless Device Technology
Less than 15,000	**Smart Phones** (e.g., Qualcomm PDQ, Nokia 9000 Series)
160,000	**Personal Digital Assistants** (e.g., Palm VII)
4,000,000	**Web-enabled (WAP) Phones** (e.g., Samsung SCH-3500, Ericsson MC 218)
65,000,000	**SMS-enabled Phones** (virtually all digital cell phones)

[2]*The Yankee Group*

Smart Phones

From a developer's perspective, by far the easiest wireless device technology to support is wireless modem technology. Smart Phones—such as the Qualcomm PDQ and Nokia 9000—rely on wireless modem technology to create a standard Point-to-Point Protocol (PPP) connection to the Internet, using the cellular radio as a modem. Working much like a laptop computer, the smart phone uses its modem to dial an Internet Service Provider (ISP), gain access to the Internet, and connect to the server. The devices use Web standards, but only support a limited subset of HTML.

The device connects to the application in the same way any dial-up user would, so the differences between carrier networks are masked. Unlike a wireline dial-up user, the connection to the server may be severed at any time. Applications built to support smart phones should consider this aspect of wireless connectivity.



Though wireless modem technology offers simple implementation, the market for smart phones is severely limited and unlikely to grow significantly because the devices are awkward, expensive and slow (9600 bps). In situations where the audience is controlled and organizations can require the use of smart phones, wireless modem technology offers a very fast implementation. It may also prove useful for prototyping.

From a security perspective, most smart phones support the Web standard, SSL, but they have limited or no support for storing private keys on the device.

PROS

- No carrier relationships necessary
- Web standards utilized, including an HTML subset
- Some graphics support

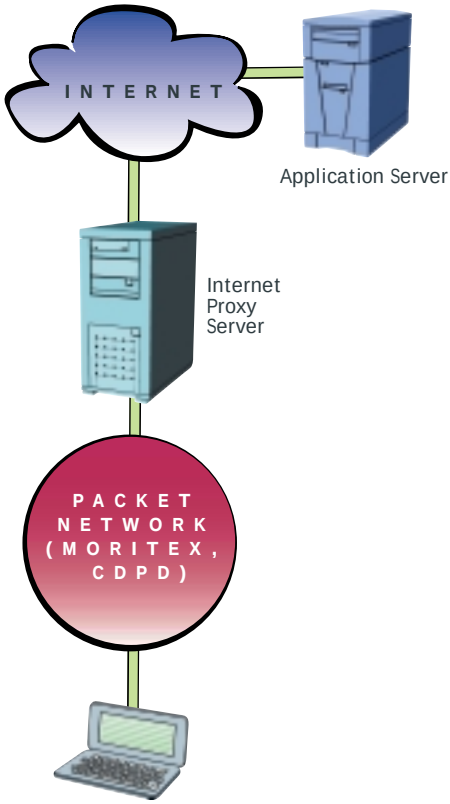
CONS

- Very small market
- Highly variable HTML support and device characteristics
- No direct support for push content or audio
- Slow transmission speeds
- No support for storing private keys, and therefore no direct support for non-repudiation

Personal Digital Assistants

Applications built to support PDAs and some 2-way pagers are the second easiest wireless applications to build. These devices connect wirelessly to the Internet using data-only networks. The networks—called packet networks—slice the content being transmitted into small packets of information and provide each with a unique identification and destination address. The diced information is sent out over multiple channels, providing better throughput than technology that utilizes a single voice channel. Identification and sequencing information on each packet lets the data be reassembled in proper sequence by the receiver.

For PDAs there are two primary packet networks to be supported: AT&T's CDPD network and BellSouth's Mobitex network. Connection to the networks is made using a proxy server run by a wireless Internet Service Provider (ISP), such as Palm.net or GoAmerica. Developers are abstracted from the specific details of the two networks, and the devices support a subset of HTML. The complicating factor is Palm's requirement to download the first HTML in a binary form—called a Palm Query Application—to the device. Developers must allow for the download dialog on their sites and the download process itself.



PDAs and 2-way pagers have fairly good input mechanisms and decent displays that support graphics features, including some color. They also have shorter initial connection times and offer better throughput. However, both the devices and the data transmission rates are expensive. In addition, nationwide coverage for each network is spotty and global coverage is even worse. This lack of complete coverage makes testing difficult.

From a security standpoint, most networks require that SSL traffic be unencrypted on the proxy server and re-encrypted in a wireless form before transmission to the device. This means that for an instant in time the data is in an unencrypted form on the proxy server. Because this period of time is so short, and because of the safeguards implemented on the proxy servers, most developers consider the security threat to be minimal. As with wireless modem technology, there is no client-side support for private keys.

PROS

- Steadily growing market
- Short initial connect times
- Web standards utilized, including HTML
- No carrier relationships
- Some graphics support
- Good throughput

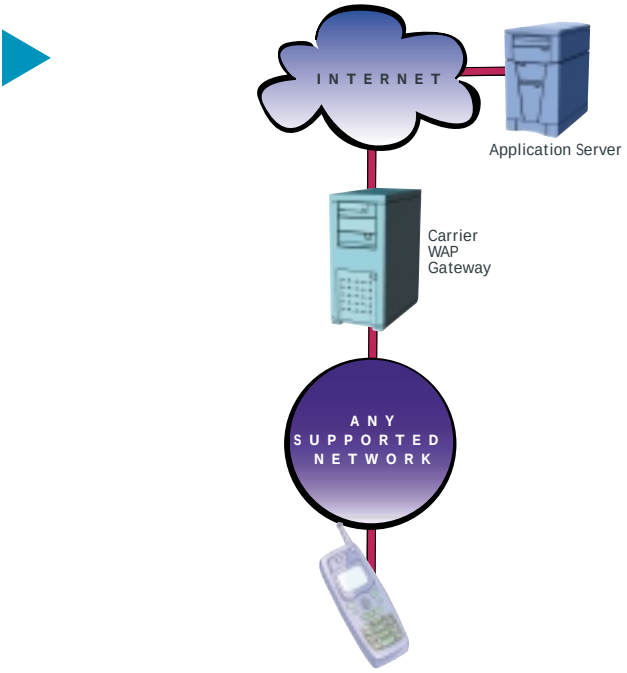
CONS

- Incomplete network coverage
- Potential market decline in favor of WAP
- Device prices are high
- Data delivery costs are high
- Some implementations require download and synchronization
- No consistent support for push content
- No audio capabilities
- No support for storing private keys, and therefore no direct support for non-repudiation

Web (WAP) Phones

A technology receiving a great deal of attention today is the Web-enabled phone. Introduced within the last three years, there are already an estimated 4 million in use in the U.S. and 12 million in Japan. With the growth predicted, there is no question a wireless application benefits from supporting this technology.

The primary U.S. and European standard for Web-enabled phones is WAP. WAP specifies network server, phone software, and the communication protocol between the two. The protocol is designed to use any wireless network as a bearer for the data, including CDPD, CDMA, GSM, PDC, Mobitex, and more. In Asia, I-Mode is the dominant standard. Typically, programming is performed with WML or HDML—variants of XML, which is an easy mark-up language for existing Web developers to learn.



Though simple in concept, WAP is challenging to work with because the hardware designed to implement WAP lags behind the current version of the standard. For example, the microbrowser firmware that resides on each device is hardwired into the phone. Users in the U.S. cycle their phones at an average rate of every two years. Consequently, developers must support every version of microbrowser employed today and in the foreseeable future. In addition, the previously mentioned two markup languages coupled with the two variants used in the Asian markets force developers to learn at least four markup language "dialects" in order to create a global application. The application must respond with the appropriate markup language based on the network and the version of browser resident on the phone, so developers must maintain an understanding of the differences between all of the versions available in the market, as well as those currently in the standards bodies.

In addition, the devices themselves present challenges to developers. WAP phones are tiny devices with slow and faulty connections to the Internet, a weak user input mechanism, and only two buttons to control navigation. The way the application behaves, the keys used to interact with the application, and the screen real estate available differ widely from one device to the next. WAP was designed to support user-initiated (pull) interaction. Server pushed interactions require additional support from the carrier, and adoption varies from carrier to carrier. And finally, without an extensive investment in text-to-speech and related telephony infrastructure, there is no way to provide audio support, which is the device's best capability.

WAP devices support security in a manner similar to PDAs, and they also translate to and from SSL to a wireless protocol. Again, the devices themselves do not support private keys.

PROS

- Growing worldwide market
- Formatable content
- Some graphics support
- Good and improving throughput
- Some support for push content
- Some client-side programming support

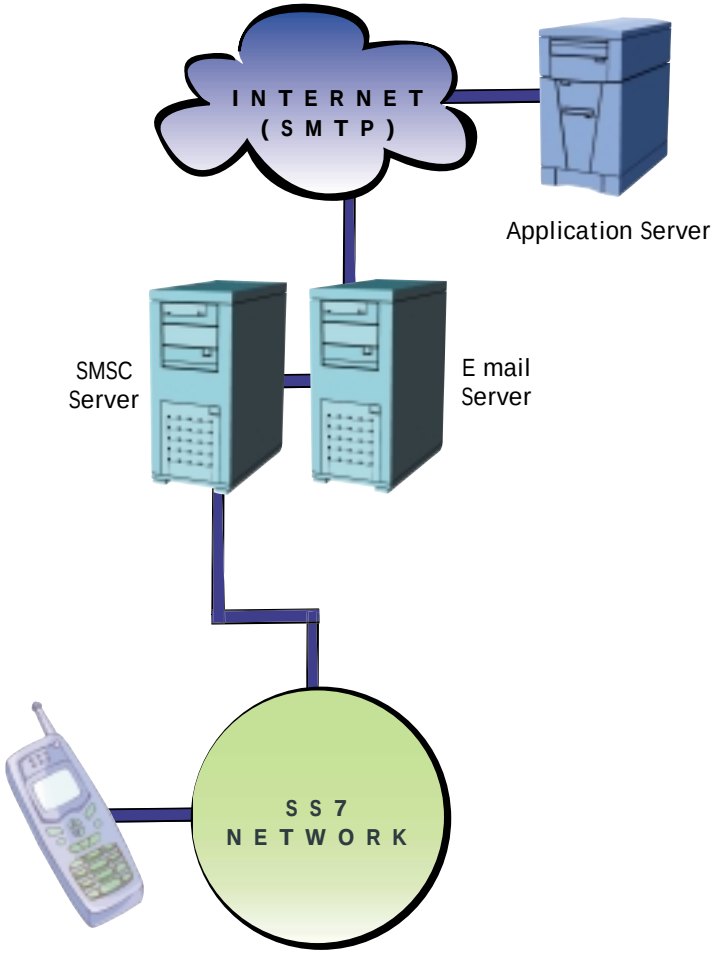
CONS

- Burned-in browser creates version and form factor variability
- Slow carrier adoption of network server upgrades mandates support of multiple gateway versions
- Multiple mark-up language variants
- Challenging input mechanism
- Inconsistent support for push
- No direct support for audio
- No support for storing private keys, and therefore, no direct support for non-repudiation

SMS Phones

By far the largest segment of the wireless device market is digital cell phones equipped with Short Messaging Service (SMS) capability. SMS is the most primitive technology available, with the most severe operational limitations and the greatest development investment. However, analysts predict that given its current market dominance relative to other wireless data technologies, SMS will continue to be used at least until the year 2005. A wireless strategy that doesn't embrace SMS cuts off an organization from millions of existing and long-term mobile users.

SMS takes advantage of the signaling channel that a cellular phone uses to communicate with the rest of the wireless network. The cellular network element responsible for handling message traffic is called a short messaging service center (SMSC). The SMSC is responsible for routing a message to a cellular phone as well as storing messages while the phone is out of coverage or turned off. Although cellular networks are private and require special permission and equipment to access directly, SMSCs are often connected to the Internet via Web front ends or email gateways. When using one of these email gateways, the protocol for message format is simple, plain text, and delivery is usually over SMTP. Developers need to be aware that there are limits to the message length (typically between 100 and 160 characters), and those limits vary from carrier to carrier.



SMS wireless applications are cumbersome to program because each carrier has its own limits on message size, its own gateway address, and its own conventions for generating the email address associated with a particular cell phone. In order to derive an email address, the programmer must know the device phone number, the carrier, the carrier gateway, and the semantics the carrier uses to convert from a phone number to an email address. The most practical way to obtain this information is to ask the user to provide the device email address

directly. However, users typically only know their device phone number and carrier. The email address is often unknown even to the seller of the device. Applications that ask users to provide their phones' email addresses create a negative enrollment experience, and uptake suffers. Other solutions for deriving the device email address involve the establishment of extensive provisioning logic, significantly increasing the development effort.

Another difficulty associated with SMS is that most carriers in the United States only support one-way communication from the Internet server to the device. Most European GSM networks allow cellular phones to both receive and send SMS messages. However, outside of Europe the availability of GSM is very limited, particularly if SMTP is the delivery protocol. Developers seeking to create user-initiated SMS applications must limit this capability to a very small audience or invest considerable resources in telephony infrastructure and logic.

Similar to WAP, the SMS technology ignores most devices' best feature—audio. In order to add audio to wireless applications, a developer must integrate with telephony hardware supporting text-to-speech, .wav files and streaming audio. If 2-way interaction is desired, this telephony infrastructure must also include DTMF tone detection, as well as voice recognition.

Generally speaking, SMS phones have no direct support for security. All traffic is transmitted in the clear to and from the servers and over the air. In certain circumstances on GSM networks, it is possible to add security support to the device, but support of this kind is uncommon in the United States.

PROS

- Largest existing customer base
- Simple protocol for message format and delivery
- Good built-in model for push applications

CONS

- Extensive tracking of carrier information or direct carrier relationships required
- Limited service capabilities
- No direct support for audio
- High degree of latency
- Primarily one-way, push-only
- Little control over format and delivery time
- Very weak security

A Look Ahead

Wireless technology is evolving quickly with many major initiatives underway that will impact both transmission speeds and device programming options. When developing a wireless application, programmers should consider these and other new technologies that are likely to emerge.

■ **SIM Toolkit** – is an environment that allows SMS messages to interact with the phone's operating system in a common way. It supports over-the-air application downloads and is supported by a number of European cellular phone manufacturers and networks—but notably, not Nokia whose competing technology, Smart SMS, was co-developed with Motorola. SIM Toolkit presents many challenges. For one, it must be supported by the handset and the carrier, neither of which are available in the U.S. In addition, with the ascendance of WAP, SIM Toolkit's future is unclear. Some analysts believe WAP makes SIM Toolkit unnecessary, but others believe the two technologies are complementary.

■ **Embedded Java** – includes two main initiatives currently underway: J2ME and MxEx. Micro edition JVMs are available for PDAs and some pagers already, and several phone manufacturers are about to release devices with built-in Java support. At this time, all existing Micro JVMs are in early stages and will probably undergo significant changes. Furthermore, none of the cellular phones with JVMs are

readily available. Once this technology matures, however, it will provide a very rich client-side computing environment already familiar to many developers.

■ **Bluetooth** – a technology garnering a great deal of attention, is basically a non line-of-site replacement for the Infrared Data Association (IrDA) standard. Bluetooth eliminates the need to align devices carefully within short range. Although it is not strictly a wide area wireless data technology, Bluetooth may make it possible for devices such as PDAs to leverage the wireless capabilities resident in cell phones and eliminate the need for compromise devices like the Palm VII, Neopoint NP1000 and the Nokia communicator. In this scenario, large screen and input would remain the province of the PDA, whereas wireless connectivity would belong to the cellular phone. The two devices would cooperate to provide the user with a wireless data application.

■ **GPRS** – is a packet data overlay for GSM networks similar to what CDPD provides for analog networks. GPRS networks are expected to provide theoretical throughput up to 171.2 kilobits per second. Actual throughput will probably be somewhat lower and will vary from carrier to carrier.

■ **EDGE** – builds on the packet capability introduced by GPRS. By changing the mechanism used for frequency modulation, throughput up to 384 kilobits per second will be possible. EDGE is considered a transition technology from GPRS to 3G.

■ **3G** – is the collective term used to refer to any number of proposals for re-engineering the wireless networks to make them more data-capable. It is expected that transmission speeds up to two megabits per second will be achievable.

Alternatives to Internal Wireless Development

There are a number of alternatives to internal development of wireless applications. The intense competition and emerging nature of the market have served to cloud the choices available to an organization developing, a wireless strategy. This section provides a brief description of the major wireless deployment options and an assessment of their strengths and weaknesses.

Aggregrators

Aggregators purchase generic content from content providers, package it and resell it to carriers. The information is then private-labeled by the carriers. Aggregators pay the provider for the content; and carriers generally position the content as a value-added service.

From the content provider's perspective, aggregators deliver the following benefits:

- An easy means of distributing their content across all carriers with which the aggregator has a relationship.
- A revenue generating option, since the aggregator pays for the content
- Very fast time to market

However, because the aggregator packages the content generically and the carrier private-labels it, this strategy effectively takes an organization wireless anonymously. As a result, the content provider does not receive any increase in brand recognition or credit for providing users with wireless functionality. Other disadvantages to the content provider include:

- Lost opportunity to capitalize on advertising and promotional revenue
- Lost opportunity to connect directly with customers
- Push only (e.g., just alerts); no opportunity to interact with the customer
- Limited market access because not all carriers are supported by any one aggregator
- Only devices compatible with the carrier's networks are supported
- Less compelling applications because the content is typically not customizable by the end-user

Wireless Portals

Similar to Internet portals, wireless portals are an entry point for wireless access where users can choose wireless content options covering a variety of subjects. The content comes from a number of different suppliers, but is private-labeled by the portal. Many of the major Internet portals have wireless extensions, and the draw of their powerhouse brands gives them the opportunity to charge content providers for the distribution channel. Portals provide the following benefits:

- The opportunity to align with an Internet powerhouse
- A fast means of getting content to the wireless user
- Broader market access, since portals typically support multiple carriers

The following are disadvantages associated with a portal partnership strategy:

- Lost opportunity to associate brand with content
- Lost opportunity to connect directly with customers
- Primarily push only (e.g., just alerts); applications are typically not interactive
- Content scheduling is limited, placing restrictions on the user
- Users must provide information at enrollment that is difficult to obtain, making the process unpleasant and impacting usage
- No inherent revenue-generating opportunity

Wireless Application Service Providers (WASPs)

WASPs accept content in the form of data feeds from a provider and couple it with business logic hosted by the WASP to produce a wireless application. The resulting application typically carries the content provider's brand. As is typical of the ASP model, content providers pay for the service on a usage or transaction basis. The obvious benefit to WASPs over aggregators and portals is the content provider's ability to associate its wireless offerings with its own brand, thereby extending its consumer reach.

Providers do, however, lose control over the wireless application's features because they only provide a data feed—the WASP builds and deploys the application. Another disadvantage of WASPs is their scope. Most do not have the technology to access mobile users independent of carrier, network and device, so market reach may be limited. Many WASPs also don't have the infrastructure to support interactivity across all devices, so the content provider is restricted to push support across most devices with limited opportunity for m-commerce, or with m-commerce limited to a select number of devices. When m-commerce is provided, the content provider must ensure that the WASP offers a secure environment, and security support varies widely from WASP to WASP.

Wireless Application Platforms

E-commerce solutions went through phases of innovation beginning with custom development projects that became "one size fits all" packaged applications and culminating with the emergence of platforms such as Ariba and Commerce One. The platforms provide core functionality with interfaces for customer-specific business logic and capabilities for tight integration with existing information systems. The same evolution is occurring with mobile commerce solutions. Companies such as Air2Web are extending the WASP model to offer comprehensive platforms for wireless application development and delivery.

Like WASPS, platforms enable businesses to associate wireless offerings with their brands while shielding them from the complexities of developing their own solutions. However, platforms have the added benefit of feature/function control. They enable businesses to go beyond a wireless Web strategy to truly integrate mobile access—including secure m-commerce — into their enterprise information systems. They also provide the infrastructure to support fully interactive multimedia applications independent of carrier, network or device.

As a platform provider, Air2Web delivers to businesses seeking a wireless strategy:

- Full control over their brand
- Full control over their wireless applications, incorporating m-commerce
- True integration of mobile access into the Enterprise Information System
- Access to the entire mobile community
- Full interactive multimedia capability
- Secure transactions across all devices

Of course, organizations that don't need a full-fledged infrastructure for wireless application deployment (e.g., they can dictate device types to users, are following a niche strategy, etc.) may find the services of a platform provider to be more than they need. In addition, time to market can be slightly longer than with a WASP because the WASP is deploying "cookie-cutter" applications that are not integrated with existing enterprise information systems.

Conclusion

The promise of wireless technology cannot be ignored, but organizations pursuing a wireless strategy face a number of challenges. The industry itself is in flux with no established standard. Wireless device limitations mandate the establishment of a core competency in wireless application design, and the state of the technology forces a tradeoff between limiting the investment in a solution and reaching the largest addressable market. Both internal development strategies and purchased solutions offer a host of options. Some are simple to implement and adopt, but provide limited payoff. Others require a greater investment but deliver a higher reward.

With this roadmap, organizations can effectively navigate the wireless landscape to determine the most appropriate strategy for their wireless objectives, as well as the most effective means for deploying that strategy.

1. How much market reach do I want and which technologies must I support as a result?
2. What is my objective for offering wireless access? Is interactivity and/or m-commerce important to my business?
3. Is there a benefit to going wireless without my brand attached?
4. How much control do I need over my wireless application's features and functions?
5. How suitable is my existing application for wireless use? How much redesign will be required?
6. How well prepared am I internally to develop and host wireless applications?

Air2Web
1230 Peachtree St., NE
Promenade II
Suite 1200
Atlanta, GA 30309
404.815.7707 tel
404.815.7708 fax

www.air2web.com