



Wireless Data Communications

The Considerations

Copyright © 1995 by Motorola, Inc. All rights reserved.

™ and Motorola are registered trademarks and AirMobile, CelTAC, DataTAC, Envoy, Marco, and PowerSave are trademarks of Motorola, Inc.

ARDIS is a trademark of ARDIS Company. AT&T PersonLink is a trademark of American Telephone & Telegraph. General Magic is a trademark of General Magic, Inc. Mobitex is a trademark of Telia AB (or Televerket). Newton is a trademark of Apple Computer, Inc. RadioMail is a trademark of RadioMail Corporation. SprintNet is a trademark of Sprint, Inc. Windows is a trademark of Microsoft Corporation. Other products and services mentioned in this document are identified by the trademarks or service marks of their respective companies or organizations.

No part of this publication, or any software included with it, may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, including photocopying, electronic, mechanical, recording or otherwise, without the prior written permission of the copyright holder. Motorola, Inc. provides this document as is, without warranty of any kind either expressed or implied including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Motorola, Inc. may make changes of improvements in the equipment, software, or specifications described in this document at any time and without notice. These changes may be incorporated in new releases of this document.

This document may contain technical inaccuracies or typographical errors. Motorola, Inc. and its subsidiaries waive responsibility for any labor, materials, or costs incurred by any person or party as a result of using this document. Motorola, Inc. and any of its subsidiaries or other affiliates shall not be liable for any damages (including, but not limited to, consequential, indirect or incidental, special damages, or loss of profits or data) even if they were foreseeable and Motorola, Inc. has been informed of their potential occurrence arising out of or in connection with this document or its use.

Computer Software Copyrights

The Motorola products described in this manual include copyrighted Motorola computer programs stored in semiconductor memories or other media. Laws in the United States and other countries preserve for Motorola certain rights for copyrighted computer programs, including the exclusive right to copy or reproduce in any form the copyrighted computer program. Accordingly, any copyrighted Motorola computer programs contained in the Motorola products described in this manual may not be copied without the express written permission of Motorola, Inc. Furthermore, the purchase of Motorola products shall not be deemed to grant either directly or by implication, estoppel, or otherwise, any license under the copyrights, patents or patent applications of Motorola, except for the normal non-exclusive, royalty-free license to use that arises by operation of laws in the sale of a product.

U.S. Government Restricted Rights

For the United States Government, use, duplication, or disclosure of SOFTWARE and its associated documentation is subject to “restricted rights” as set forth in FAR 52.227-19 (c)(1) and (2), unless being provided to the Department of Defense. If being provided to the Department of Defense, use, duplication, or disclosure of SOFTWARE and its associated documentation is subject to “restricted rights” as set forth in DFARS 252.227.7013(c)(1)(ii) (Oct. 1988), if applicable.

Title: Wireless Data Communications: The Considerations

Document Number: 6804110L99-O

First Edition: September 1995

Printed in the United States

Motorola, Inc.

Wireless Data Group

19807 Northcreek Parkway North

Bothell, WA 98011-8214 U.S.A.

Telephone: (206) 487-1234

Fax: (206) 483-3400

Contents

Wireless Data: The Considerations 1

General Feasibility Considerations	2
So Why Would I Buy This?	2
To Connect or not to Connect	4
Device and Network Considerations	5
Device Types	5
Network Types	6
Battery Life	10
Design Considerations	10
Middleware and Tools	11
Limitation of Data Transmitted	11
Challenges for Packet Data Networks	14
Security	17
User Interface	18
Protocols	18
Getting Started	22
Development Environment	22
Development Tools	23
Network Operator Relationships	23
Development and Testing	23
Release	24
Developer Program Contacts	25

Wireless Data: The Considerations

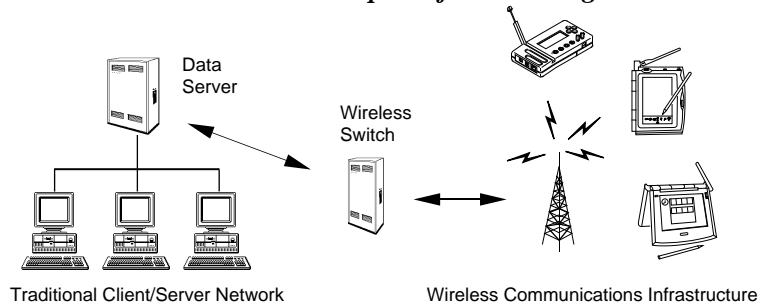
Potential users of wireless data applications are looking for usable solutions to their data communications needs. If you're thinking of developing a wireless data application, you'll need to also consider the hardware it runs on, the network services required, and the sales channel through which you'll market your product. Now is the time to start thinking in terms of total wireless solutions and how your application might fit into them.

Motorola believes that wireless data applications should be simple—simple for the end-user to use, simple for the existing enterprise network to accommodate, and simple for the application developer to develop.

End-users. Users shouldn't have to do anything special, other than turning on the radio and in some cases choosing the wireless option, to take advantage of an application's wireless capabilities. In general, wireless operations should be transparent to the user, although for some applications users will want to be able to select whether operations such as a file transfers will be performed wirelessly or using a wireline connection.

Enterprise network. Ideally, wireless applications should be device-independent, so that they can be run on all devices supported by the existing networks without modification. This independence will accommodate differences in users' preferences and changes in market conditions and company strategies. Wireless devices should ultimately appear to be just another node on enterprise networks, expanding the range of solutions available to end-users and allowing their needs to mature and expand, as illustrated in the following figure:

Wireless communications as part of the existing network



Application developer. Connectivity tools have reduced the need for applications developers to understand wireless integration and communication protocol issues.

However, in order to develop efficient wireless applications, developers still need to be aware of the basic nature of wireless networks and the effect that lower bandwidth, higher cost structure, coverage availability, and network congestion have on application performance and customer appeal. This document addresses those issues particular to two-way wireless data communications that the application developer will need to consider prior to developing a wireless application.

General Feasibility Considerations

If you're trying to determine whether to develop a wireless data application, or whether to adapt an existing application to be used wirelessly, you'll need to decide whether your application can transmit the required data wirelessly in a cost-effective manner.

The following are some of the broad issues to consider.

So Why Would I Buy This?

With current technology, it appears to be more expensive to transmit data over the radio waves than over a wireline connection. So the first question to ask is, why would an application user want to do it?

Wireless data transmissions can totally change the way an application is used. It can change the way people work, and can enable new functions and services. A developer needs to consider the benefits afforded by a particular wireless data application. Can those benefits be translated directly into profits or savings, for example, more sales calls per day? Are the benefits more a matter of convenience, such as enabling managers to stay in touch with their home offices, no matter where they are?

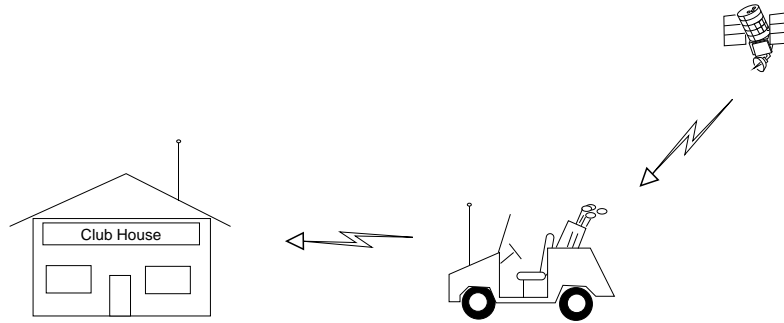
Don't forget to consider the residual benefits. For example, a national security company has installed a system that sends alarm notifications from Motorola radio packet modems located at client sites over the wireless data network and back to a central monitoring station. The primary benefit of the system is enhanced security; alarm notifications can be transmitted even when telephone wires are cut. A residual benefit is cost savings; the company has discovered that, because of the amount of data transmitted, it's actually cheaper to use wireless than wireline communications. For their supervised service, calls are periodically made to the security panel to verify integrity. The company was able to recoup installation costs in 36 months, just by switching from wireline to wireless services to make these calls.

But the tremendous benefits offered by wireless data communications must be weighed against the ongoing cost of network service. Consider how critical it is that the mobile worker transmits data immediately. How often does the data need to be transmitted? If it is only necessary to update a database at the end of the day, perhaps wireless data transmissions would be difficult to justify.

Let's look at a wireless application that would help a ranger at a golf course, whose job is to quicken the pace of golfers around the course. The application would be used to track the position of golfers on the course by placing mobile devices with GPS capabilities on golf carts. The devices would send messages back to the station every 15 minutes during a 12-hour day to indicate their location. If 27 carts were on the course at any given time and each message cost \$.10 to send, it would cost around \$130 a day to run the application. Since the data

would not eliminate the job of the ranger, the benefits gained would not warrant the cost of gathering the data and providing the equipment.

What's it worth?



To Connect or not to Connect

Data can be transmitted using either connected or connectionless (wireline or wireless) communications. An example of connection-oriented communications is the standard wireline telephone modem. When users dial into services or access computers via wireline modems, they are establishing continuous connections or “sessions.” While the connections last, users are billed for each minute of “connect time,” just as they are for voice calls. This is also true for some wireless communication technologies, such as circuit-switched cellular.

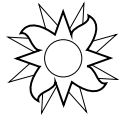
However, with connectionless packet data communications, information is transmitted as packets of data, which is similar to sending telegrams (datagrams). Users pay for the amount of data (the number of packets) sent, rather than for connect time.

Generally speaking, it is cheaper to send small bursts of messages using wireless packet data communications and to save the large file transfers for wireline connections. If you’re developing a wireless application, you’ll need to optimize applications for bursty transmissions and a narrow bandwidth. Consider what data is time-sensitive to users when away from the office and send, or allow users to request, only that data.

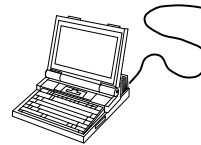
Often users will want to be able to choose between wireless and wireline communications. For example, they might want to choose whether to receive e-mail using a wired or wireless method, depending on the size of the message, whether or not an attachment is included, and the availability of the networks.

Members of a sales force might also choose to receive pricing updates wirelessly, whenever they occur, and then dial in, using a wireline connection, at the end of the day to download brochures or catalogs.

*Use wireless connections
for time-sensitive data . . .*



*use wireline connections
to get the rest.*



Device and Network Considerations

In a wireless mobile environment, there are many aspects to consider in developing an effective design for a wireless data application. These include the characteristics of the device the application will reside on, the technology used by the network that will transport the data, the battery life of the device, the message routing functionality, and the level of security required, as described in the following paragraphs.

Device Types

A mobile device usually consists of a computing unit together with a modem that allows it to send and receive data over the radio waves. The wireless modem may be built into the device (internal modem), attached to the device through the communications port (external modem), or may reside on a standard PCMCIA card. The types of devices available for wireless communications are described in greater detail in the document “*Wireless Data: The Choices.*”

When developing an application for wireless data communications, it is important to consider the capabilities of the mobile device on which the application will reside:

- Is the device capable of sending as well as receiving data?
- Is interoperability a concern? Connection-oriented wireless modems such as cellular modems are best for the transfer of large files, for online services, e-mail with attachments, and faxes because they can connect to any other modem, at any number, with no changes required at the other end.
- Does the application have a user interface requiring a screen monitor on the device? If so, how large a monitor?
- Does the application require computing power or does the user require access to other applications that would necessitate a laptop computer (or its equivalent)?
- Does the nature of the application require a device with a long battery life? For example, does the device need to be able to transmit all day?
- Are there mobility requirements for the device?

Some devices have built-in application programming interfaces (APIs), third-party APIs exist for others, while others require developers to work from scratch, as described in “Middleware and Tools” on page 11.

Network Types

The networks that transmit the wireless data from one device to another differ in the technology they use to transmit messages, in the range of compatible devices and services offered, in the coverage provided, and in the amount they charge to use the network. You can learn more about the different network technologies by reading the document entitled *Wireless Data: The Choices*.

When developing a wireless application, it’s important to consider the characteristics of the network that will be used:

- **What is the coverage area of the network?**

Users have different geographical coverage needs, which can be satisfied by different network technologies. For example, wireless LANs have been deployed to provide coverage in the major metropolitan areas while satellite networks can provide global coverage.

Developers need to consider how to handle out-of-coverage situations in their applications. For example, a point of sale terminal needs to be able to take money even if the network is down. Also, if users will change from one service provider to another as they roam from one area to another, as is true

for the regional cellular companies providing CDPD services, then developers need to consider how to handle any delays that might result.

- **What is the transmission speed of the network?**

This can vary from 512 - 1200 bps (bits per second) for paging networks, to 2400 - 28.8 kbps (kilo bits per second) for wireline transmissions, to 4800 - 19.2 kbps for two-way wide area packet data networks, to 1 Mbps (Mega bits per second) or higher for wireless LANs. The actual throughput is lower than indicated by the transmission speed and is determined by the network load and overhead. Developers need to take both the transmission speed and the actual throughput into consideration when designing an application, as described in “Design Considerations” on page 10.

- **What packet sizes are available on the network?**

Wide area network technologies such as DataTAC and Mobitex send messages in 240 or 512 byte packets. Messages larger than this size are divided into multiple packets, which are then reassembled before they are delivered to the application. For example, a 600-byte user message might result in the delivery of two packets that are reassembled in the wireless device. Each packet requires acknowledgment from the device, which takes a few seconds to complete. The fewer packets in a message, the shorter the delivery time. For multiple-packet messages, the host and wireless device application must provide the segmentation and reconstruction functions.

- **How much will it cost to transmit data over the network?**

Using networks employing packet data technology is cheapest if you require two-way capabilities with frequent transmissions and relatively small amounts of data. Circuit-switched technology works poorly in these cases because of the setup and disconnect time required for each session. If you are creating an application to run on a packet data network, you’ll need to design your application to minimize the amount of data transmitted, as described in “Limitation of Data Transmitted” on page 11. You also might want to bundle your data or combine messages so that each packet carries as much data as possible, to make the most of the basic packet charge.

- **What type of communications do you require?**

Using wide area wireless data networks, both solicited and unsolicited communications are possible. Solicited communications means that the server does not send data to the client until the client requests it. Unsolicited communications means that the client receives data without prior knowledge

of the communication. Some services are solicited (database access) and some are unsolicited (paging).

E-mail applications can be solicited, in a manner similar to postal service mail, where the mail sits in your mailbox until you fetch it. Or e-mail can be unsolicited, in a manner similar to Federal Express, where the carrier comes to your door and hands you the package. AT&T's PersonaLink Services is an example of solicited e-mail, where mail waits for you to fetch it from a query box. RadioMail is an example of unsolicited e-mail, where mail arrives at your device without you requesting it.

- **What protocols can be used on the network?**

Networks require applications residing on both the mobile device and the host computer to use protocols to translate information to and from the network. Some protocols, such as the standard TCP/IP protocol used by the Internet and by CDPD networks, might already be familiar to the developer. Other protocols, such as the standard context routing (SCR) protocol used by DataTAC™ host applications or the MASC protocol used by Mobitex networks, will most likely be new to the developer. In addition, protocols address different levels of the OSI model and developers may choose to write to different levels, depending on the API they use or their level of expertise, as described in "Protocols" on page 18.

Messaging Functionality

The type of messaging offered depends largely on the networks, although some networks provide a range of services. The developer determines which services to use based on the requirements of the application and the costs involved.

When developing an application, consider the following questions regarding messaging:

- Can the messages be stored in the network until a person comes into coverage or turns on the radio?
- Is interactive data necessary?
- Does the person on the receiving end of the information need to be notified that a message is waiting?
- Does everyone require two-way communications or can a two-way system be mixed with a one-way system (paging for example)?
- Should message flow be solicited or unsolicited?

An unsolicited service requires that the wireless modem be powered continuously. Since this severely limits battery life in some devices, strictly unsolicited services might require users to carry spare batteries or power adapters. Some services, such as e-mail, work best if both solicited and unsolicited communications are supported, allowing the user the flexibility of trading off responsiveness and battery life.

- Should message delivery be guaranteed?

Messaging can be one of two basic types: guaranteed delivery or best effort. The guaranteed delivery scheme provides error detection and an automatic retry request (ARQ) mechanism to confirm to the sender that all the parts of a message have been delivered, and in the proper order. The guarantee requires extra packets, extra time, and extra cost. The developer decides what type of message delivery guarantee is required in the end-to-end application.

Fixed Connection Options

There are three primary options for connecting a host to a network:

- Directly connect the customer's data system (host) to the network with a high speed leased line (e.g. X.25).
- Connect to the network through a gateway service provided by a value added reseller such as RadioMail or SprintNet.
- Use an RF connection.

The decision depends on volume and performance requirements. You can start with RF connection and then move up to other methods as necessary. If you use an RF connection, you'll pay for packets going in both directions. If you use an X.25 connection, packets sent and received by the client are the only ones charged for.

Direct connection to the network is an appropriate choice if you are planning to install and maintain your own host to provide any store-and-forward, interconnect, information provision, or other processing service that your application might need. You might then be responsible for any billing to your customers, depending on the arrangements you've made with the network operator. This is possibly the least expensive method of obtaining network service, but it comes with less direct services available.

Battery Life

When designing applications for mobile use, battery life becomes an important consideration. Battery life is affected by the device, by the network, and by the design of the application. Developers will need to consider the amount of data the user is going to transmit and whether the batteries will last or if it will be possible to change batteries when necessary. Battery technology is changing rapidly and every new device comes with better batteries.

The following modem power management options can be included in an application to maximize battery life:

- PowerSave™ or sleep mode
- Inactivity time-out
- Radio on/off on user demand
- Radio on/off on application command

In addition to these specific power management options, you can increase battery life by designing the application to reduce the amount of data transmitted and the frequency at which data must be sent, as described in “Limitation of Data Transmitted” on page 11. Data compression can significantly reduce traffic volume, which improves device battery life and also reduces network usage costs. You can also extend battery life by batching outbound data transmissions to transmit less frequently.

Design Considerations

The radio channels available for use by wireless modems are narrow-band and have limited information carrying capacity (bandwidth). Additional capacity can be gained only by increasing the number of channels, improving the hardware technology (such as network speeds), or by developing more efficient applications. As an application developer, you can design your applications to perform as effectively as possible within the constraints of the wireless environment.

In a wireless packet data network, where users are charged for each packet rather than for time, this means maximizing the use of each packet. It is important to write applications that make the most out of each packet transmitted or received. This requires a different approach to writing applications. For example, let's say an oil well is equipped with a monitoring device to monitor the

flow rate. If it needs to transmit this information at regular intervals anyway, it could just as well monitor the ambient temperature at the same time.

Middleware and Tools

Most wireless applications can be developed using one of a number of interfaces that insulate the application and its developer from wireless' toughest oddities. These programs are commonly called middleware or toolkits. Some tools are device or network specific, and you'll need to choose a tool appropriate for the environment you're developing for. Some tools can be used only for the end-user device application or for the host application. Tools are provided for different OSI levels and you'll also need to consider the level appropriate for your application and level of expertise.

The Motorola AirMobile™ software development kit (SDK) is an example of such a toolkit. It provides a reliable transport interface as well as an application-level interface. AirMobile for Lotus cc:Mail is an example of an end-user application written using the AirMobile SDK. Many other middleware choices exist.

We recommend strongly that you utilize a wireless toolkit. The design issues discussed in the following pages will serve as useful background information even if you're shielded from having to deal with these issues by the middleware you've chosen.

Limitation of Data Transmitted

In wireless solutions, you can't send large amounts of data cheaply over the radio waves and can't display it easily on most mobile devices. In addition, when you're out in the field, the speed at which users can access information is as important as the speed at which users can process information.

These facts require a somewhat new approach to data-centric applications. In a wireless environment, you can't increase speed and throughput by continually adding bigger and faster lines to transport the data. Instead of the standard approach, which provides the user with relatively large amounts of data and with tools to navigate through it, the user is sent only critical data, which is often all that is needed when away from the office. Quantitatively, this amounts to very little.

In order to send only the data desired, a potentially large amount of raw data that a user has requested must be filtered to determine what parts of it are essential.

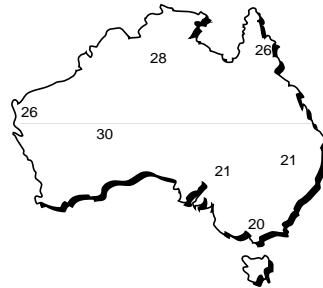
The data might also need to be manipulated and finally formatted before being sent to the user. These tasks are best accomplished on the network component that possesses the most computing power, usually the host server. The user is able to harness the powerful resources of desktop and network computers to find the data and the small form factor and mobility of the client device to view it.

For example, a user regularly receives a map of Australia with an updated weather picture consisting of isobars with temperature gradings. It would be expensive to transmit this graphic wirelessly. However, the application could be modified so that the map itself is stored on the wireless device. And it might suffice to send only a few data points to indicate temperature gradings, or to send only the data points that have changed since the last update.

You can transmit the entire graphic . . .



or just representative data.



In another example, a dispatcher communicates wirelessly with field technicians regarding their location, availability, etc. Time could be saved and airtime reduced if the client application used predefined acknowledgment messages and the technician transmitted only codes identifying the messages, such as:

- 1 = Call received, going now
- 2 = busy now
- 3 = going to lunch
- 4 = off duty

The host application would then expand the code and display the message to the dispatcher.

Helpful Hints

The following are some specific suggestions for minimizing the amount of data to be transmitted.

Application Level

- Keep screens on the client devices, only filling them with data received from the server.
- Use tables matching codes to long descriptions so that only the codes need to be sent over the air.
- Send only data that has changed.
- Don't use polling. The network automatically sends information from clients to their destined server. Servers can inquire if a client is available if they have been out of service for some length of time
- When designing an application for wireless technology, don't assume that all data sent to the client needs to be updated wirelessly and real-time. For instance, updating salespeoples' inventory files every night using a wireless modem makes sense and also enables them to have the latest product number/product descriptions in the field. Only transmit the product codes wirelessly during the day, for inventory updates while at the customers' sites.
- Use an intelligent agent at the server side to monitor a specific situation so the client doesn't have to continually poll the server.
- Concatenate, or couple, messages wherever possible. For example, if a database order procedure requires an inventory query, include the order in the same message. An out-of-stock condition is reported, followed by an error message for the order. But the total amount of RF traffic is reduced.

Transport Level

- Optimize all data transferred in order to gain maximum value from the packets you are transmitting. Attempt to keep the size of messages under 2K bytes.
- Use compression techniques such as v.42bis, JPEG, and GIF at both the end user and host ends whenever possible.
- Limit chattiness. Attempt to accomplish as much as possible in each data exchange.

- Piggyback multiple stop-and-wait exchanges whenever possible. This reduces delays and makes more efficient use of the shared channel.
- Don't ACK each packet, just enumerate the ones received correctly. Retransmit only those not received correctly.

Challenges for Packet Data Networks

If you are developing an application for a wireless packet data network, you'll need to take several issues into account:

- Packet delay
- Packet sequencing, including dropped packets, duplicate packets, and packets out of order
- Mobility
- Network congestion
- Network latency

Packet Delay

Delays in the time required for a data packet to travel across the network vary depending on communications equipment on the path and the network load, which is the amount of data in the network at any given time. Some packets might have to be delivered several times before they are received correctly.

Additional delays are encountered when the wireless device is in the process of roaming from one cell on one radio channel to a cell on another radio channel. If the cells are controlled by the cell controller, the delay time is quite short, but can increase if the cells are controlled by different cell controllers on different subnetworks.

Mobile applications must allow plenty of time for packets to arrive at their destinations. For example, for a DataTAC 5000 network operating at full rated capacity, the mean transit delay between a network host and a wireless device is typically up to four seconds. The application developer must be careful to evaluate operational scenarios to accommodate the variable transit time in the application design. This is the most significant problem to be overcome by application writers who are porting IP applications from a LAN or wireline modem network to a packet data network.

Packet Drops

When the network load becomes too great, some form of congestion control may be invoked. Packet drops are the most common type of congestion control and can occur at many points along a data path. Even without congestion, noisy lines and marginal radio coverage can cause packet drops.

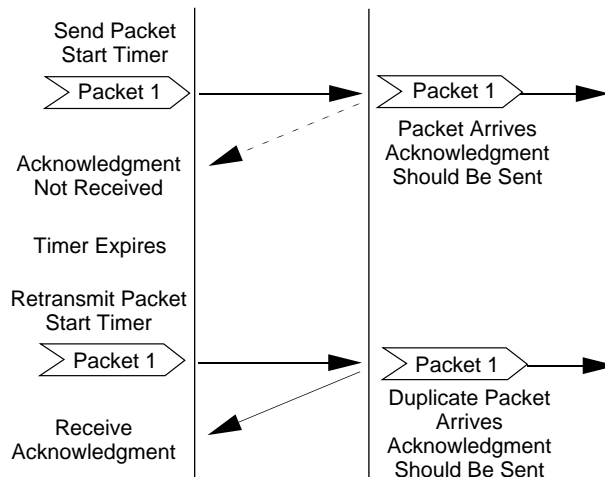
Most connection-oriented protocols such as TCP include methods of recovering from dropped packets; however, applications using datagram protocols such as UDP need to implement their own recovery mechanisms.

Developers can employ recovery mechanisms such as stop-and-wait and sliding-window. However, make sure the retry mechanisms have some kind of exponential backoff, where the first retry occurs after 30 seconds, the second follows after an additional 60 seconds, and so on.

Duplicate Packets

Duplicate packets are typically introduced by pieces of the network that have a re-transmission facility. For example, if the acknowledgment of a packet is lost, the sender of the packet will time out and resend the packet. The receiver will acknowledge the second packet and forward the second packet, the duplicate, on to the application. This is particularly troublesome when the packet contains commands.

Duplicate packets

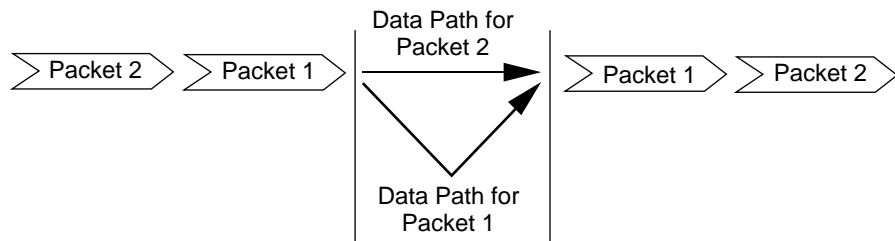


Packets Out of Order

Packets can be placed out of order when the data path (and delay) changes due to a cell hand-off, rerouting, an equipment failure, or a recovery which causes a primary/secondary switch. Specifically, packets can be received out of order when the path changes from a longer delay path to a shorter delay path.

The simplest solution is to add packet sequence numbers, and applications can be designed to ignore unexpected packets and to request retransmissions of missing packets. Packets arriving out of order should be logged to determine whether the problem resides in the network or in the application.

Packets out of order



Mobility Issues

A wireless device can roam without restriction and can exit the network radio frequency (RF) coverage area. When it is out of coverage, it is unable to receive or successfully transmit messages, which causes unexpected delays in processing or responses.

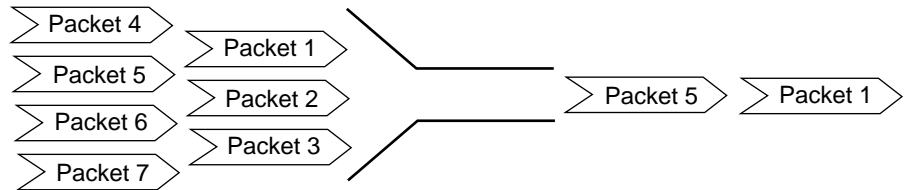
When a device is outside the coverage area, the applications are informed of failed delivery. It is their responsibility to take appropriate recovery action. Developers should thoroughly design for and test exception handling for the times when the client is out of range or off.

Wireless communications links can occasionally become suddenly unavailable. Applications must always allow for this possibility and recover gracefully. Error recovery routines should be thoroughly tested. Complex transactions should, if possible, be broken into smaller ones. Applications such as point of sales systems should be able to accept money even with the network temporarily unavailable.

Network Congestion

Most applications have some kind of retry or error recovery mechanism, and if these mechanisms are invoked, due to retransmits, the host has to do two or three times the work of a normal transaction. The network might slow down, and the application has to allow for delay in the delivery of messages and responses.

Network congestion



Applications can handle network congestion in the same way that they handle packet delays. They can also prioritize control and status packets.

Network Latency

The amount of time it will take for a device to register with a network is unknown and will vary depending on network coverage and load. Because of this, an application should not begin sending packets immediately.

Security

For users transmitting data over public networks, security might be a concern. Some security measures are provided by the network. For example, packet data networks that divide data into packets for routing by their nature provide some data security. CDPD networks provide encryption for the data they transport.

Wireless data communications also allows developers to incorporate customized, end-to-end security solutions into their applications. The level of security implemented depends on the requirements for the application and the data. These security techniques can range from the simple, such as CRC or checksum, to the sophisticated, such as digital signatures and message authentication codes.

User Interface

Application developers are urged to remember the needs of the users. This means knowing what the users need to know and when they need to know it.

The following list includes typical information that users want to know. Consider using pop-up information boxes to provide status information at the user's request, without distracting from the application.

- In-range or out-of-range conditions
- Relative signal strength and/or channel quality
- Application delays resulting from communication failures
- Low battery conditions
- Positive confirmation of application and modem interaction
- Work-in-progress feedback (for example, transmit notifications)
- PowerSave or sleep mode data (if it is at the user's discretion)
- Device buffer-full condition
- Other fault conditions
- Radio on/off

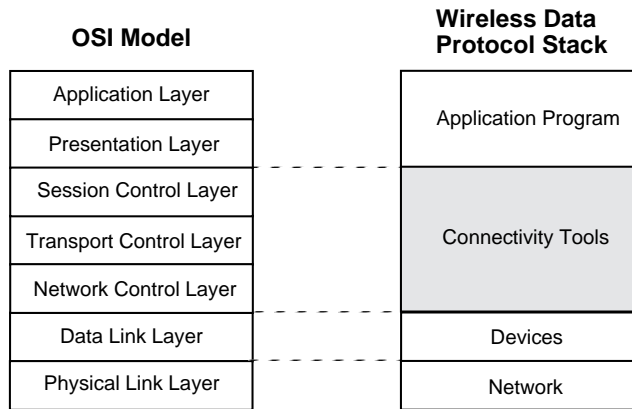
Protocols

Developers need to consider the protocols available for the different wireless networks. Some network technologies require specific protocols and others, such as CDPD, allow a variety. The developer needs to examine the characteristics of the available protocols and decide which is most appropriate.

Some protocols, such as TCP, use acknowledgment-oriented connections that are very chatty and time out quickly. This means that the protocol has built-in response dialogs across the communications channel: *Here's packet no. 50. I acknowledge pack no. 50 and am echoing its contents. Here's the acknowledgment of your echo.* This type of protocol provides delivery guarantees at a cost. Consider reducing traffic across the network or using an alternative protocol, such as UDP (user datagram protocol) which isn't as chatty as TCP but also doesn't guarantee delivery.

In addition, developers need to consider the overhead involved in using a protocol. For example, let's say you are using the common TCP/IP protocols to send a message. IP is a network layer protocol which identifies the destination of the packet, anywhere in the world. It requires a minimum 20-byte message header. TCP is a transport layer protocol that keeps track of how many packets are in the message. It also requires a minimum 20-byte message header. If you used TCP/IP to send a one-byte message, you would be using 40 bytes of overhead to send one byte. When designing an application that uses TCP/IP, it's important to maximize the content sent in a single packet.

Although data communication applications are designed to appear simple to the user, several layers of protocols are required to enable computers to connect and share data, as shown in the OSI model. You can use connectivity tools, as described in "Middleware and Tools" on page 11, at the transport layer, the network layer, or the data-link layer of the OSI stack.



Which level of connectivity tool you choose, or which protocol level you write to, depends on the particular requirements of your application and the expertise of your development team. The Motorola AirMobile Software Development Kit, for example, provides two application programming interfaces, performing at the transport layer and the driver layer.

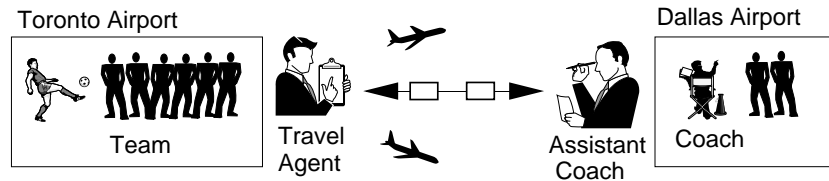
APIs operating at the transport level reorder packets in the proper sequence before presenting them to the application. Developers can use APIs to write to the device level if the application will never be sending messages requiring more than one packet.

The roles of the protocols (TCP/IP, for example) in message routing can be illustrated by an analogy. Let's say a soccer team (the message) from Toronto

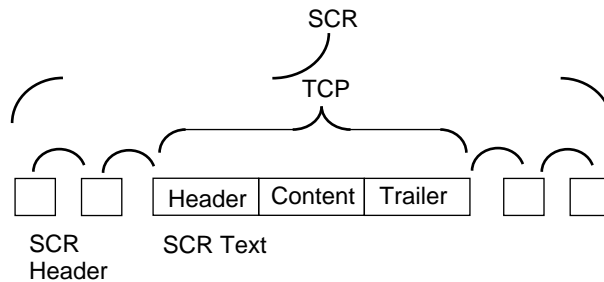
has been selected at the last minute to play in a championship match in Dallas. All the team members go to the airport, but only two can get on the plane headed directly to Dallas (the packet isn't big enough). A couple of other players are routed through Los Angeles, and the rest are routed through Denver. The coach (TCP) waits at the Dallas airport (the IP address) to assemble and transport the team to the stadium.

Knowing the tendency of the coach to fret, the assistant coach (an intelligent agent) takes on the responsibility for checking off the players as they arrive and assures the coach that everything is proceeding as planned. The team travel agent (another intelligent agent) is making sure all the players get on the plane in Toronto. When the travel agent checks off the last departing player, he notifies the assistant coach. However, the assistant notices that the goalie has not arrived and notifies the travel agent, who is able to hunt down another goalie and send him out.

Finally, the assistant tells the coach that all the members have arrived. Together, the assistant and the travel agent (the protocol satisfiers) were able to transport all the players with only a couple of calls back and forth regarding the goalie. The coach, who on his own would have been calling home after each arrival, wondering where the rest were, never found out about the close call with the goalie and was able to concentrate on the game strategy.



Developers should, if possible, avoid wrapping protocols around protocols, providing unnecessary layers of address information. This could happen, for example, if an application transmitted TCP/IP messages within a closed network such as ARDIS, using the SCR protocol at the data-link layer.



Again, an analogy can help to clarify the issue. Let's say Linda wants to send some information to Terry, who works for another company. She puts as much information as she can fit into an envelope and writes the exact address of the destination (the postal address) on the front. This is the task of IP. Since several envelopes are needed, and she wants them to be read in order, she labels each envelope 1 of 6, 2 of 6, etc. This is the task of TCP.

When the envelopes are delivered by the mailman to the company, the company mail clerk stuffs all the envelopes addressed to Terry into an interoffice mail envelope and labels it with Terry's name. This is the task of the SCR protocol, which transports data within the closed (DataTAC) network.

At this point there are two layers of addresses (postal and interoffice) for the information contained inside the envelopes. If all the information had fit inside one postal envelope, the TCP layer would not have been required. If both the sender and receiver worked within the same company, only the interoffice envelope (SCR) would have been needed. Using only the protocols required increases the efficiency of the application.

Getting Started

Once you've determined how to effectively design your wireless data application, you're ready to begin. The following figure illustrates the steps involved in bringing your application into production in a wireless environment.

These steps are described in the following paragraphs.

Development Environment

Most wireless applications today run on industry-standard platforms or variants of those standards. The wireless applications are merely extensions of existing communications solutions. By the same token, wireless application development is an extension of application development on various platforms. A developer's first step is to obtain the standard development tools, such as the following:

- For Windows-based PCs, become familiar with the tools and test environ-

Development Tools

As described in “Middleware and Tools” on page 11, we recommend that you use a toolkit to develop your wireless application. Toolkit providers offer assistance to developers that ranges from technical support over the telephone to a formalized developer’s program. Consider the type of support provided when you choose a toolkit.

A list of developer tools, including descriptions and contact information, is available on the World Wide Web at the following address:

<http://www.mot.com/wdg/>

Network Operator Relationships

At some point during the development cycle, developers should also establish a relationship with the network operator, such as ARDIS or RAM, that will be used to transport the data. Their programs provide a number of support services to developers who wish to use their network. These services might include:

- Training on network functionality
- Consulting on application development
- Consulting on radio-specific issues
- Help with business development
- Application design review
- Certification or confidence testing

If service providers such as AT&T or RadioMail will be involved in handling any of the data traffic from the application, they should also be contacted.

Development and Testing

During development, organizations typically uses dial-up connections to networks instead of dedicated private lines. Alternatively, they can use a wireless approach to connect equipment to the network, especially when a pilot project is done with more than one type of wireless system. Once a customer has committed to using a network operator and the number of potential users has been defined, a leased line can be set up. Customers have the choice of installing and managing this connection themselves or allowing the network operator to manage it.

Testing under various combinations of load, packet delay, and network reliability is the most effective way to ensure that the application will work in a live network. Network emulators can be used to accomplish this part of the development process.

Plan to start small with your wireless project. Implement a pilot system first with only one of the major users. Then, compare its success against the objectives of the wireless project. If possible, use more than one type of wireless system to support your pilot project. Network operators such as ARDIS operate a development and test host that is separate from the production host. Developers can use this host to test their application prior to public roll out.

Once a developer is ready to go live with an application, certification or confidence testing is usually performed by the network operator to ensure that a developer's application does not adversely affect the operation of the network.

Release

Once an application is released, the customer support structure should be in place. The developer can proceed with sales and marketing strategies, which might include bundling agreements with carriers or device vendors, and be confident that the application will perform well in the wireless environment for which it was designed.

Developer Program Contacts

The wireless data communications industry is still in its infancy and many interested parties are committed to making it a success. Developer's programs are available to assist you in the wireless issues of application development and to provide support once your application is in production.

Your task will be easier if you take advantage of the available assistance. Here are some contacts to get you started:

AirData Developers Program (a CDPD network service provider in the U.S.)
800-524-3388
<http://www.airdata.com>

Apple Programmers and Developers Association
800-282-2732
<http://www.info.apple.com/dev/developerservices.html>

ARDIS Developer Program (a Motorola DataTAC network service provider in the U.S.)
800-57-ARDIS

Metrowerks (General Magic development environment):
800-377-5416
sales@metrowerks.com

Motorola AirComm Developer Program (a Motorola DataTAC network service provider in Hong Kong)
852-2599-2865
e-mail: kai_chan@hongkong.super.net

RAM Developer Program (a Mobitex network service provider in the U.S.):
800-RAM-3210
developer@ram.com

Telstra (a Motorola DataTAC network service provider in Australia)
61 2 911 3153 from outside Australia, 800 633 785 from within Australia
e-mail: kpank@nentmdr1.telecom.com.au

