

Maximizing IP-Based Applications for a Wireless Environment

Mike Koch
Motorola Inc.

Abstract

The intent of this paper is to discuss ways to maximize the use of IP based applications in a wireless environment. Over the years, the wireless environment has posed many challenges to developers. We are in a time where society is putting IP based applications in an environment where IP was never designed to operate. It's the traditional square peg-round hole problem. This paper will hopefully provide some ideas on how to put the square peg in the round hole and maximize IP in a wireless environment.

Introduction

Who would have guessed that a requirement placed by the Department of Defense in the 1960s for a packet-switched network would have grown into what is known today as the Internet? The Internet and the Internet Protocol (IP), have changed the way society communicates and conducts business. Our LANs and network infrastructure products that make up the Internet and World Wide Web have reached incredible speeds of 1 Gigabit/sec or faster which allow them to support a variety of media including text, audio and video. Having a common communications protocol such as IP, which runs on top of these networks, has opened the door for many application programmers to develop unique applications for businesses.

Today's fast-paced, customer-centric focus and other business objectives have pushed the workforce into being mobile. Though not being tied to a desk has many advantages, mobile workers need to have the same information at their fingertips whether they are in the office or on the road. Though wireless data communication has been around for over 15 years, the limitations of radio still don't allow the same data rates to be obtained in the wireless world as in the wireline world. Today, the current over-the-air data rates in the private data industry peak out at 19,200 bps. Compared to their 10 Megabit and 1 Gigabit wireline cousins, wireless data systems are considerably slower. Therefore, when you marry applications that were developed to be used on a high speed, IP-based network with a wireless network of today, the result can be (and usually is) very disappointing.

Though data rates are important in wireless communications, the efficiency of the application and the protocols used for communication play equally important roles — especially in the wireless world. An application can have all of the bells

and whistles that make it a killer application, but if that application wasn't designed to be used in a wireless environment its performance will most likely be less than desirable. This goes for both IP based applications and non-IP based applications. Likewise, the Internet Protocol was designed to be used in a wireline environment, not a wireless environment.

To Connect or Not To Connect

IP based applications are either connection-oriented or connectionless. As shown in Figure 1, a connection-oriented application uses TCP and IP when communicating over a network while a connectionless applications use UDP and IP when communicating over a network.

Figure 1: The Internet Protocol (IP) Model

Connectionless

Connection-Oriented

UDP

TCP

IP

(Network & Network Protocols)

Ethernet over 10BaseT, Token Ring over Twisted Pair, CDPD over wireless....

Application, RIP, DNS, SNMP, TFTP, DHCP...

Application, HTTP, Telnet, FTP, SMTP, POP...

Though you may not have realized it, some common connection-oriented applications are web surfing, email, file transfer and remote terminal log-in. When one of these applications initiates a remote connection, it is the responsibility of TCP to establish the connection. A TCP connection consists of a three-step process:

1. Establish a connection between the sending application and the receiving application
2. Transfer the information and acknowledge that it was received properly. (i.e. the receiving application must acknowledge that the data was received.)
3. Tear down the connection that was established in step 1.

A connection-oriented process is similar to the process of having a conversation with a friend on the telephone. First, you must call your friend to see if he is home. If someone answers then you would ask to speak to your friend. Assuming your friend is there and picks up the phone, you have now established a connection with your friend. Once the connection is established you can begin a

conversation. This exchange of conversation usually consists of something being said and a response from the receiving party to indicate that they heard what was said. If their attention was diverted while you were speaking and they didn't hear everything you said you may have to repeat what you had just said. (In a TCP/IP based application, TCP works the same way. If it doesn't receive an acknowledgment within a certain amount of time it will retransmit the message.) This part of the process, speak then receive an acknowledgment, is a very important part of the process. It tells you if the other person received the information that you just told him. Once the conversation has ended you would start the process of saying goodbye and then terminate the call by hanging up the phone.

As you can see, a connection-oriented process provides some form of reliability. The reliability doesn't mean that each time you attempt to make a connection that your friend will be there or that you'll have a good phone line. What the reliability does provide is an series of acknowledgments to let you know that 1) a connection could be established and 2) the information was transferred successfully. Though they provide a certain level of reliability, connection-oriented processes have a lot of overhead associated with them. Here lies one area that can be optimized.

In contrast to the connection-oriented process, the application in a connectionless process does not establish a connection with the receiving application. A connectionless process makes the assumption that either the medium being used (wireline or wireless) will get the message there or the application will determine that the message was received . For example, rather than call your friend on the phone, you could have written everything that you wanted to tell your friend in a letter and sent it using the United States Post Office. The U.S.P.O. does not guarantee that your letter will get to the recipients address, only that it will use its internal network of mail handlers and carriers to make their best attempt to get your letter there. As we all have experienced one time or another, sometimes this process works and sometimes it doesn't. If you want to verify that the letter was delivered to your friend, you could ask your friend when you saw him/her if they got your letter.

Who determines if an application will use a connection-oriented or connectionless process? The application developer does, and their decision is typically based on the type of network that the application will be used on. Up until now, networks as we know them have predominately been wireline-based, operating at 10Mbps or faster. Since capacity on most wireline networks isn't an issue, application developers haven't worried about the additional overhead required for connection-based services.

How To Maximize An Application For A Wireless Environment

Here is a collection of ways to get the most out of an application that will be used in a wireless environment. Some of these techniques are applicable to new applications, some to existing applications and some to both. I've categorized them in three areas:

1. Application
2. TCP Optimization
3. Other

I. Application

Applications are those computer programs which people use for business or for recreational purposes. Here are some guidelines to follow when designing or evaluating an application that will be used on a wireless network.

1. Reduce the amount of information that will be sent over the air. Though this may seem obvious at first, you would be surprised how often this is overlooked. Below are some ideas to keep in mind when developing either a new application, purchasing an application or evaluating an application that will be used on a wireless network.
 - a. A network has what is known as a MTU (Maximum Transmission Unit). This is the maximum message size that can be accepted. Messages that are larger than the network's MTU get broken down into multiple messages that are the MTU length or smaller. Ask the network provider what their MTU is and try to design around it. If you are going for speed, keep your messages smaller than the MTU. If you are going for efficiency, then your messages will most likely be larger than the MTU — but work with the network provider to find the right size messages for the performance that you desire.
 - b. A second way to reduce the amount of information that is sent over the air is to use the concepts of forms. Forms use the same basic principal as records in a database. When you look at a record in a database you have a piece of information, a record separator, more information, record separator, and the process continues until the last piece of information in the record. Contrary to this format, when VT100 is used to send information from one computer to another it is sent as a full screen of text. If your screen is 40x80 and you only have 2 characters to send, then you'll be sending a lot of blanks. Therefore, the rule of thumb is to send only the information that needs to be sent.

- c. If the application allows a user to access and retrieve large amounts of text, such as historical information about a work order, consider using a filter to throttle how much information should be sent. For example, let's assume that the historical information that can be obtained has 12 months of information and is 15 kilobytes long. Rather than send the entire 15K over the air, you could allow the user to select the information that is either relevant to the particular job being performed or possibly allow just one month of information to be sent at a time. If more information is required then let them request it.
 - d. A practice that you can almost never go wrong with is to use some form of compression. However, before you add compression at the application level, check with the network provider to see if they are providing compression. The rule of thumb is don't compress a compressed file.
 2. Stagger download of the morning orders. In the utility industry where wireless data systems are used in Field Service, maintenance and repair, the busiest time of the system is always the morning download of orders. Let's assume that a typical Field Service representative can complete 10 orders per day and that there are 100 Field Service representatives at the start of the shift. Assuming that an average order is about 2000 bytes, the system would have to send out 2 Million bytes of information in a relatively short amount of time. If the data system has a 19,200 bps raw data rate and an effective data rate of 14,400 bps, and it was an ideal system (the system had no queuing delay, transmission delays or transmission error due to collisions) it would take ~20 minutes to download all of these orders. Since we don't live in an ideal world and ideal systems don't exist, it would take somewhere between 30 and 45 minutes to download these orders. However, if only a few orders (1 or 2) were sent to the technician and the remaining orders were sent to them while they worked on the first orders, the load would be distributed over time and the technicians wouldn't lose any time. A variation of staggering the orders is to send a summary of the orders and allowing one or two of the order details to be sent.
 3. Consider a UDP based application. UDP (User Datagram Protocol) has been nicknamed the Unreliable Data Protocol since there aren't any acknowledgments given by the system to inform the sending application that the message was received. With that definition, why would you want to consider designing or using a UDP based application? These days, not only have networks become faster, they've become better. Many RF service providers, both public and private, are able to give coverage performance guarantees to their customers. Be aware, a coverage guarantee isn't a map that shows where the provider expects there to be some signal level. A coverage guarantee not only shows where you will

have coverage but it tells you the quality of the coverage: i.e. the packet success rate or message success rate. If a system is designed to provide a 95% message success rate for a MTU of x bytes or less, then you can design your application around the MTU and be relatively assured of the messaging. To reduce the probability of lost messages, adding sequence numbers to the messages and requiring time-sensitive messages to be manually acknowledge will help fill the gaps.

- a. Sequence numbers help the application to keep track of which messages were received and which messages were not received. If each inbound message had a sequential sequence number and the receiving application could echo back the last received sequence number in its reply, both sender and receiver would have a way to determine if any messages were missed. Though this concept has been greatly simplified, it's easy to see how such a technique could help prevent messages from being lost.
4. To help ensure that high priority messages are received, or normal priority messages for that matter, the application can be designed to require the recipient to manually acknowledge that the message was received.

II. TCP optimization

1. Keeping TCP sockets open. As previously mentioned, TCP-based applications are connection-oriented and require the sending application to establish a connection with the receiving application. Once the connection is established, the data can be exchanged, then the connection can be torn down. Here lies an area of improvement. Once the connection is established, the connection could be left alone to allow data to be transferred whenever it was requested. The application could be written to close the connection when the originating application requested it to be closed, if one application hasn't heard from the other application after a predefined time period, or after a particular time of the day. If you are an IT or computer application developer, then you're probably thinking of all the resources that this technique will require — and you are correct. When considering this option, the cost of the additional server and networking equipment should be weighed against the cost of the productivity that will be lost due to the delay of setting up and tearing down the connections.
2. Middleware for improving the TCP connection. Middleware is the software that fits in the gaps between different between the application and the TCP/IP stack or between the TCP/IP stack and the network interface. It usually consists of two components, a server running middleware software for the host and software running on the client. Not only can middleware

help improve the performance of the application in a wireless environment, it can features such as compression, encryption, and even authentication.

Over the past couple of years, several vendors have realized that there a lot of customers that have existing TCP/IP based applications that aren't wireless-friendly, and would like to use them in a wireless environment. With this in mind, companies like IBM, MDSI and Nettech have developed middleware that improve the performance of TCP/IP-based applications in a wireless environment. The main idea thought behind each of the vendor's solutions is to minimize the amount of overhead that is associated with TCP/IP based applications. Since there is very little overhead associated with UDP/IP-based applications, there isn't much value that middleware could provide in terms of performance.

Basically, the middleware works in one of two ways: either the middleware manages the TCP retries or it spoofs the application. In the first method, the middleware knows that TCP transmissions are time-sensitive. If a TCP transmission does not receive an acknowledgment within a certain amount of time, then TCP will retransmit the message. Since the TCP/IP suite was designed to operate on a high speed network where delays are in the millisecond range, you can imagine how many retries would occur on a wireless network where the delays could be up to several seconds. The middleware realizes that the wireless network is slow, therefore it buffers the retries and prevents them from being sent over the network. When the middleware finally receives the acknowledgment from the remote device, it will pass it on to the sending application and the process will continue. If the middleware had not filtered the retransmissions the RF system potentially could have been flooded with TCP retries.

The second method involves spoofing, which takes a completely different approach than the technique just described. Unlike the example above where the TCP/IP messages are actually transmitted over the medium between each computer, in spoofing the middleware intercepts these messages and responds to the sending TCP/IP stack as if the receiving machine had responded. The TCP/IP setup and teardown message only go between the application and the middleware on each device, not over the wireless network. The middleware takes the responsibility of delivering that message to the destination machine.

As you can see, there are some significant advantages and disadvantages to both methods. Each vendor has some unique features in their middleware and you should work with them to find out what is the right solution for your application.

3. Middleware for improving Web based applications. IBM also has taken this a step further and offers middleware called Web Express that improves

the performance accessing web pages over a wireless connection. Though one might think that accessing a web page consists of nothing more than fetching that page and displaying it on your computer, it is actually quite a bit more complex. On a web page, text as a whole is treated as a single entity that requires a single connection to fetch it. However, each graphic on a web page (banner, button, moving objects) is considered a separate entity and requires a separate session to fetch it to your web browser. IBM's Web Express improves this process. Instead of opening a session for text and each graphic it will spoof the application into thinking that separate sessions have been established but it will only open one connection between the sending application and the Web Express gateway. The gateway will communicate to the web site using standard HTTP requests with separate sessions for text and graphics, but it will reduce it to one session when it sends the information to the requesting application. The Web Express product has other features like web caching for commonly used graphics and web pages to improve the performance even further.

III. Other

Earlier I discussed the concept of staggering the download of morning orders and how it can help reduce the peak traffic periods during normal operations. One other concept that can be added to assist this further is to use a wireless LAN to supplement the wide area wireless data system. Since many operations keep their field service and maintenance vehicles in a central location, it is possible that a wireless LAN could be put in these areas and could be used to download orders to the vehicles in the morning and receive order completions and reports in the evening. Since wireless LANs operate at speeds up to 11 Mbps, they could easily handle all the orders that an organization could send to their field personnel. Once users received their orders for the morning they could restart their terminals and use the wide area mobile data system for updates and emergency messages.

The Future

Though next generation cellular systems boast much higher data rates of reaching raw data rates up to 384 Kbps their actual throughput will be about half of their raw rate. Even so, these next generation systems will still be considerably slower than their wireline counterparts. Though WAP (Wireless Access Protocol) enabled devices with microbrowsers are beginning to emerge on public systems, their size and form factors are not conducive to the types of applications that the traditional utility and public safety wireless data users operate with. WAP is basically an efficient TCP/IP suite of protocols and applications that was designed for IP communications in a wireless environment. If WAP continues to

develop and application developers begin to understand the limitations of the wireless world, then we'll begin to see applications that are not only useful, but also very efficient over-the-air. Since the wireless industry follows the wireline industry, people will find ways to consume the bandwidth that is given to them. If applications are written with the characteristics of the wireless networks in mind, then you can expect to see a good return on your investment.