Enabling Pervasive Mobile Applications with the FM Radio Broadcast Data System

Ahmad Rahmati[1], Lin Zhong[1], Venu Vasudevan[2], Jehan Wickramasuriya[2], Daniel Stewart[3]

[1]Dept. of Electrical & Computer Engineering
Rice University
Houston, TX
{rahmati, lzhong}@rice.edu

[2]Motorola Labs
1301 E.Algonquin Road
Schaumburg, IL
{venu.vasudevan, jehan}@motorola.com

[3]Qualcomm, Inc.
8041 Arco Corporate Drive
Raleigh, NC
stewartd@qualcomm.com

ABSTRACT

Many existing electronic devices lack data connectivity but carry an FM radio receiver. Such devices include media players, vehicular audio systems, low-end mobile phones, and mobile phones whose owners cannot afford data plans. We observe that the highly available FM radio data system (RDS) provides a low-rate digital broadcast channel that is specific to the radio station an FM receiver tunes to. While RDS is mainly intended for delivering simple information about the station and current program, we argue that it can be employed to enable a broad range of new applications and enhance existing ones. In this position paper, we discuss a number of applications that can be enabled or enhanced by RDS, and analyze the challenges evolved. We then present RDS-Link, a protocol to efficiently transfer broadcast data over RDS, and characterize its performance under real-life settings.

1. INTRODUCTION

FM radio enjoys almost ubiquitous availability and provides relatively long range coverage, spanning entire metropolitan areas. Not only most vehicles carry an FM receiver, but so do many portable entertainment devices, e.g. the Microsoft Zune, the new Apple iPod Nano, and many phones, e.g. the Motorola Ming. The latest iPhone and iPod Touch models are also equipped with an FM receiver, and the software to enable it is reportedly under development [1]. More importantly, FM radio is a more popular feature for mobile phones than the cellular data service in developing markets due to its low cost. For example, 22% of low-income owners of mobile phones in Thailand use their phones to listen to FM radio, vs. 2% for Internet access [2].

We observe that the Radio Data System (RDS) and the virtually identical Radio Broadcast Data System, (RBDS) provide a low-rate digital broadcast channel alongside each traditional FM broadcast channel; and many FM receiver chipsets sport a RDS decoder, and RDS is already implemented in many radio stations. With RDS, when an FM receiver tunes into a station, it will receive not only the traditional analog radio broadcast, but the RDS stream too. The main design goal of RDS is to broadcast metadata regarding the radio program, such as station and program name. However, the RDS standard also allows for the development of custom applications, such as proprietary traffic information used in some GPS navigation devices.

In this position paper, we seek to demonstrate that it is feasible to utilize RDS beyond its originally intended usage to enable new mobile services and enhance existing ones. The new services may be supplemental or independent from the broadcasted FM radio program. Supplemental content could enhance the FM radio program by broadcasting program or station-specific data, e.g. lyrics, program guides, and small images. Such supplemental content can further enable on-line social networking to promote listener experience and loyalty. Independent content may be data services, e.g. news, weather, traffic, job postings, or large scale queries, e.g. polls or participatory sensing.

Yet, we identify three categories of challenges regarding realizing the RDS-based applications above. First, network challenges: the transmission of data over RDS and characterizing the reliability, performance, and energy efficiency of RDS on mobile devices in real-life settings. Second, challenges for radio stations: how can they maximize their utility while maintaining customer loyalty? Third, challenges for clients: how can they maximize their reward while minimizing energy consumption? While we will discuss all three categories of challenges, we will provide solutions to the network challenges in this paper.

In this work, we tackle the network challenges through an efficient protocol for downlink data transfer on RDS and characterize its reliability and performance in real-life conditions. To the best of our knowledge, we make the following contributions.

- We discuss applications of RDS based data services on mobile systems and research challenges thereby invited.
- We design and implement a low-overhead protocol, RDS-Link, to enable applications to subscribe and receive data over RDS. RDS-Link supports graceful degradation, power management, and scheduling, and achieves a net data rate of to 17.4 bytes per second, out of a total raw data rate of 28.7 bytes per second.
- We present a characterization of the reliability, performance, and energy efficiency of RDS-Link in real-life settings.

The remainder of this paper is organized as follows. In Sections 2 and 3, we present related work and provide an overview of RDS. In Section 4, we present several applications that can benefit from RDS based data broadcasting and in Section 5 we present several identified challenges for RDS based applications. We provide details on the RDS data link layer, present RDS-Link for transferring data over RDS, and evaluate its performance in real-life settings in Section 6, and conclude in Section 7.

2. RELATED WORK

Transferring digital data over sidebands alongside FM radio broadcasts is not limited to RDS. DirectBand is a data broadcast standard developed by Microsoft, and is used in products such as the SPOT watch and GPS traffic applications utilizing MSN Direct Services [3, 4]. DirectBand uses a different subcarrier and a wider channel width compared to RDS, and provides a higher data rate. DirectBand and RDS can coexist; however DirectBand requires additional hardware as most existing FM receivers are unable to

HotMobile'10, February 22–23, 2010, Annapolis, MD, USA.

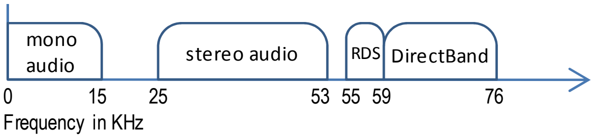

Figure 1. Right half side of demodulated FM radio spectrum

decode DirectBand transmissions. This can negatively impact the success of DirectBand compared to RDS. Furthermore, DirectBand is designed so that each user listens to only one station. However, we assume that users and applications alike can choose between all available radio stations.

Digital broadcast can also be realized using cellular data services or Wi-Fi hotspots. However, cellular data services are unavailable on many portable devices, and require expensive subscriptions and therefore are still luxury to many mobile users. On the other hand, Wi-Fi hotspots have much more limited coverage and are power-hungry for continuous connectivity [5]. From this perspective, RDS is truly complementary to cellular data services and Wi-Fi, with a low data rate, low power, low cost, and high availability. The RDS standard allows custom applications, and some devices use the RDS for proprietary data application. For example, a number of GPS devices use the RDS standard for receiving traffic updates [6]. Next generation programmable communication thermostats (PCTs) have been proposed, e.g. in California [7], to utilize RDS broadcasts. In contrast, we provide insight into a much broader range of applications that will benefit from RDS broadcast. In addition, we present a protocol, called RDS-Link, to transfer data chunks for various new and existing applications over RDS, and provide a characterization of the reliability, performance, and energy efficiency of RDS on mobile devices in real-life settings.

Nokia Visual Radio [8] uses a cellular data connection to provide graphics and text as value added services for FM radio. In contrast, we focus on utilizing RDS broadcast, either independent of or supplemental to a cellular data connection.

3. OVERVIEW OF RDS

We next provide an overview of RDS.

3.1 FM Radio and RDS

FM radio uses the UHF band (87.5 to 108.0MHz). Therefore, it has a line-of-sight range of up to 80 to 160 Kilometers, and can cover major metropolitan areas. FM stations utilize Frequency Division Multiple Access (FDMA) to share the UHF band. An FM transmitter may broadcast one or more *channels*. Adjacent channels are 200 KHz apart. In the rest of the paper, we assume a station utilizes one channel and therefore we use "station" and "channel" interchangeably. Monaural FM stations use only 15 KHz of each side of the channel center frequency, while stereo broadcasts use 53 KHz. The remainder, up to approximately 75 KHz may be utilized for sideband applications, as shown in Figure 1. For example, RDS utilizes the 55 KHz to 59 KHz region, and Microsoft DirectBand utilizes the 58.65 KHz to 76.65 KHz region for a raw broadcast data rate of 1500 bytes per second.

RDS uses Phase Shift Keying (PSK) at a physical-layer data rate of 1187.5 bps. It utilizes error detection and correction, reducing the effective data rate to approximately 731 bps.

3.2 RDS Data Fields

RDS provides for the broadcast of both custom data fields and standard metadata for radio stations. These include:

- A 16 bit Program ID (PI) uniquely assigned to each radio station. In the US, each radio station has unique four-character identifier known as the *call sign*, which can be decoded from the PI.
- A 8 character Program Service (PS), which typically contains the station name
- A 64-character Radio Text (RT), which typically contains the artist name and song tile.
- A 5 bit Program Type (PTY), from a list of predefined radio program types (e.g. news, talk, rock)
- Custom data fields, known as Open Data Applications (ODA), for new services and applications.

3.3 RDS Data Link Layer

The RDS data link layer sends data in the form of *blocks*. Each block consists of 16 bits of data, along with 10 bits of error detection, correction and synchronization overhead. Every four blocks constitute a *group* (Figure 2). Every group can be of version A or version B. In this work, we assume transmissions are always version A[1]. Every group contains a 16-bit Program Identifier (PI), A 5-bit Program Type (PTY), a 1-bit version code (B_0), a 1-bit Traffic Program (TP) identifier to indicate whether the radio station is a special traffic bulletin broadcast station, and a 4-bit Group type, showing the content type of the remaining 37 bits of the group. The remaining bits may be used to transmit, the whole or part of one of several predefined metadata or custom information fields. The raw data rate for RDS is 1187.5 bps or 11.4 groups per second. Error correction reduces the data rate to approximately 731 bps or 91 bytes per second. Of the 11.4 groups per second, 2 are reserved for PS and 3.2 for RT, leaving 6.2 groups per second available for a custom applications.

3.4 RDS Power Consumption

If the user is already listening to the FM radio, the RDS data comes with virtually no additional energy cost. Otherwise, the system has to keep the FM receiver powered on to receive RDS. The Motorola Ming A1200e phone used in our study is equipped with a Philips TEA5764 FM stereo radio receiver. The TEA5764 consumes approximately 39 mW in operating mode. According to our measurements, this is comparable to the power consumption of an idle GSM phone which typically ranges from 17 to 100 mW [5]. Assuming the FM radio module is left always on, at 39 mW, RDS consumes approximately 20% of a typical phone battery (1250 mAh, 3.6 V) per day.

3.5 Advantages and Limitations of RDS

There are several benefits of using RDS for digital broadcast in comparison to that based on cellular and Wi-Fi:

- Hardware availability and low cost: Portable devices are increasingly equipped with FM radio, and RDS decoding capabilities are already built into many FM receiver chipsets. For such devices, it is only necessary to add software support.
- Low power consumption: An FM receiver consumes little power, e.g. in our case 39 mW, comparable to an idle GSM radio.
- Universal availability. FM radio has a long range and enjoys truly universal coverage, even more than cellular networks.
- No network charges: In contrast to cellular data services, RDS does not require a monthly subscription and/or per KB fees.
- Multiple subscription options: RDS enables the user to choose between many different FM radio stations. While current chipsets can tune to only one station at every single time, the user has the choice to subscribe to RDS broadcasts from any one or more stations.

[1] Version B groups are similar to version A, but transmit the PI twice per group, leaving 21 bits for other items, vs. 37 bits in version A.

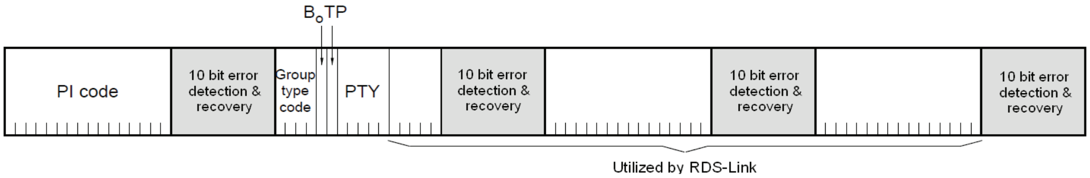

Figure 2. RDS Group structure: Each Group consists of four 16-bit Blocks (based on a figure from US RDBS Standard, 1998)

RDS is not the only digital sideband broadcast standard for FM Radio. For example, the proprietary DirectBand standard by Microsoft offers considerably higher data rates, but requires custom hardware. In contrast, RDS is already implemented in many FM receiver chipsets and radio stations.

RDS is not without limitations. First, it is unidirectional, and can only provide a downlink channel. Second, it has a broadcast nature; each radio station can only broadcast one RDS channel that is received by all radios tuned into that station. Third, the FM receiver can receive the RDS from a channel only if it tunes to the channel, and existing FM receivers can only tune to one channel at a time. Fourth, it has a very low data rate. We will see in Section 6.2 that our RDS-Link protocol can transfer data at approximately 1KB per minute. Yet, even with these limitations, many low data rate, delay tolerant applications can be enabled by RDS.

4. RDS-ENABLED APPLICATIONS

We next describe a representative set of applications that can be enabled or enhanced by an RDS-based data broadcast. RDS constitutes a low-rate, low-cost data broadcast channel for the radio station (Figure 3). It can be used to enhance the user experience of the corresponding FM program, attract more listeners, broadcast community-relevant information, and be combined with existing wireless interfaces to improve system efficiency. In addition, we show that not only can new applications may be designed to utilize RDS directly, but RDS can be used to store web pages on the device to be utilized by a web browser, enabling many existing services over RDS.

4.1 Value Added Services for FM Radio

A radio station can utilize its RDS data broadcast channel to provide low bandwidth, delay-tolerant value added services to their audience, beyond the standard station name and song title metadata already provided by RDS. For example, a station can broadcast the Electronic Program Guide (EPG) for scheduled radio programs. An EPG can enable users to plan for listening and/or recording future programs, in addition to viewing more information about current programs. As another example, items such as lyrics, subtitles, and album art could be broadcast via RDS. Lyrics and small album art are already embedded into many popular music formats and supported by music stores, e.g. MP3 and iTunes.

This can add to the value of basic FM radio services, whose functionality has been unchanged in decades, and enable a myriad of new applications. For example, EPG and lyrics/subtitles can enable users to automatically and easily provide feedback (e.g. voting) to the broadcasters, share their experience through social networking sites, search for interesting programs among different stations and even automatically record them for future listening. The value of such services alongside radio broadcasts has been shown for Digital Audio Broadcast [9].

4.2 Promoting Station Loyalty

Radio stations can employ RDS to provide incentives to not only attract listeners to their FM programs, but to encourage them to remain tuned in during commercials, in a noninvasive manner. Although radio advertising is a multi-billion dollar business, its revenue is decreasing, putting pressure on radio stations. In the US, total radio station advertisement revenue dropped from $21.3

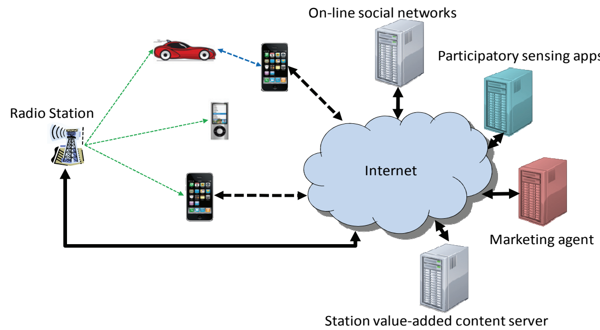

Figure 3. RDS can enable new applications on mobile devices with a low-rate but highly available digital broadcast channel for each radio station

billion in 2007 to $19.5 billion in 2008 [10]. A station can utilize RDS to broadcast coupons and special offers to its audience. Such incentives can also supplement on-air broadcast advertisements. For example, the device can display further information (e.g. text, small images, URLs) or special offers (e.g. coupons) related to an on-air advertisement. On-air advertisements can be synchronized with RDS incentives to discourage a station's audience to tune away from the station during times when advertisements are being broadcast.

Note that similar ideas have been exploited by TV broadcast after digital TV recording devices such as TiVo have been widely adopted. For example, an advertiser can choose to superpose a coupon code in the commercial to encourage TV viewers not to skip the commercial. While this is viable for TV commercials because of the spatial nature of human vision and TV content, it would be annoying to superpose incentive information in the audio stream of an FM program. Moreover, such practice requires changes to be made to the commercial content. In contrast, RDS provides a complete separate channel and allows much more flexibility in incentive deliveries. For example, coupon codes can be easily changed over time without changing the audio content.

4.3 Data Services

Given the virtually universal availability of FM radio, RDS can support many delay tolerant low rate data services in lieu of or in addition to cellular or Wi-Fi connectivity. Examples include broadcasting local and regional information such as weather forecasts and news, which may be presented to the user in the form of text or even audio through text-to-speech conversion. Such services will be particularly valuable for low-cost portable devices such as media players, which often lack data connectivity through cellular or Wi-Fi. Example devices equipped with an FM radio include the new Apple iPod Nano and the Microsoft Zune. The RDS channel could also be used to assist Wi-Fi, for example by announcing the availability and locations of open access points to increase the energy efficiency of Wi-Fi [5].

Furthermore, while mobile data services provide high utility internet connectivity for phones, service provisioning for emerging markets remains expensive and prohibitive, limiting the usage of mobile data services [2]. In this case, RDS can be used to broadcast community information such as daily prices for agricultural businesses and employment opportunities in nearby villages and towns. Moreover, since mobile devices are often shared among community members [2, 11], these services will serve many more users than the actual owners of FM radio equipped phones and portable devices.

As previously mentioned, RDS has already been employed for the broadcast of proprietary data, such as traffic information for GPS navigation devices [3, 4, 6]. In contrast, we provide insight into other possible applications of RDS data transfer, and present and evaluate a data transfer protocol over RDS in real-life settings. Microsoft DirectBand also offers broadcast data services alongside FM radio at a considerably higher data rate, but requires custom hardware while RDS is already implemented in many FM receivers and radio stations.

4.4 Query-to-Many, Response-from-Few

The broadcast nature of RDS matches very well with sensing and data gathering applications that require broadcasting a query to a large audience. One example is that of participatory sensing [12, 13] using the myriad of sensors in portable devices; a phone may be equipped with a camera, accelerometer (motion sensor), GPS, and thermometer, and can also record ambient sound [14] and measure cellular and sometimes Wi-Fi network conditions. Further, the human user may also provide feedback and polls regarding the FM radio programming.

In comparison to a cellular network based query system, the broadcast requirements of RDS remains constant irrespective of the number of users. Therefore RDS sensing solutions are extremely scalable. Furthermore, while network charges (e.g. text message or data services) may be prohibitive for receiving queries where the user is highly unlikely to answer, if a user is able and willing to answer a query, the network charges or the human effort can be can be well worth the response. In this case, users who respond may be compensated for their service either in the form of monetary awards or, as the case or radio station polls, in the form of a more enjoyable radio program experience.

A radio station may broadcast queries of interest to itself, or receive compensation for providing this service for other entities. Even though RDS cannot provide the uplink channel for the responses, participants can submit their response through traditional channels. These include phone lines, text messages, and data services, and may be either manual (e.g. if the query is received on a non-phone device) or automatic (i.e. on a phone with supporting hardware and software). Users who respond may be compensated for their service either in the form of monetary awards or, as the case or radio station polls, in the form of a more enjoyable radio program experience.

5. TECHNICAL CHALLENGES

We next identify several key technical challenges in realizing the RDS-based applications above.

5.1 Network Challenges

RDS has been successfully used for decades to transmit standard metadata such as station and program name. Station and program information remain unchanged for long periods of time and therefore are transmitted repeatedly and are unlikely to suffer from reliability issues. Yet, it is unclear how reliability issues would affect data application in real life settings. Furthermore, while RDS allows custom data items as long as minimum requirements for standard metadata are met, it does not provide a protocol to transfer a stream or contiguous chunks of data. Due to the unreliable nature of the radio link and the low data rate of RDS, it is crucial for any protocol design to have a low overhead and gracefully degrade in case of transmission errors.

In the rest of this paper, we mainly focus on these challenges, presenting our RDS-Link protocol and characterizing the reliability, performance, and energy efficiency of RDS in real-life settings.

5.2 Challenges for Radio Stations

There are many open challenges for broadcasters to utilize the low data rate RDS for data services. We have identified five such challenges. First, the station must encourage customer loyalty using RDS broadcasts, as explained in Section 4. While a station can provide incentives for remaining tuned into that station, it must be careful to not penalize users who lose coverage or tune to other stations to the extent that that it discourages the users from returning to their station. Second, the station should encourage users to receive (and not skip) broadcasts valuable to the station (e.g. ads) in order to receive data valuable to the users. It can achieve so with a proper profit model, e.g. through a broadcaster supplied phone software that in turn allows the decoding of encrypted RDS broadcasts. Third, the station must schedule the transfer of data properly. For example, larger routinely used data such as template graphics should be broadcast at unpopular times, e.g. at night, to enhance the experience of regular data broadcasts and reduce their bandwidth requirements. Fourth, the station must prevent rogue broadcasters from impersonating it and its RDS stream. Finally, the station should strike a good balance between error correction methods and amounts, e.g. according to our performance measurements in Section 6.4.

5.3 Challenges for Clients

Clearly, clients are interested in maximizing their reward, defined as the reception of data of interest, while minimizing energy consumption. We expect that this desire opens up a myriad of research questions. For example, a client should be able to switch and schedule between stations and support multiple applications interested in different RDS data. We call this the *station scheduling* problem for RDS. For another example, the client should maximize energy savings by turning off the FM receiver without missing data of interest. Moreover, it is unclear how the system could be optimized to allow clients to quickly search through a number of channels with minimum overhead. We envision such challenges could be partially addressed by broadcasting a program schedule and a mechanism to enable the receiver to skip each item of non-interest. We will see that RDS-Link supports skipping a non-interest item.

6. RDS-LINK PROTOCOL

While RDS can act as a data link layer for custom applications, it lacks the packet caching and assembly capabilities to transfer either a stream or data chunks for the applications described in Section 4. To address this need, we have developed RDS-Link.

6.1 Overview and Design Objectives

We design the RDS-Link protocol, or *RDS-Link*, to broadcast *chunks* of data over RDS. A chunk is a block of data of a variable size that can be passed to an application or saved as an individual file. RDS-Link provides an interface between applications and the RDS enabled FM receiver (Figure 4). Applications may subscribe to certain types of data chunks from one or more radio stations. Whenever RDS-Link receives a chunk for a subscribing application, it forwards the received chunk to the application. To support existing RDS-unaware applications, such as a web browser, one can develop an RDS-aware intermediate application to receive certain chunk types from a station and save them as files. The browser or other RDS-unaware applications can then access the locally stored files through a URL.

To support a broad range of ways to utilize RDS, we ensure RDS-Link is flexible enough to support and leverage all the following cases: 1) one application may be interested in chunks from one or more stations; 2) one or more application may be interested in the same chunk; 3) an RDS station may broadcast a chunk one or more times; and 4) the RDS broadcast schedule may or may not be available beforehand.

Beyond supporting the cases described above, we have several important objectives in designing RDS-Link. First, we want to operate on existing hardware and remain fully compliant with the RDS protocol, respecting the minimum RDS transmission requirements for predefined metadata such as station and program name. Second, we want the protocol to gracefully degrade in case of errors, instead of losing or corrupting the entire data chunk being transferred or even other data chunks. Third, we want RDS-Link to have minimal overhead and complexity, even for small data chunks. Finally, we want RDS-Link to support power management of the FM receiver and support *station scheduling* as described in Section 5.3. That is, it can schedule the receiver to change stations or switch off during the broadcasts of data chunks that no application has subscribed for.

6.2 Description and Analysis

RDS-Link leverages the RDS link layer, as described in Section 3.3, to provide 37 bits per RDS-Link group. To reduce the data transmission overhead, RDS-Link provides, for each chunk, a 12-bit application type and a chunk ID of 13 bits. The application type allows for a total of 4096 preset application types per radio station (the station is identified by its PI code). The Chunk ID allows for 8192 uniquely identifiable chunks (e.g. filenames) per application. The maximum allowed length of each chunk is just under 192 KB, which is considerably large compared to the data rate of RDS. Upon completion of a chunk, RDS-Link can forward it to the appropriate application (based on PI code and application ID), or store it on disk as a file for existing applications (e.g. web browsers).

As shown in Figure 2, each RDS-Link group has 37 bits available to it. The 37 bits are used as follows:

- 24 bits for the data payload
- 12 bits for the lower part of the index
- 1 bit to indicate change-of-header

A 16 bit index, when multiplied by 3, indicates where the 24 bit (3 byte) data payload belongs in the chunk. The lower 12 bits of the index are transmitted with every group, while the higher 4 bits are transmitted with the header, described below. The change-of-header bit will enable the receiver to distinguish when a transmission is no longer related to the previously received header, even if the header was missed, e.g. due to noise. The change-of-header bit will flip whenever an updated header is transmitted, to prevent overwriting and corrupting a previously received part.

For each data chunk, we define a 48 bit header, transmitted as two groups at the special index of FFFEh- FFFFh. (address range of 2FFFAh – 2FFFFh). The header contains:

- A 13 bit chunk ID
- A 12 bit application type
- An 18 bit chunk length
- 4 bits for the higher part of the index
- A 1 bit chunk version

The header is transmitted at the beginning and end of every data chunk, and at every 32 groups in between. Repeated broadcast of the header will enable the quick identification of the chunk if the FM radio tunes into the station at the middle of the transfer, or if transmission errors cause the header part to be lost.

The chunk version bit will flip whenever a new version of a data chunk is transmitted. This will enable the receiver to distinguish when a previous transmission is repeated and when a

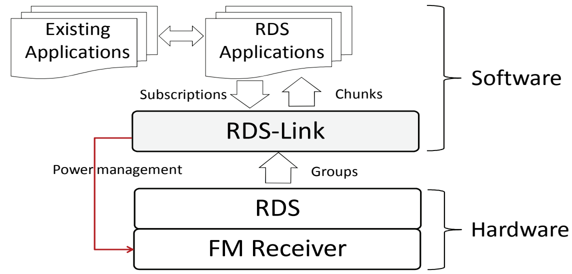

Figure 4. Relationship between RDS-Link, applications, and hardware

new version is being transmitted instead of the old one. In the former case, the receiver can use the retransmission for error correction, as will be discussed in Section 6.3.

At 6.2 groups of 37 bits available per second for a custom protocol (Section 3.3), the total raw data rate available would be about 28.7 bytes per second. At 24 bits (3 bytes) data payload per group, and with a header retransmission every 32 groups, the net RDS-Link throughput can be calculated as 17.4 bytes per second, or approximately 1 KB per minute. While this is very small in comparison to cellular data services, it is sufficient for many applications, as discussed in Section 4.

To minimize power consumption, RDS-Link allows the device to switch off the FM receiver, or to switch to another radio station during transmissions that have no application subscribed for. These could include chunks that the device has already received correctly, and are being retransmitted and chunks types where no application has been registered for. In such case, whenever the two header groups for that chunk and at least one other group are received, the device can calculate the remaining time for that chunk according to the location index and chunk size, and switch off the receiver or switch to another station for that period.

6.3 Dealing with Errors

While RDS utilizes built-in error detection and recovery, due to the nature of radio broadcasts, e.g. fading, it is not always possible to recover errors. In this case, the block checksum will fail, resulting in a failed block and/or group. There are two methods for further error recovery that may be utilized by RDS-Link. First, each file may contain its own efficient error correction code, such as those in many commercial compression formats (e.g. RAR Data Recovery Block [15]). Second, each part of a file may be transferred multiple times. Note that as long as the 1-bit file version in the file header remains unchanged, the same file is being retransmitted. With a second broadcast, the receiver can fill in missing parts due to failed groups. Upon a third transmission, the receiver can implement a reliable voting system for each received block. Multiple transfers provide a very simple mechanism for error correction at the expense of bandwidth. Our performance evaluation provides insight for application designers to choose error correction codes and/or retransmission depending on their application and design goals.

6.4 Performance Evaluation

We have measured the reliability and performance of the RDS-Link protocol in real-life settings, and found the error rate to be low (average: 11%) as long as the perceived audio quality is good.

We have used a Motorola Ming A1200e Linux-based phone that has built-in FM radio for our measurements. While the radio software provided with the phone does not support RDS, the radio chipset does support RDS. The radio chipset is controlled by an I^2C bus, and we have modified the radio software to enable and

Figure 5. Measured RDS-Link Performance: (Left) Error rate vs. perceived audio quality for indoor and outdoor measurements. Box Plot shows 25th, 50th, and 75th percentiles. Round dots indicate mean, and whiskers indicate maximum values. (Right) Outdoor measured error rates vs. distance from radio stations

read RDS blocks received by the radio. Our software enables us to evaluate the performance of the RDS-Link protocol.

We have measured the average group error rate of several commercial FM radio stations both indoors and outdoors. While we did not have access to the radio stations, we utilized radio stations already transmitting RDS to measure the number of successful and erroneously received groups for the purpose of RDS-Link. We must note that for the purpose of RDS-Link, any failed block within a group, with the exception of the first group, will result in a useless group. As shown in Figure 2, the first group always contains the PI, therefore it is not critical for decoding a single RDS-Link group. Each test was run for approximately two minutes to ensure reliable measurements. The measurement locations were in the Motorola Schaumburg campus, in the suburbs of Chicago, IL. We recorded the perceived audio quality along with each measurement, to corroborate with the results. We utilize FCC data to locate the physical location of the radio towers, and Google Earth to measure the line-of-sight distance to the stations.

We observed that the error rates of RDS-Link were tightly correlated with the perceived radio signal quality in both indoor and outdoor measurements, as shown in Figure 5 (left). When the perceived signal quality was good, the group error rate was on average 11%. This error rate is very respectable, compared to standard RDS metadata that correct errors by retransmitting the same metadata many times over the length a program, and attests to the feasibility and practicality of RDS-based applications as discussed. Our measurements suggest that the design goal for built-in FM receivers could be a good perceived signal quality.

We did not find a correlation between transmitter distance and RDS-Link reliability for our indoor measurements. However, outdoor measurements showed significant correlation, shown in Figure 5 (right). We can see that the performance for stations at distances up to 15 KM was generally good in all conditions, which is sufficient for many urban areas. We must note that traditional radios appeared to have less static compared to our Motorola Ming phone at the same locations of our measurements, leading us to believe that our measurements present a pessimistic picture of RDS-Link performance, and more sensitive implementations are capable of increased range and less errors.

7. CONCLUSION

The highly available FM radio data system (RDS) provides a low data rate digital broadcast channel alongside each FM channel. In this work, we demonstrate that it is feasible to utilize RDS to support services beyond its originally intended usage. We present a broad range of services that can be enabled or enhanced by RDS. The new services add values to existing FM broadcast services, promote station loyalty, provide low-cost data services, and support large-scale queries.

Yet, we show that research challenges exist for the network, FM stations, and receivers. In this position paper, we mainly focus on tackling the network challenges of RDS, and present RDS-Link, a simple protocol to efficiently broadcast data chunks over RDS. RDS-Link is flexible to support a broad range of applications. It degrades gracefully under transmission errors, and supports receiver power management and station scheduling. We report a characterization of the RDS-Link performance in real-life settings, and have found that its error rate is low (average: 11%) as long as the perceived audio quality is good. This further attests to the feasibility and practicality of RDS-based applications as discussed.

Most importantly, we hope our work invites more research to address other challenges toward fully realizing the potential of RDS; and we believe our work is a first step toward this direction.

ACKNOWLEDGEMENTNS

We thank our reviewers and our Shepherd, Professor Landon Cox, whose comments and shepherding helped improve this paper. This work was supported in part by a gift from Motorola and NSF Awards IIS/HCC 0803556 and CNS/CRI 0751173.

REFERENCES

[1] 9 to 5 Mac. In-house Radio.app in the works for iPhone and iPod touch. http://9to5mac.com/iPhone-fm-app, September 13, 2009.

[2] LIRNEasia. India's rural millions: Connected? Findings from a six country study of Teleuse at the Bottom of the Pyramid. http://lirneasia.net/wp-content/uploads/2009/04/coai-tabop3-mumbai-10feb09_final2.pdf, 2008.

[3] Microsoft MSN Direct. http://www.msndirect.com/, 2009.

[4] Microsoft Corporation. MSN Direct Services Overview http://msdn.microsoft.com/en-us/library/cc510527.aspx.

[5] Rahmati, A. and Zhong, L. Context-for-Wireless: Context-Sensitive Energy-Efficient Wireless Data Transfer. *Proc. Int. Conf. Mobile Systems, Applications and Services (MobiSys)*, 2007. 165-178.

[6] Garmin Garmin FM Traffic. http://www.garmin.com/traffic/fm/, 2009.

[7] California Demand Response Research Center. Programmable Communicating Thermostat (PCT) Reference Design. http://drrc.lbl.gov/pct/.

[8] Nokia Corporation Nokia Visual Radio. http://www.visualradio.com/.

[9] Geelhoed, E., Barfield, L., De Bruine, A., Hull, R. and Jeffries, H. Augmented Digital Audio Broadcast Home Trial. *HP Labs Technical Report HPL-2005-203* 2005.

[10] Radio Advertising Bureau Radio Revenue Trends. *http://www.rab.com/public/pr/yearly.cfm*, 2009.

[11] Liu, Y., Rahmati, A., Huang, Y., Jang, H., Zhong, L., Zhang, Y. and Zhang, S. xShare: Supporting Impromptu Sharing of Mobile Phones. *Proc. Int. Conf. Mobile Systems, Applications and Services (MobiSys)*, 2009.

[12] Burke, J., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S. and Srivastava, M.B. Participatory sensing. *Proc. ACM Sensys WSW Workshop*, 2006.

[13] Cornelius, C., Kapadia, A., Kotz, D., Peebles, D., Shin, M. and Triandopoulos, N. Anonysense: privacy-aware people-centric sensing *Proceeding of the 6th international conference on Mobile systems, applications, and services*, ACM, Breckenridge, CO, USA, 2008.

[14] Lu, H., Pan, W., Lane, N.D., Choudhury, T. and Campbell, A.T. SoundSense: scalable sound sensing for people-centric applications on mobile phones. *Proc. Int. Conf. Mobile Systems, Applications and Services (MobiSys)*, 2009.

[15] RAR Lab. The RAR Archiver. http://www.rarlab.com.