Polytechnic
UNIVERSITY

Delivering Information "To Go" via Infostations

Phyllis G. Frankl
David J. Goodman

November 2002

WICAT TR 02-008

Delivering Information "To Go" via Infostations

Phyllis G. Frankl
David J. Goodman
Polytechnic University
6 Metrotech Center
Brooklyn, NY 11201
pfrankl@poly.edu, dgoodman@poly.edu

Abstract

An Infostation provides wireless information services in a limited coverage area. As users with notebook computers or PDAs pass by an Infostation, they receive useful information, with little or no human interaction. This could be information that is most relevant near the Infostation, such as local maps, restaurant listings, or information about courses at a university. Or it could be information of general interest, such as news articles or music. This paper presents an overview of the Infostation system we are developing at Polytechnic University, including motivation, system design, our initial system prototype, and preliminary data on performance.

1. Introduction

In the 1990s, the hottest items in information technology were the Internet and wireless telephones, which until now have followed separate paths. The "wireless revolution" anticipated in the present decade refers to the convergence of wireless communications and the Internet. While the value of a Wireless Internet has been apparent for many years, the reality has been retarded by the low speed and low quality of wireless links, their high cost, the power demands of portable terminals, user interface issues, and the mismatch of Internet content and the capabilities of small, portable devices. The present optimism rests on the belief that a few emerging technologies - most notably third generation cellular radio systems (3G) and the Wireless Applications Protocol (WAP) - will overcome these obstacles.

A close examination of 3G and WAP reveals that they address only a fraction of the challenges of the Wireless Internet and that they introduce problems of their own. The main attraction of 3G is the high bit rates it offers to individual information terminals: up to 2 Mb/s, compared to 10 kb/s in present cellular systems. However, 2 Mb/s can only be achieved in under favorable conditions of radio propagation, mobility, and network activity. Otherwise, the rate adaptation mechanism of 3G will deliver considerably lower bit rates, with the bit rate changing as the terminal moves from one place to another and as the demand on the network changes [NAN]. This situation is far removed from the "anytime, anywhere, any type" of information services anticipated by consumers. On the software side, technical and business professionals express a growing list of concerns about WAP including issues of security, interoperability, and applicability to new generations of portable devices [BAN, LEA].

These and other doubts about the intrinsic merits of existing technologies suggest that new, rather than incremental, approaches to hardware and software design will be needed to fulfill the promise of the Wireless Internet. With respect to hardware, we have

suggested that the future can be well served by reversing five basic assumptions of cellular system design [GOO]. The IEEE 802.11b ("Wi-Fi") WLAN standard differs radically from cellular systems in three respects: it uses unlicensed, rather than licensed, spectrum bands; it provides service in isolated, rather than ubiquitous, geographical areas; and it offers asymmetric, rather than symmetric two-way information transfer between terminals and access points. In addition, it offers much higher data rates than current cellular technology or forthcoming 3G technology. This paper describes the Polytechnic Infostation project, in which we are developing protocols and application programs that use 802.11b WLAN technology to distribute data within isolated coverage areas.

As shown in Figure 1, an Infostation consists of a radio tranceiver (access point, or AP), that provides high bit rate, low cost, low power network connections to portable terminals in a restricted coverage area, along with computer hardware and software that caches relevant data and schedules transmissions and receptions. Because a subscriber to an Infostation service may spend a short time in the service area of each Infostation, the information transfer should be organized in advance and should take place at the speed of electronic processes rather than the speed of human-computer interactions. We are developing protocols and applications that provide useful information services to users, while requiring little, if any, human interaction as the mobile user passes through the coverage area.

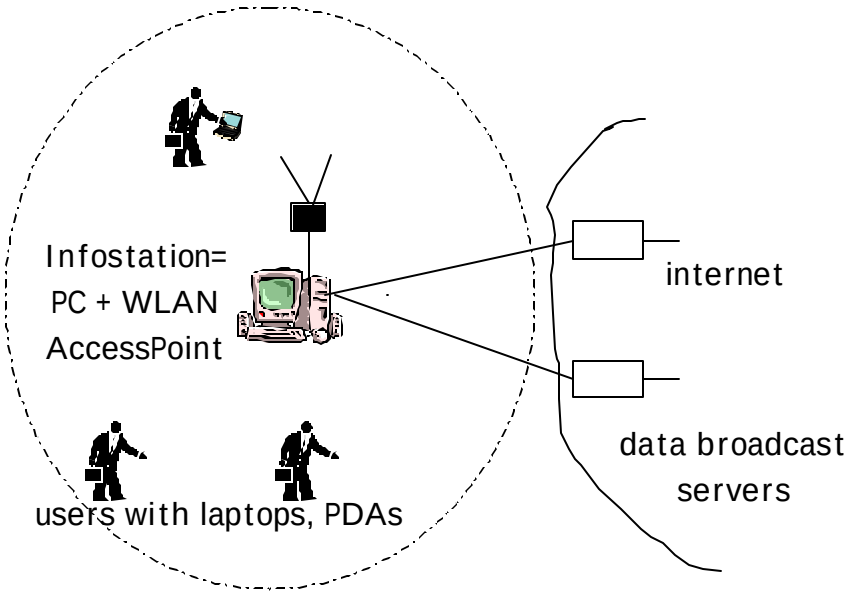

Figure 1: Infostation system elements

The Infostation paradigm is motivated by the premise that no single "one size fits all" technology is suitable for all wireless information services. The best way to deliver information may depend on various factors about the "geography" of the information, including spatial and temporal aspects, as well as characteristics of the user(s). Some

information, such as emergency messages, must be delivered "anytime, anywhere." The ubiquitous coverage offered by the cellular network is needed for such information. For many other types of information, "many-time, many-where" coverage is sufficient. Some information is only relevant in certain areas; it may be preferable to receive the information in the area where it is relevant. Other information is relevant everywhere, but not urgent, so users may be willing to wait for it, if doing so is less expensive than receiving it immediately. Some information is relevant to a single individual, some to a small group, and some to many people. All of these factors affect the selection of an information delivery mechanism. Table 1 shows the eight extreme points in the three-dimensional information geography of spatial, temporal, and personal relevance. It suggests examples of information that conform to each one and technologies appropriate for information delivery.

Who	Where	When	Information	Technology
L	L	L	Urgent meeting request	Cellular
L	L	G	Personal reference material	LAN/Cellular
L	G	L	Phone call	Cellular
L	G	G	Email	Infostation
G	L	L	Traffic reports, weather	Infostation/Cellular
G	L	G	Maps, tourist information	Infostation
G	G	L	Stock market prices	Cellular, satellite
G	G	G	Music, e-books	Infostation

L=local relevance G=general relevance

Table 1: Geography of Information

The Infostation concept was originally proposed at WINLAB at Rutgers University [FRE, GBM, FBB]. While the Rutgers work focuses primarily on the data link layer and below, our work focuses on developing Infostation applications and transport layer mechanisms to support those applications. Much of the Rutgers work is intended to support "drive-through" Infostations, in which mobile terminals pass through the coverage area at vehicular speeds. In contrast, we are initially considering "walk-through" Infostations, where users can be expected to spend on the order of 20 seconds to two minutes in the coverage area. Some of the lower layer technical challenges due to channel fading are less severe (though not completely eliminated) in this context.

At the other extreme on the mobility scale, many universities, including Polytechnic, have installed campus-wide wireless LANs and numerous commercial ventures have recently been launched to provide wireless Internet access in coffee shops, airport lounges, and hotels. These allow stationary users to use the Internet in the same way that they would use it over a wired connection from home or from the office.

In contrast, the Infostation system we are developing is targeted to users who are on the move. Consequently, we wish to organize information transfer so that little or no human interaction is needed while the user is in the coverage area. For example, consider an Infostation located in a railway station. While sitting on the train, a user writes some memos on her PDA . She decides that she'd like a nice lunch after her meeting, so she

tells an Infostation application program running on the PDA that she'd like information about restaurants. As she walks past the Infostation on her way out of the train station, without her interaction, software running on her PDA uploads the work she's done on the train, downloads personal messages that have been pre-fetched to the Infostation, and downloads reviews of local restaurants. Soon, but perhaps after she's left the area, the Infostation forwards her memos to her office computer. Later, when she's sitting in the meeting, she reads the restaurant reviews.

An Infostation is loosely analogous to a take-out restaurant. Customers (often) decide roughly what they want ahead of time, pass through the restaurant quickly, then take their food away for later consumption. Similarly, Infostation users get their information "to go", then consume it at a more convenient time.

The remainder of this paper describes the prototype Infostation system we are developing for deployment on our campus. Our initial applications involve downloading information of interest to a group of users, but the system is designed to accommodate more general classes of applications and to facilitate experimentation to various approaches to a variety of technical issues. Section 2 describes the system design and Section 3 describes the current prototype. Section 4 reports a trial deployment of the system, including preliminary data on performance. Section 5 discusses continuing work and open problems. We illustrate some aspects of the system by reference to our first Infostation application program, which distributes course notes to students' notebook computers. When students first subscribe to the application, they enter the courses they are currently taking. Each time they walk past an Infostation, the application checks whether any new files (or new versions of existing files) have been posted on the relevant course web page. If so, those files are downloaded to the notebook, without user intervention.

2. System Design

The system architecture has three components, as shown in Figure 1, above:

1. Mobile terminals (laptops or PDAs) whose users would like to receive or transmit information when they are near an Infostation transceiver. We refer to the software running on these devices as *client software*.
2. The Infostation server, comprised of an access point along with a PC that caches data items available for transmission to mobile terminals and stores/forwards data items sent from the mobile terminals. We sometimes refer to this subsystem as *the Infostation*.
3. Connection to the Internet via proxy servers. Software running on the server interacts with the outside world through these proxies (e.g., by crawling relevant web sites or by pre-fetching data from users' home computers) in order to cache relevant information on the server.

Our Infostation system uses IEEE 802.11b ("Wi-Fi") technology for data transmission. This offers data rates of 1, 2, 5.5, or 11 Mb/s within a coverage area of up to roughly 50 meters radius (with smaller coverage area at higher data rates).

A number of high-level goals underlie the Infostation system design:

1. To accommodate mobile users who are walking past the access point, the system should support applications requiring little or no user interaction while in the coverage area.
2. Protocols and applications should be designed to conserve battery energy of the mobile terminals. In particular, transmissions from the mobile terminal to the access point should be used sparingly.
3. Transmissions to and from the Infostation should be scheduled so that users will be likely to complete desired transactions while in range of an Infostation.
4. A variety of applications should be supported, including those that are tolerant to loss of some data and those that are not; those that disseminate information of general interest and those that disseminate information intended for a particular individual or a small group; those that download information from the server to the clients, and, to a lesser extent, those that upload from clients.
5. The software infrastructure should be sufficiently modular to allow experimentation with a variety of data transmission protocols, scheduling algorithms, etc.
6. The design should facilitate development of new applications that use the underlying infrastructure.

One approach to the system design would be to build applications on top of standard Internet application layer protocols, such as http and ftp. In this approach, a mobile client would establish an Internet connection through the access point and request particular files from the Infostation server or another host. There are several problems with this approach. Establishing a connection takes time. Unless the mobile user knows what information it wants and where it is located, there will be overhead to determine this information before the client can request the files it wants. If certain information is popular, in the sense that many clients want it at the same time, the system will be congested sending individual copies of the data to each client.

Broadcasting or multicasting data items clearly allows more efficient use of one scarce resource, the wireless channel, when the items are popular, i.e., when there are several clients in range at the same time desiring the same items. Since we anticipate that many Infostation applications will involve popular information, we have based the design on data broadcasting, or, more precisely, data multicasting. We are exploring the hypothesis that multicasting is also a good approach for less popular items. Even data intended for a single user can be transmitted (suitably encrypted) via broadcast or multicast. After all, no matter what transport protocol is used, the data is ultimately broadcast through the air. Although the design is oriented toward multicasting, it is flexible enough to allow support for traditional unicast applications to be added later.

Communication between the Infostation server and mobile terminals proceeds in cycles, as shown in the timing diagram in Figure 2. At the beginning of each cycle, the server broadcasts an index listing those data items that will be transmitted during that cycle. The index includes information that clients can use to determine how or when to "listen" for each item, so they can conserve power by ignoring items they don't want. After pausing to allow clients to decide which items to listen to, the server transmits the data items.

Interspersed within this cycle are time periods during which clients can send requests, upload data, or send feedback to the server. A scheduler running on the server decides which data to transmit during each cycle and prepares the index.

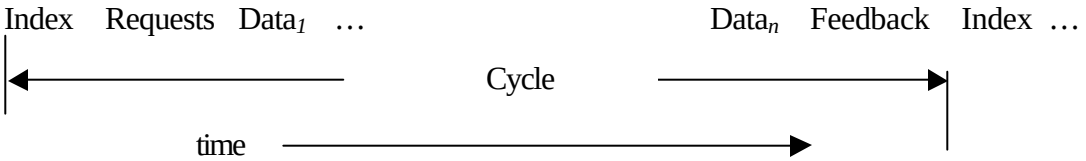

Figure 2: Broadcast cycle

When a client is within range of an Infostation transceiver, it listens to the index to see what the Infostation is offering in the current cycle. Optionally, the client may also receive a Table of Contents (TOC) containing brief summary information about items that are available, but are not necessarily scheduled for the current cycle. Infostation application programs running on the mobile terminal examine the index/TOC and determine whether there are any available data items that are of interest. The exact definition of "item of interest" and the mechanism for determining whether an item is of interest may vary from one Infostation application to another. If items of interest are available, the client listens for those items and ignores other items. If the Infostation has items of interest that are not scheduled, the client may request them. The Infostation will then schedule them for a future cycle. If the client would like to upload data, it sends a request to the Infostation; if possible, the Infostation reserves time in a subsequent cycle during which the upload can occur and informs the client via a message in the next index.

This approach conserves mobile terminal power in two ways. The client makes few, if any, transmissions (which use more power than receptions), and the client ignores packets from items that are not of interest. There are (at least) two approaches to allowing clients to ignore data items that are not of interest:

1. Use IP multicast: In this approach the index lists a multicast address for each item. The client joins the multicast groups for the items it wants; consequently it will receive data packets for those items, while other data packets will be dropped by the IP layer.
2. Explicitly control power on the client. In this mechanism the index includes information about when each item will be transmitted and the client goes into a low power mode, waking up to receive the desired packets, then going back to sleep until it's time to listen for the next index. If timing cannot be controlled sufficiently precisely, this approach can be combined with the IP multicast approach, with a client waking up shortly before the time when the item of interest is scheduled, then using the multicast address mechanism to ignore unwanted packets.

In either case, the client will "listen" again, when the next index is broadcast at the beginning of the next cycle.

A scheduler running on the Infostation server determines which data to transmit during each cycle. Files are broken into packets that, after adding headers described below, are

small enough to be transmitted without MAC layer fragmentation. As discussed below, there is a high probability that the client will not successfully receive the entire item of interest within a single cycle. This can occur due to transmission errors or if the mobile terminal leaves the coverage zone of one Infostation before it has received the whole item, then wishes to finish at another Infostation. To minimize the inefficiency caused by partial reception, it is desirable for applications to be able to listen for or request units that are smaller than whole items. Consequently, the Infostation breaks large data items into units called "chunks", that are potentially larger than packets and smaller than whole items. Each chunk appears in the index, so when there are gaps in received data items, [VSC1]clients can listen for or request these smaller units selectively. In addition to their role in power conservation, chunks facilitate fair scheduling, by allowing the server to spread out large items over several cycles so that users desiring smaller items do not incur unacceptable delays.

Assuring reliable data transfer is always an issue and is particularly important for wireless systems, where, due to channel fading, one expects the loss rate to be relatively high. There are two general approaches to handling packet losses:

1. Build applications on top of reliable transport protocols, such as TCP or various reliable multicast protocols [OBR]. In this approach, the transport layer will keep track of which packets have been successfully received and request retransmission of lost packets.
2. Build applications on top of unreliable transport protocols, such as UDP, but require them to deal with reliability at the application layer. (In this context the "application" includes the Infostation software infrastructure, not just the application programs running on the Infostation.) The UDP checksum mechanism will detect corrupted packets and drop them, but the transport layer will not take further action to keep track of which packets were lost or request retransmission.

Our Infostation design currently uses the second approach. Reliable transport protocols necessarily require receivers to send ACK or NACK messages back to the sender. Transmitting these control messages could cost the mobile terminals considerable battery energy. If a large number of mobile units are receiving information from the Infostation, contention for the shared wireless channel to send ACKs or NACKs will use additional time and battery energy[1]. Overhead entailed in establishing a connection (or joining a reliable multicast group) will also take time and energy. Using TCP would limit us to unicast, thus wasting bandwidth when several mobile terminals are interested in the same information. It is not clear that existing reliable multicast protocols are well suited to the highly dynamic Infostation setting, where mobile terminals enter and leave the coverage area at random times, and thus may drop out of multicast groups while data is being transmitted. Although we are building the initial Infostation prototype on UDP, we plan to further explore the reliable multicast approach in the future, and to compare the performance of various approaches.

[1] Readers familiar with the 802.11 standard may wonder whether the MAC layer acknowledgement mechanism will result in transmissions from the mobile terminal to the server. Because we are using the 802.11 broadcast and multicast mechanisms, there are no MAC layer acks.

There are several possible approaches to achieving (sufficient) reliability for applications built on UDP.

1. Data Carousel [ACH]: Items are broadcast repeatedly and the receiver keeps track of which packets it has received. If a packet is dropped, the receiver listens for it during a subsequent cycle.
2. Forward Error Correction (FEC) codes, such as Tornedo codes [BYE], transmit redundant data in such a way that if any *k* of *n* transmitted packets are received, the receiver can reconstruct the message. Simulations by [BYE] indicated that the FEC approach yields better response time than data carousel for traditional multi-cast applications, such as distributing a file to a fixed group of clients. However, it is not clear that the advantage will always hold up in the Infostation context, where client power consumption and efficient channel utilization may be more important metrics than response time. Popular data is being rebroadcast anyway, so filling in "holes" is often almost free and errors tend to be distributed non-uniformly based on fading characteristics of the wireless channel.
3. For some types of data, including audio, image, and video data, and for some applications, loss of some data can be tolerated. Various encoding schemes exist in which more important data is transmitted redundantly, while loss of less important data is tolerated [WAN].

3. Prototype Infostation Implementation

We have implemented a prototype Infostation system and two applications, one that delivers course notes and one that delivers music clips. The access point and PCMCIA cards are IEEE 802.11b Symbol Technologies Evolution2 products; the Infostation server is a Windows NT desktop PC with Pentium III processor, 256 MBytes of RAM and 30 Gbytes hard disk, connected to the AP by a 100Mb/s Ethernet; and the mobile terminals are Pentium III notebook computers, with 128 Mbytes of RAM running Windows 98. The software is written in C++ using the Winsock2 library. We have also ported a version of the system to a Windows CE PDA.

We are initially focusing on information services from a few points in the matrix of spatial-temporal-personal relevance shown in Table 1, namely those involving distribution of information of interest to a group of users or to the world at large. We assume that appropriate information has been cached at the Infostation. Some of our applications use a web-crawler to update the Infostation cache. In the future, we will incorporate applications that upload data from mobile terminals to the Infostation and will introduce heuristics for predicting which Infostations a user is likely to encounter and pre-fetching personal information accordingly.

The IP multicast mechanism described above is used to allow clients to listen to data selectively. To assure that all (or enough) packets are received reliably, we are initially using the data carousel approach, described above. The server adds headers, including sequence numbers, to packets. The client keeps track of which packets have been dropped by UDP then "listens" for the chunks containing those packets when they are retransmitted in subsequent cycles.

We plan to use the Infostation prototype as a test bed for exploring the advantages and disadvantages of various approaches to coping with packet loss. We suspect that there will not be a single best mechanism, but rather, that suitability of a mechanism will depend on many factors, including the nature of the Infostation application program, the information being transmitted, the channel conditions, and the battery energy available on the terminal. Consequently, we want to offer many possible mechanisms and have flexibility to add new ones. At the same time, we want to make it easy for application writers to develop new Infostation application programs, without requiring them to have detailed understanding of these mechanisms, of the intricacies of distributed programming, or of the details of the sockets library.

The software infrastructure of our Infostation system provides this flexibility. The server and client software each consist of three main components: a manager, a collection of encoders/decoders, and application programs. On the server side, the manager communicates with the scheduler, the applications, and encoders. Applications periodically decide what data items they are offering for broadcast (in some cases by running a crawler), and then register those items with the manager, which informs the scheduler. Each cycle, the manager gets an updated index from the scheduler. Based on instructions from the applications, it decides how to encode the data, passes the data through the encoders, packetizes it and transmits it on the appropriate multicast addresses. For example, if the data is to be "encoded" for the carousel method the server adds carousel headers, to each packet. Periodically, the manager gets updates to the collection of available files from the crawler and informs applications of these updates.

On the client side, the manager examines the index, selects those index entries that are relevant for active applications, and passes them to the applications. Each application examines the index entries it has received and issues requests to the manager for data items. The application programs we have considered so far assume that the available data items have been categorized according to topic. For example, in our course notes delivery application, each course corresponds to a topic and the web pages posted for that course are items. Users subscribe to particular Infostation applications and select topics in which they are interested. For example a student could subscribe to the course notes application and select CS-637 and EL-501. Each item receives a unique ID. If a modified version of that item is posted, for example a correction to homework solutions, the new version gets a new ID. For each application, the client maintains a "local index" that keeps track of which items of which topics of that application it has already received. Each time the client receives an Infostation index or Table of Contents it compares the current Infostation offerings with its local index to see whether there are any new items in any topics of interest.

The index entry for each data item is tagged with the encoding scheme used. For each data item requested by any application, the manager starts a thread that receives the item by joining the appropriate multicast group and then decoding the received data with a decoder of the appropriate type. For example, if the data has been "encoded" for the carousel method, a carousel decoder thread uses the header information to keep track of which packets have been received. At the end of each cycle, these threads send status reports to the manager. The manager uses this information, along with parameters supplied by the application, to decide whether enough of the data item has been received.

If so, it writes the data to disk, notifies the application, and terminates the decoder thread. If several cycle time periods go by without receiving an index, the manager assumes that the mobile has gone out of range of the Infostation and cleans up, terminating decoder threads and optionally, saving partially received files. When (partial or complete) files are saved, applications update their local indices, accordingly.

Application writers can use inheritance to reuse the portion of the code responsible for communication between the application and the manager. We hope this will facilitate application development. We plan to supply a library of encoders/decoders, including carousel, FEC, data encryption, and loss-tolerant source coding schemes for images, audio, and video. We also expect more ambitious application writers to develop new encoding schemes, tailored to particular families of applications and add them to the library. For example, one could imagine text-based applications in which different portions of text are tagged to indicate their importance and loss of some portions is tolerated; developers of such an application could write customized encoders that add more redundancy to more important portions of the text.

The first application we developed delivers updates on course notes to students. The Infostation manager, running on the server, communicates with a web crawler that checks specified sites (in this case, web pages for selected Polytechnic courses) periodically for new and modified files. The crawler is configured to only follow links to subdirectories of the course directories and to ignore files other than html, gif, and jpg files. For initial system testing, we crawled 14 course web pages, with a total of 396 files ranging in size from 65 to 89,980 bytes. The average file size was 6,176 bytes.

The second application delivers short clips of MP3 music files. As with the course notes application, subscribers choose topics of interest, in this case genres of music, and the client compares a local index to the broadcast index to determine whether there are new items of interest. There are two differences between this application and the course notes application: the files are much larger and users can tolerate loss of a small number of packets.

4. Experience

In addition to small-scale tests in our lab, we demonstrated the system at a University function. Seventy students participated in the demonstration, with up to ten simultaneous users. Each student subscribed to one course (from ten choices, a subset of the fourteen web pages crawled), to one genre of music (from six choices), and to a “name that tune” game. For each genre of music, the Infostation was offering one MP3 file with roughly 30 seconds of music, encoded using 64 kbit/s encoding, yielding approximately 240 kbytes per file. The “name-that-tune” game involved 6 two-second clips of songs, encoded at 128 kbit/s, yielding an average size of about 30 kbytes per file. The course notes files had average size of about 6 kbytes. A round-robin schedule was used in which half the course notes, half the larger MP3 files and half the smaller MP3 files were transmitted each cycle. In order to avoid overflowing buffers in the hardware that we used, long pauses (5ms) were inserted between packet transmissions, which substantially diminished the effective data rate.

Data were collected to indicate the percentage of each file that had been successfully received at the end of each cycle. Most of the course notes files were successfully received within one or two cycles. The histogram in Figure 3 shows the number of (larger) MP3 files that were successfully received after *n* cycles, as a function of *n*. The leftmost bar represents the number of clients (out of 70) that received 100% of the packets within *n* cycles. The middle bar and rightmost bar show analogous data for reception of 95% and 90% of the packets, respectively. The bars labeled *n=0* at the far right of the histogram represent users who did not successfully receive their files, because they aborted the application very shortly after entering the coverage zone. Figure 4 shows analogous data for the 420 downloads of the smaller "name-that-tune" MP-3 files. Note that each item was only transmitted every other cycle. Most song files were received 100% successfully within six cycles (within three cycles in which they were transmitted). Most name-that-tune files were received 100% successfully within two cycles.

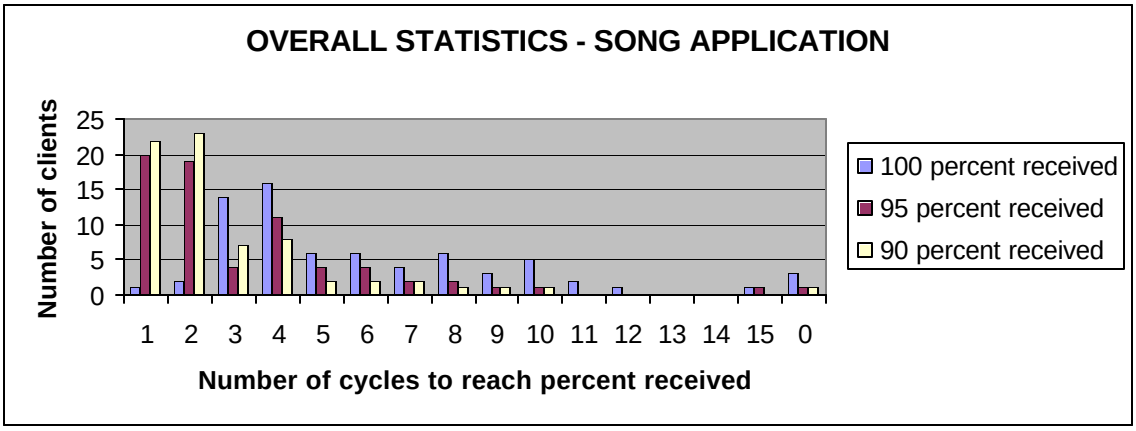

Figure 3: Number of cycles required to receive 30 Kbyte music files

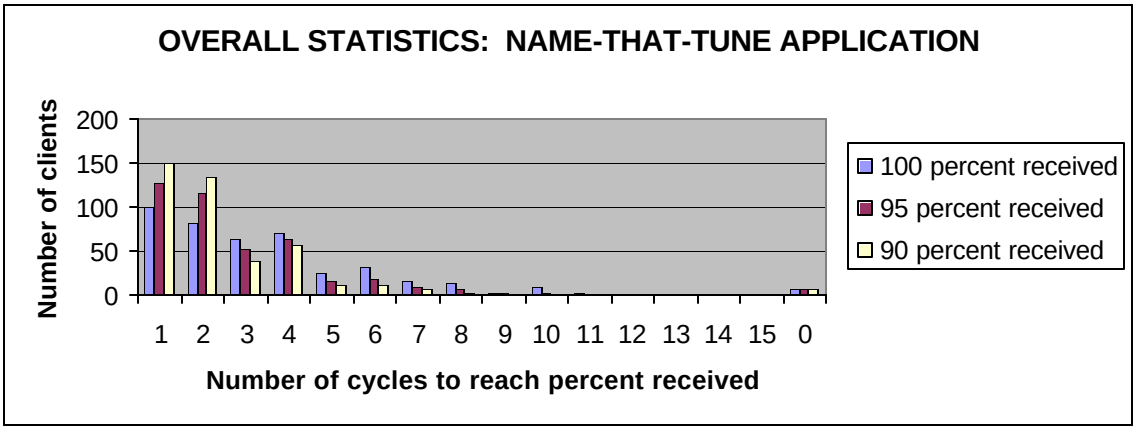

Figure 4: Number of cycles required to receive 240 Kbyte music files

We plan to do more extensive analysis and measurement of various aspects of system performance, including effective data rate, probability that clients will receive the desired items while in the coverage area, and power consumption by the mobile units. There are many parameters to tune, and there are complex trade-offs between these choices. For example, to avoid overflowing buffers on the AP and on the receiver, the server inserts delays between packet transmissions. If these delays are too short, many packets will be dropped, and clients will have to wait for many cycles to receive complete items; if the delays are too long, bandwidth is wasted. The optimal delay depends, among other factors, on the rate at which the data is being transmitted. IEEE 802.11b supports rates of 1, 2, 5.5, and 11 Mb/s. For unicast traffic the rate can vary, based on estimates of signal quality. For broadcast and multicast traffic, the rate is fixed. Selecting a higher data rate allows one to decrease the delays between packets, but also decreases the coverage area, and consequently the mobile may not be in range long enough to receive the desired items.

5. Ongoing Work

Our initial Infostation prototype work has provided a platform on which we can begin to experimentally evaluate a variety of issues. It has also helped us to identify several research issues and several system development challenges.

We plan to consider light-weight protocols for various aspects of coordination between mobile terminals and the Infostation. For example, in our initial prototype demonstration, we assigned appropriate IP addresses to each mobile terminal beforehand. This is not realistic. Alternatively, mobile terminals could receive IP addresses dynamically, upon arrival at an Infostation, by using DHCP. However, DHCP is quite slow, so this approach could waste valuable time. If the mobile terminal is only communicating via Infostation operations, it may be feasible to relax some of the constraints underlying DHCP, including the constraint that each terminal has a unique address. We are exploring ways to increase efficiency by relaxing such constraints. Similarly, the full power of IGMP is probably not needed for terminals to affiliate with multicast groups.

A second group of protocol issues involves extending the capability of the system to enable a wider range of applications, including upload applications and connection-oriented applications. Key issues here are how clients should coordinate with the Infostation to reserve timeslots for such applications and how the client-side managers should mediate the data transfers to assure that transmissions are done at the scheduled time. Coordination among Infostations and coordination between an Infostation and a client that has not yet arrived pose additional challenges.

In addition to protocol issues, system performance is strongly influenced by decisions the Infostation makes about what data to transmit when. To that end, we are exploring various scheduling algorithms through simulation and implementation/measurement. Broadcast scheduling has been fairly widely studied, for example in references [ACH,LEE,STA,AM,IMI]. Our research in this area differs from previous work in a number of respects. We plan to develop and incorporate performance measures that are

more appropriate in the Infostation context, including probability that a client successfully receives the desired data while in range of the Infostation, measures based on clients' power consumption and measures that incorporate some information about the perceived urgency that the client receives the data. We also plan to incorporate realistic assumptions about the Infostation environment, including transmission error models that reflect various types of interference to which the wireless channel is vulnerable. Results of preliminary simulation studies are reported in [COR].

Planned system development work includes improving the effective data rate by tuning system parameters, implementation of techniques to conserve power by sleeping and waking up, incorporation of appropriate security mechanisms, and development of new applications. In addition, as solutions to the above research challenges emerge, we will implement them and evaluate them through experimentation.

Acknowledgements

Vladislav Shkapenyuk, Atiyah Conry, and Miriam Adlerstein implemented the Infostation prototype and, along with Tom Cortina, Tim Moors, Gleb Naumovich, Torsten Suel, and Malathi Veeraraghavan, contributed to the system design. Work was supported in part by the New York State Office of Science, Technology, and Academic Research (NYSTAR) through the Center for Advanced Technology in Telecommunications. Symbol Technologies provided equipment and technical support.

References:

[ACH] S. Acharya, M. Franklin, and S. Zdonik, Balancing push and pull for data broadcast, in: *Proc. of ACM SIGMOD Int. Conference on Management of Data* (1997), pp. 183-194.

[CAI] J. Cai and D.J. Goodman, "General Packet Radio Service in GSM", *IEEE Communications Magazine*, Vol. 35, No. 10, pp. 122-131, October 1997

[COR] T. Cortina, P.G. Frankl, and D.J. Goodman, "A Simulation Study of an Infostation-based Data Dissemination Service for Universities", Polytechic University CATT Technical Report, May 2000.

[FBB] R.H. Frenkiel., B.R. Badrinath, J. Borras, R.D. Yates, "The Infostations Challenge: Balancing Cost and Ubiquity in Delivering Wireless Data," *IEEE Personal Communications*, Apr. 2000, pp. 66 -71.

[FRE] R.H. Frenkiel and T. Imielinski, "Infostations: the joy of 'many-time, many-where' communications," WINLAB Technical Report TR-119, April 1996.

[GBM] D.J. Goodman, J. Borras, N. B. Mandayam, and R. Yates, "Infostations: A new system model for data and messaging services", *IEEE Vehicular Technology Conference*, May 1997.

[GOO] D. J. Goodman, "The Wireless Internet: Promises and Challenges" *Computer*, Vol. 33, No. 7, July 2000, pp. 36 – 41.

[IMI] T. Imielinski, S. Viswanathan, and B.R. Badrinath, "Energy efficient indexing on air", *Proc. of ACM SIGMOD Int. Conference on Management of Data* (1994), pp. 25-36.

[LEA] N. Leavitt, "Will WAP Deliver the Wireless Internet", *Computer*, Vol. 33, No. 5, May 2000, pp. 16- 20.

[LEE] W.C. Lee, Q. Hu, and D.L. Lee, "A study on channel allocation for data dissemination in mobile computing environments", *Mobile Networks and Applications* 4 (1999), pp. 117-129.

[NAN] S. Nanda et al., "Adaptation Techniques in Wireless Packet Data Services", *IEEE Communications Magazine,* Volume 38, Number 1, January 2000, pp. 54 – 64.

[STA] K. Stathatos, N. Roussopoulos and J. Baras, "Adaptive data broadcast in hybrid networks", *Proc. of 23rd Very Large Data Base (VLDB) Conference* (1997), pp. 326-335.

[OBR] K. Obraczka, "Multicast Transport Protocols: A Survey and Taxonomy", IEEE Communications Magazine, Jan. 1998.

[WAN] Y. Wang, S. Wenger, J. Wen, and A.G. Katsaggelos, "Error resilient video coding techniques", IEEE Signal Processing Magazine, vol. 17, No. 4, pp. 61 – 82, July 2000.

Page: 7
[VSC1]After reading further, I realized that this is not what you had in mind in breaking items into smaller units – maybe you could explain it in somewhat more detail. I left my inserted sentences in place because they might be helpful to a reader in a later context.