The University of Kansas

Technical Report

Data Organization, Processing, and Display for a Robust Millimeter Wave Metropolitan Mesh Network

Jake Foiles

ITTC-FY2008-TR-46800-01

April 2008

Project Sponsor:
Sprint

Data Organization, Processing, and Display for a Robust Millimeter Wave Metropolitan Mesh Network

EECS Departmental Honors Research Report

Jake Foiles

Abstract

This report will explain the organization, processing, display, and analysis of weather and link performance data collected to investigate the feasibility of millimeter wave wireless networks. These millimeter wave links are susceptible to attenuation due to precipitation, and this vulnerability may be avoided by the application of new resilient routing algorithms. To better understand the effect that precipitation has on millimeter wave networks, a single link was studied first. Two different ultra-high frequency radios were used, one with Forward Error Correction (FEC) implemented and one without. Comparing the performance data from these radios to local weather data would show what effect weather had on the performance of a millimeter wave link. To facilitate this study, the data from the weather instruments and radios must first be collected and organized, then processed, and finally displayed.

Contents

List of Figures

List of Tables

1 Introduction

The primary focus of this research effort was to investigate the possibility of using a mesh network of fixed-location millimeter wave links to deliver wireless data service to a metropolitan-sized area. As the demand for high-bandwidth wireless communications continues to increase, high-capacity backhaul network solutions are needed. These millimeter wave mesh networks could prove to be practical solutions. A major part of making this a viable and cost-effective alternative to fiber-optic connections is making it robust, even in the face of adverse weather. Millimeter wave links are impacted by storm disruptions, such as rain attenuation. [1]

One aspect of this effort was to propose a dynamic, predictive routing algorithm that could be used to route traffic around incoming storms, thereby reducing or eliminating link degradation. The other part of this research was concerned with collecting and analyzing local weather data and the performance of a local millimeter wave wireless link. This data and the conclusions made from it can be used to evaluate the actual performance of millimeter wave networks in this region and climate. This data can also be used as input to the predictive routing algorithm simulation to see the results of actual weather storms passing through the mesh network. This report will focus on the collection, organization, processing, and analysis and correlation of the weather and link performance data.

2 Infrastructure

To collect weather data for this research project, several weather stations using WXT510 instruments were set up around the Lawrence area, each equipped with sensors capable of recording temperature, humidity, wind, and most importantly, precipitation [2]. To collect link performance data, two ultra-high frequency radio links were established across Lawrence. One of the links utilizes Forward Error Correction (FEC) to reduce the number of packets lost in transmission, where as the other link does not have this feature. Both links cover a distance of about 9 km. Figure 1 shows the locations of the weather stations and the radio links.

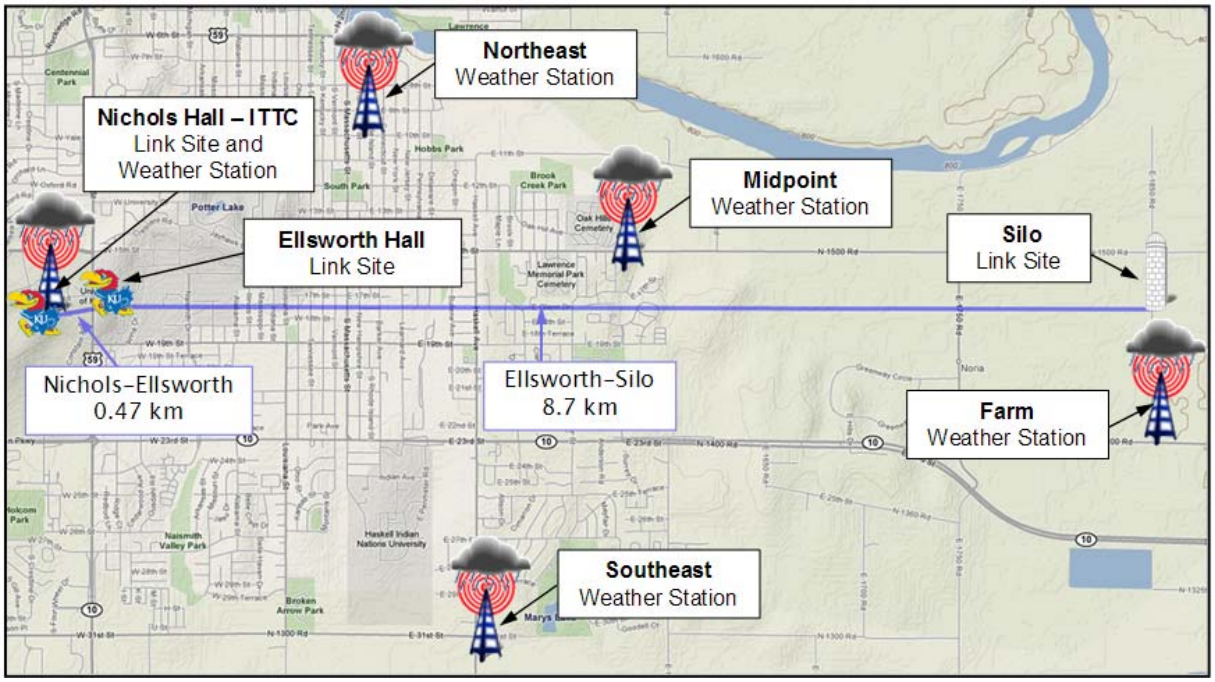


Figure 1. Locations of Weather Stations and Radio Links

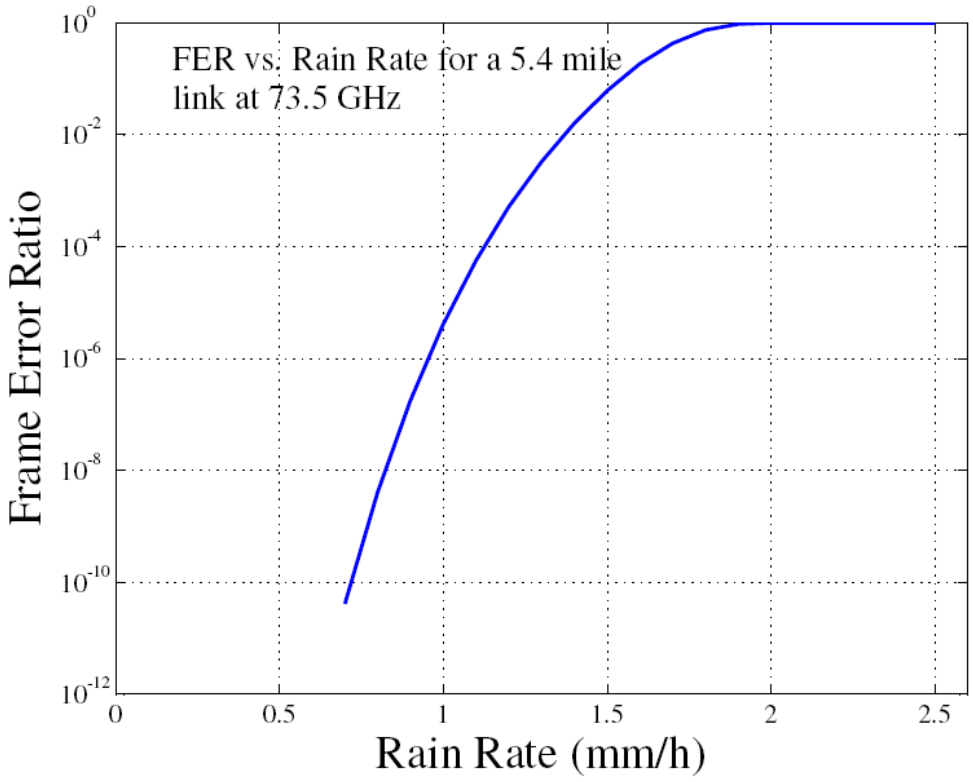
The most relevant type of data to this research that the weather stations collect is the Rain Rate (RR). This quantity is a measure of how much rain (in mm/hr) is falling as measured by the weather station sensors. The RR is measured and recorded every 30 seconds at each of the five weather stations. A link performance measurement of equal value is the Frame Error Rate (FER). The FER is the percentage of data frames that are lost during every 30 second interval. On average, around 3,348,000 frames of size 512 bytes are transmitted every 30 seconds. Ten seconds are required to process and record this and other link performance quantities, and so FER samples are calculated every 40 seconds (30 sec measurement interval + 10 sec recording interval). These two measured quantities, RR and FER, are used as the primary means to analyze the performance of these high-frequency radios during precipitation.

3 Problem Statement

Over the course of a single day, around 14,400 (2,880 per weather station) weather samples and around 4,320 (2,160 per link) link performance samples are recorded. Thus with almost 20,000 data samples being recorded daily, an efficient way of organizing this data is essential. Additionally, this data must be easily displayed so that analysis of the results can be made.

As mentioned above, the RR gives us a way to see how much rain is falling at a specific time. The FER tells us how many of our transmitted data frames are being lost, most likely due to attenuation from weather (humidity or storms). These two quantities are linked, with higher RRs causing higher FERs [3]. Figure 2 shows this relationship for non-coherent on/off keying without FEC.

Figure 2. FER vs RR for a 5.4 mile 73.5 GHz link [4]



In order to determine the empirical relationship between the RR and FER, these two measurements must be linked by their recorded times and then compared. The problems that have to be solved in order to make this comparison include the difference between measurement intervals (every 30 sec for RR, every 40 sec for FER), random missing data points, and the shear amount of data.

And so three main problems need to be solved: efficient organization of the weather and link performance data, readily available analysis of this data, and comparison of the RR and FER quantities. The following section describes the solutions used to solve these problems.

4 Solution and Results

4.1 Data Organization

The first step in dealing with this large amount of collected data is to neatly organize it so that it can be queried and updated easily and quickly. A relational database was created to accomplish this. Using the Database Management System, MySQL [5], data tables were created to hold and organize the weather and link data. Table 1 is a short description of each table, and from this, one can see the overall architecture of the database.

Table 1. Sprint MWB Database Architecture

Table Name	Description
DOWNTIME_LOG	Holds dates and comments for days where the collected data was found to be invalid.
FER	Holds the FERs calculated from the raw link performance data (number of packets lost/number of packets transmitted)
LINK_TYPE	Reference to the different types of links. Includes both types of radio, both directions, and an aggregate total. 7 different link types in total.
LOCATIONS	Reference to the five different locations with weather collections instruments. Includes Nichols, Mid-Span, Pendleton Farm, North-East, and South-East locations. See Figure 1.
RR	Holds the RRs taken from the rain intensity numbers from the WXT510_RAW data table.
RR_VS_FER	Contains the matched up FER and RR data points. Each point is uniquely defined by its collection date and time, radio link type, and location.
SMARTBITS_RAW	Contains the raw link performance data. Example fields include link type, date, time, number of transmitted packets, number of received packets, packet length, etc.
WXT510_RAW	Contains the raw weather data. Example fields include weather collector location, average wind speed, air temperature, relative humidity, rain amount, rain duration, etc.

The implementation of this database and the following applications occurred after several months of data had already been collected. To insert this previously collected data into the database, Perl scripts were created to parse the old data files and then load the raw data into the database. The raw weather data was loaded directly into the WXT510_RAW table, and the raw link performance data was loaded into the SMARTBITS_RAW table. RR and FER were the two main parameters being analyzed, so separate tables were created to hold these values. And because of changes made periodically to the format of the data files, the Perl scripts had to be highly flexible and robust, allowing for a multitude of formats.

After the past data was successfully inserted into the database, the process was further automated by creating a daily Unix *cron* job which adds newly collected weather and link data to the database automatically. Before the implementation of the database, any analysis of the raw data was a redundant and time-consuming task. Automation has eliminated the need for much of this repetitive, manual work.

4.2 Display Applications

With the database successfully constructed and populated, methods were needed to analyze and view the data contained within it. A website (http://weather.ittc.ku.edu) was designed and implemented to host these applications. Figure 3 shows the main page of this website.

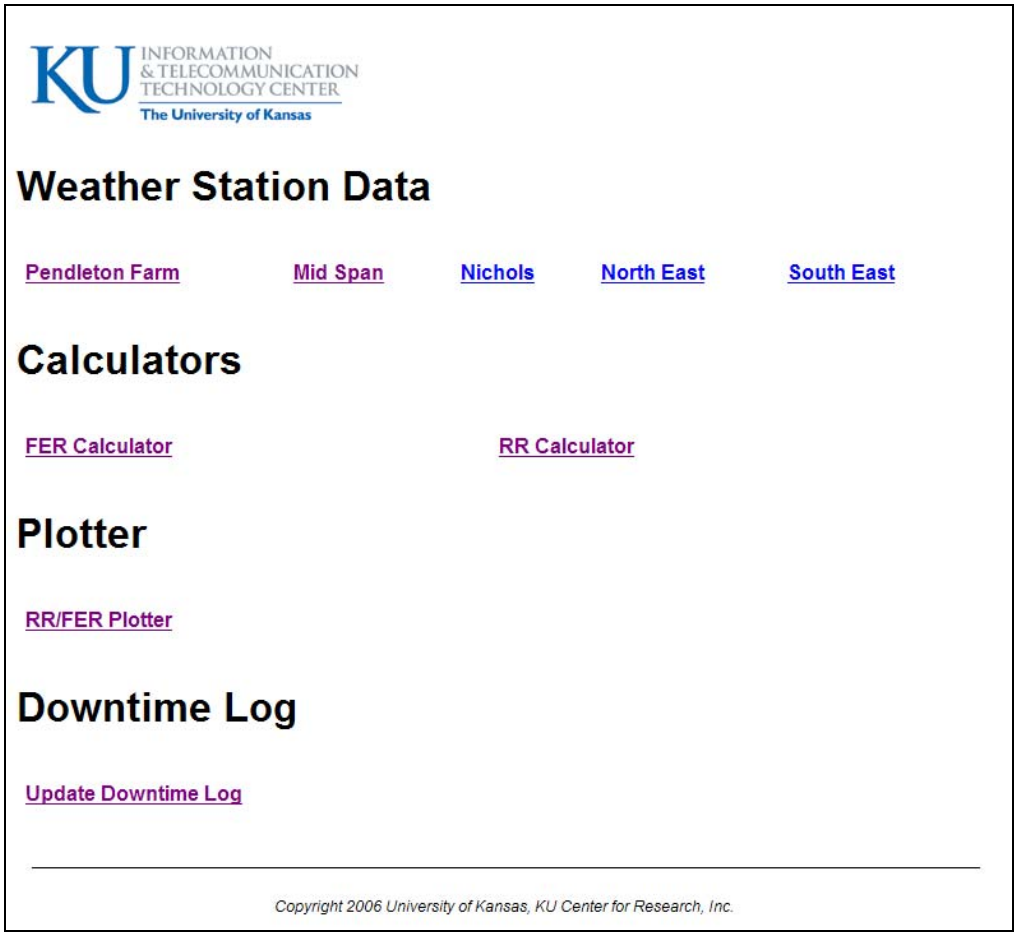


Figure 3. Main Page for Website

The first display application created calculates the availability of the wireless link over time. The availability of the link is defined as the percentage of time that the link experienced FERs less than some threshold value. A simple web form was created to take in date and time information, which link to use (the radio with FEC, without FEC, or both), and the user's definition of what makes a link "available". Figure 4 shows the input webpage for this application.

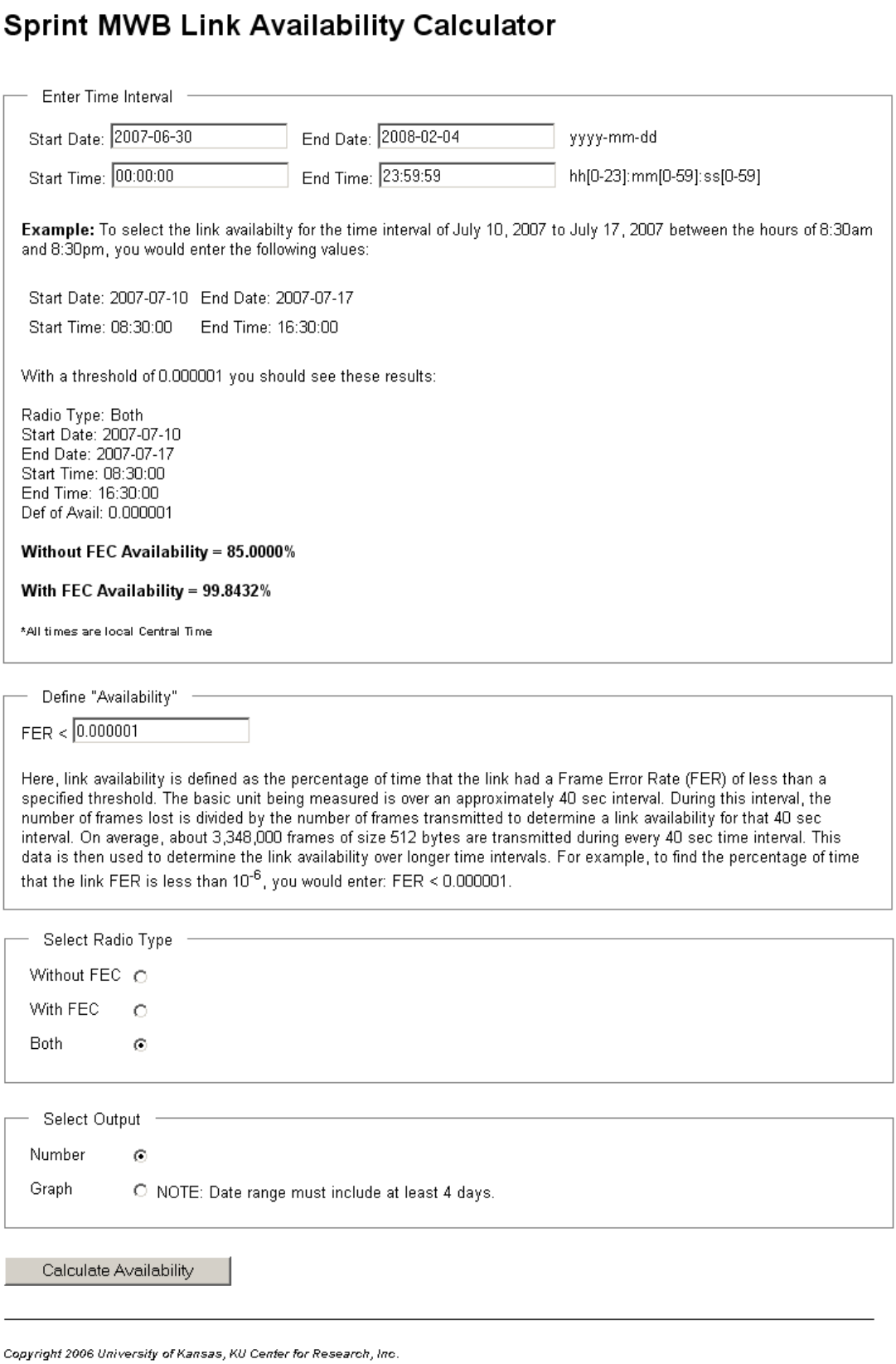


Figure 4. Availability Input Page

An example use of this application would be to calculate the percentage of time that the radio link with FEC had FERs less than 10^{-6} between October 1st and November 30th. This application can display a simple aggregate percentage over the entire time range specified, or it can graph the availability for each day in the time range. Figure 5 below is an example plot of this graphical feature. It uses the example input parameters specified above.

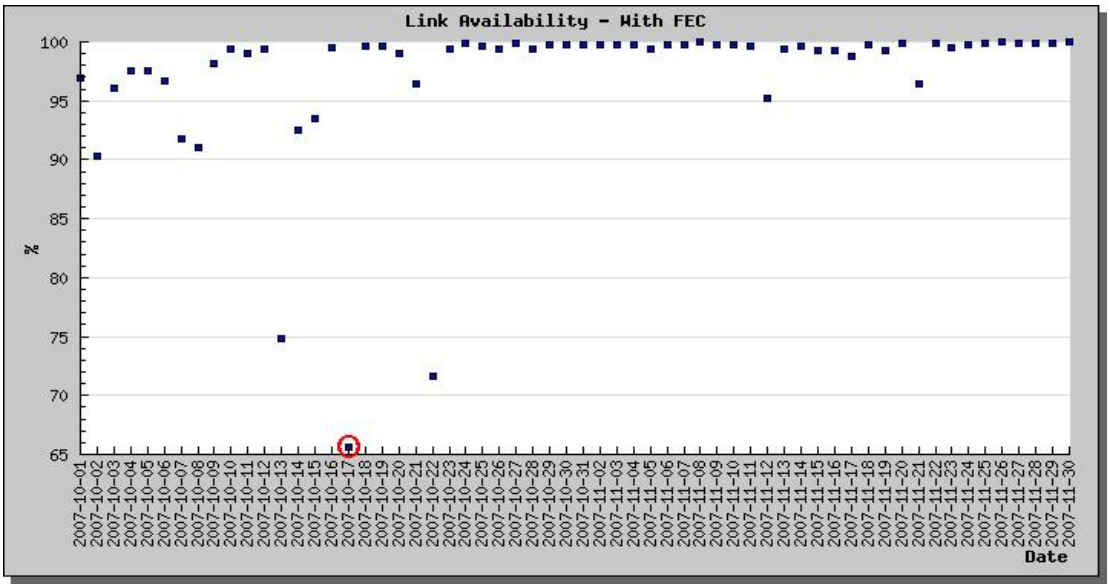


Figure 5. Link Availability for the Radio with FEC

Figure 5 shows the availability of the link with FEC over the course of 61 days. Several of the days had low availability levels (corresponding to higher FERs) and this was likely the result of precipitation moving through the link area. Table A1 in the Appendix has a complete list detailing most of the recorded weather events over a 7 month time period and the corresponding availabilities of the two radio link types.

Accompanying this availability calculator is a rain rate calculator. Similar to the availability calculator, it calculates the percentage of time that the rain rate was *greater* than a specified threshold. The application can also output an aggregate percentage or a day-by-day plot. Figure 6 below is a plot of the same time range as the availability plot above (Figure 5), using a threshold of 0.01 inches/hr (0.254 mm/hr) at the Nichols location.

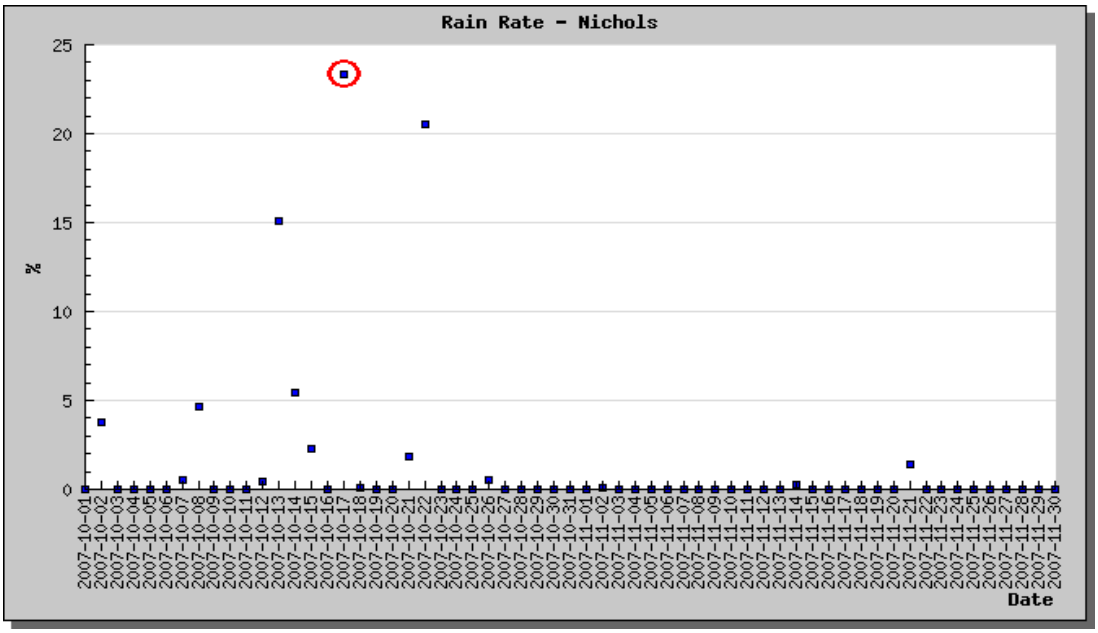


Figure 6. Percentage of Rain Rates above 0.01 in/hr at the Nichols Location

Figure 6 shows that most days had 100% of the points with a rain rate of less than 0.01 in/hr. Several of the days do show significant rain events, and it's very interesting to see that these days with high rain rates have correspondingly low link availabilities. Take October 17th (circled data point) as an example. This relationship is to be expected, precipitation degrades the strength of the wireless link.

The last application to be created uses the matched RR/FER pairs as described in the next section. This application graphically displays the relationship between RR and FER for any single day specified. Figure 7 shows the input webpage for this application.

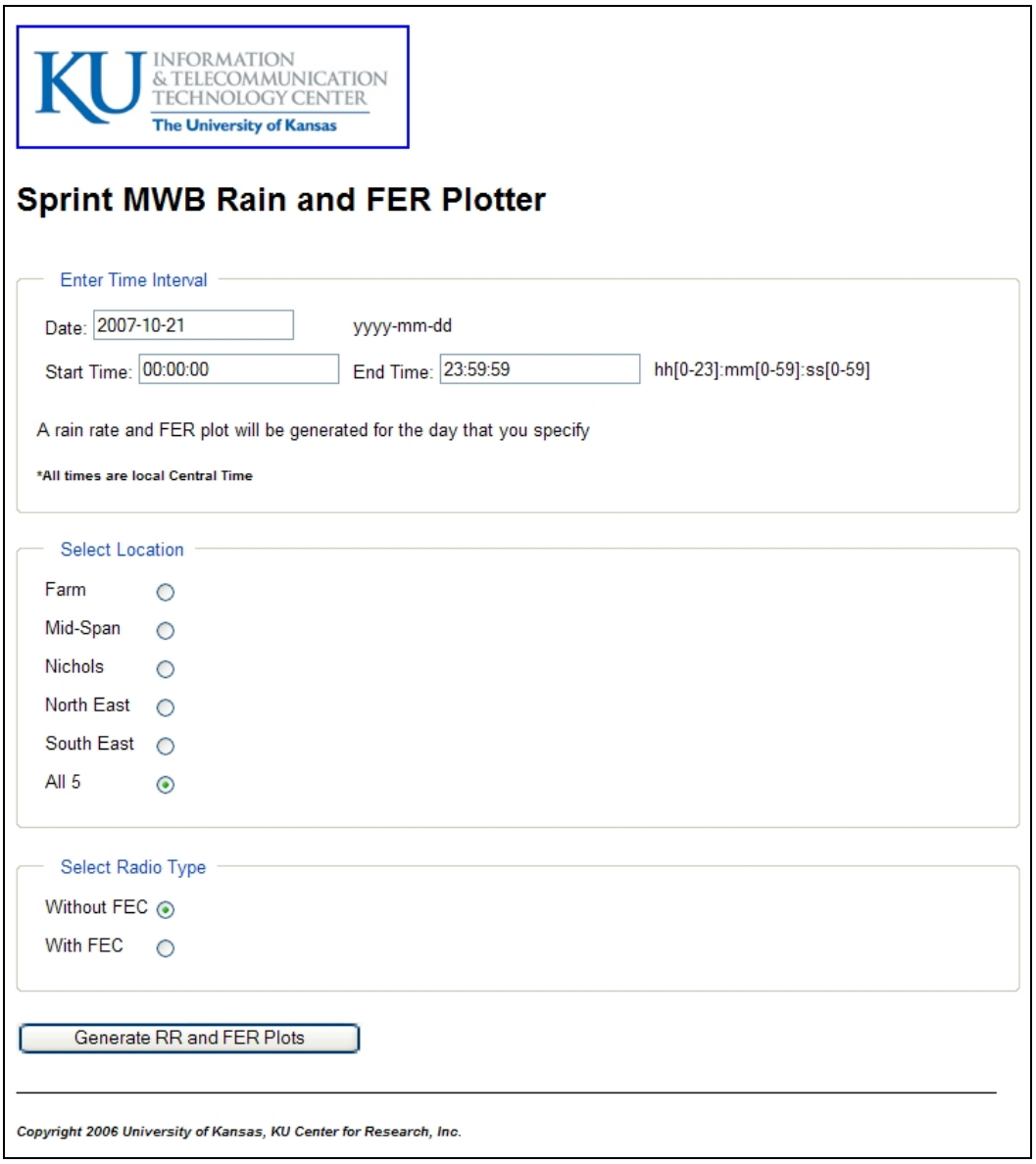


Figure 7. Input Webpage for RR/FER Plotter

The application simply plots the matched up FER and RR on two side-by-side graphs for the specified day. Figures 8 and 9 are an example of a one hour plot from October 17th. Figure 8 shows the FERs and Figure 9 shows the RRs.

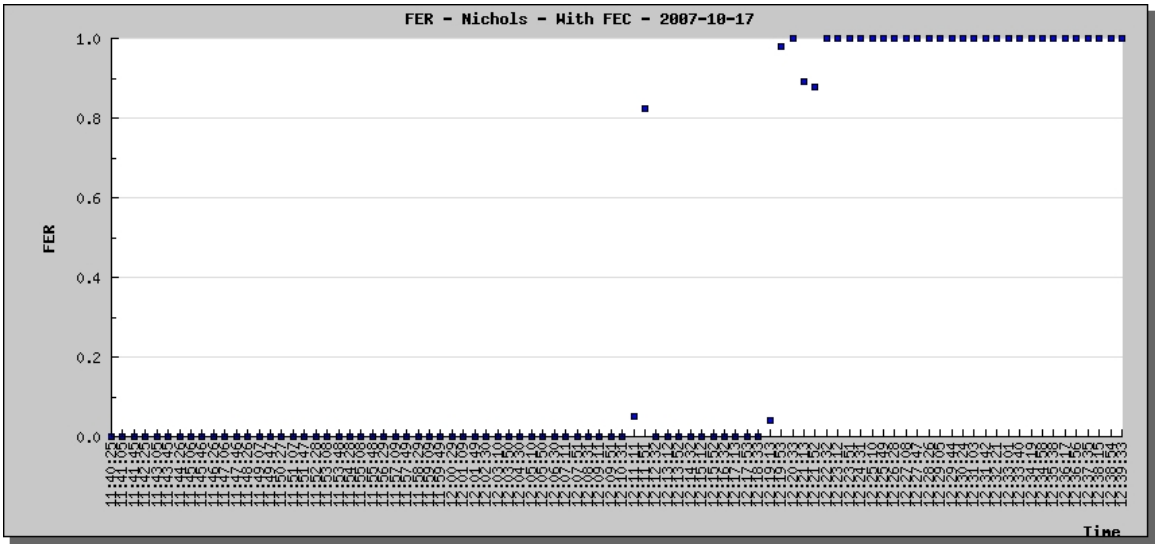


Figure 8. Frame Error Rate for a Single Hour in October, 2007

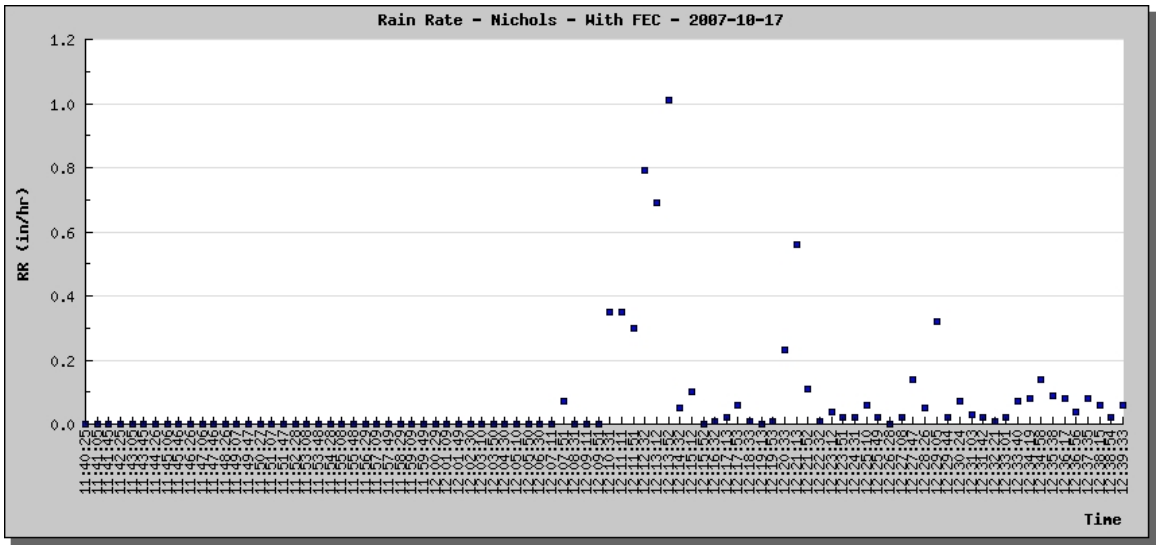


Figure 9. Rain Rate for a Single Hour in October, 2007

As can be seen by looking at these two figures, during this one hour time frame, a significant storm rolls in and the RR increases, causing the FER to dramatically increase. The FER quickly shoots up to 100% with few points in between 0 and 100 percent FER.

In order to ensure that any invalid data was disregarded for analysis, a downtime log was created. This log stores all the dates that experienced some sort of system issue and resulted in invalid link performance data. All three of the applications mentioned above use this downtime log to inform the user of any dates in his/her query that may have invalid data. A simple web form was created to view the complete downtime log and to add new log entries. Figure 10 shows the main page for this feature.

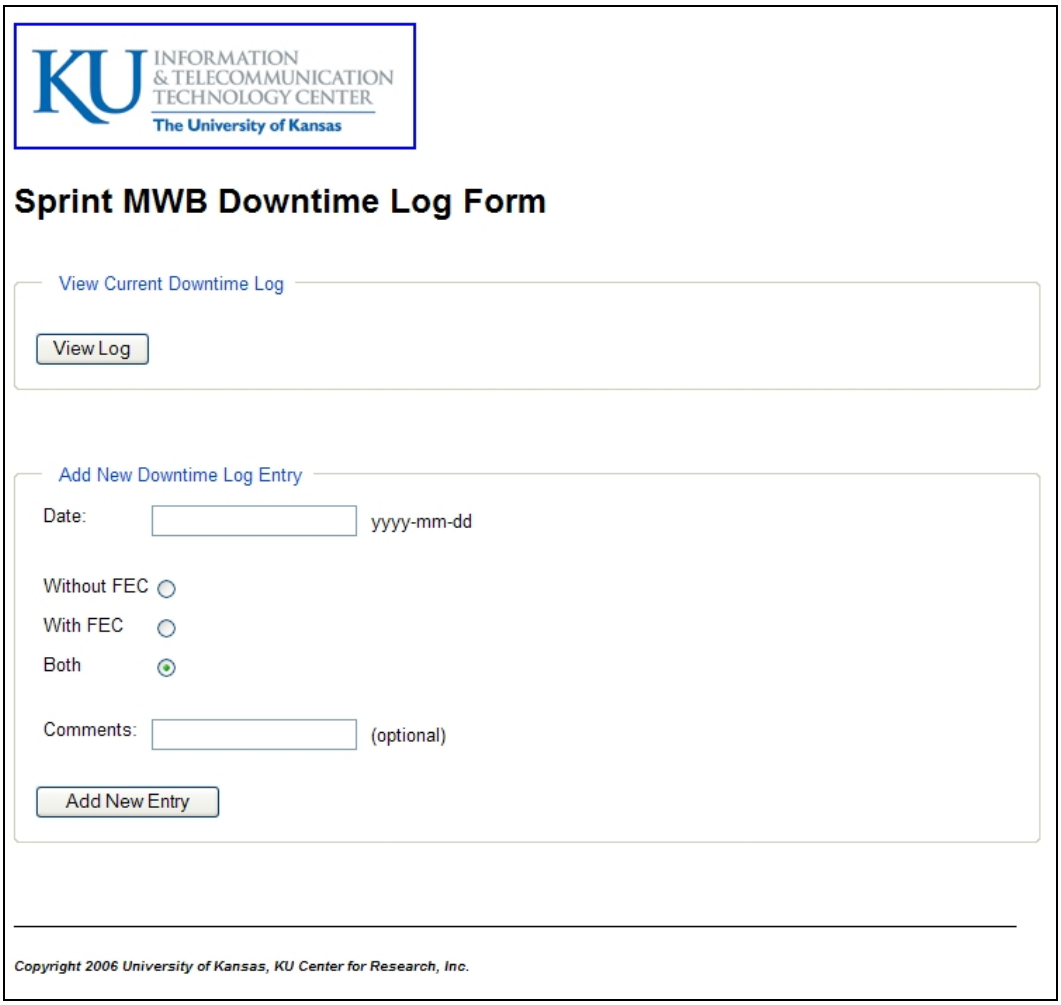


Figure 10. Downtime Log Webpage

All of these applications are easily accessible from the main website, and because of the use of cron jobs and the database, the applications have access to current data that is at most 24 hours old. The database is indexed for optimization, and considering the amount of data being processed, the applications take a minimum of time to return results.

4.3 RR/FER Analysis

In addition to the applications mentioned in the previous section, an important method for analyzing the link between FER and RR is to match the two quantities together and generate a plot of the resulting relationship. Before comparison of these two quantities was possible, the individual RR and FER data points, or samples, had to be matched up by time. Because of the difference in collection intervals, there were 25% less FER points collected than RR points. To make up for this, 25% of the RR points were simply ignored for the RR/FER analysis. The matching algorithm matched points with similar timestamps, as the following diagram (Figure 11) shows.

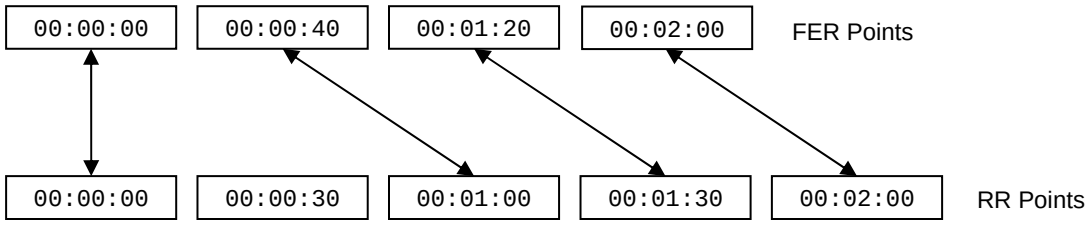


Figure 11. Matching of FER and RR points

The algorithm finds a RR point with a time value greater than or equal to the time value of the FER point to be matched, but also less than the next FER point's time value. This naturally excludes 25% of the RR points. As the figure shows, the two matched points will often not have the exact same time value; however this discrepancy is at most 40 seconds. This small difference should not result in any substantial increases in the error of these calculations.

These matched RR/FER points are then stored into a special table (RR_VS_FER) in the database for later analysis. Using another Perl script, these matched points are grouped together into bins for plotting as a histogram. Each bin is a collection of matched FER/RR points that correspond to RRs that fall into a specific range, and the average FER is calculated for all these points in the bin. A good deal of variance is found in the correlation between FER and RR, and that is why histograms were used to plot the relationship. The number of points in each bin decreases exponentially as the RR increases, which is to be expected because the higher the RR, the less likely it is to occur in nature.

Figures 12 through 17 show some examples of these histogram plots.

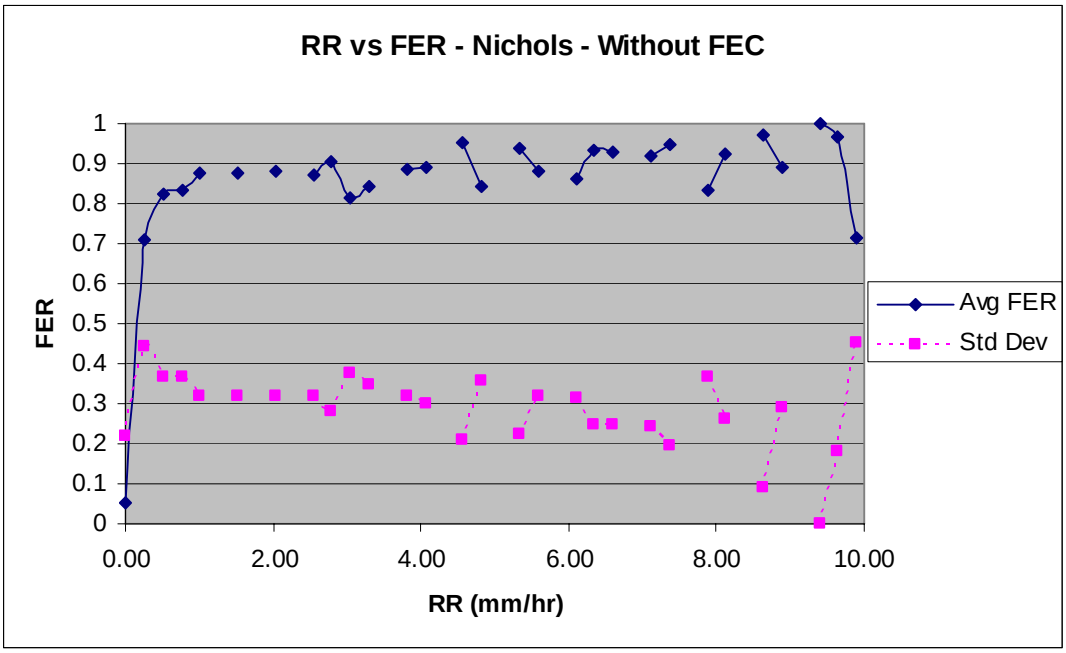


Figure 12

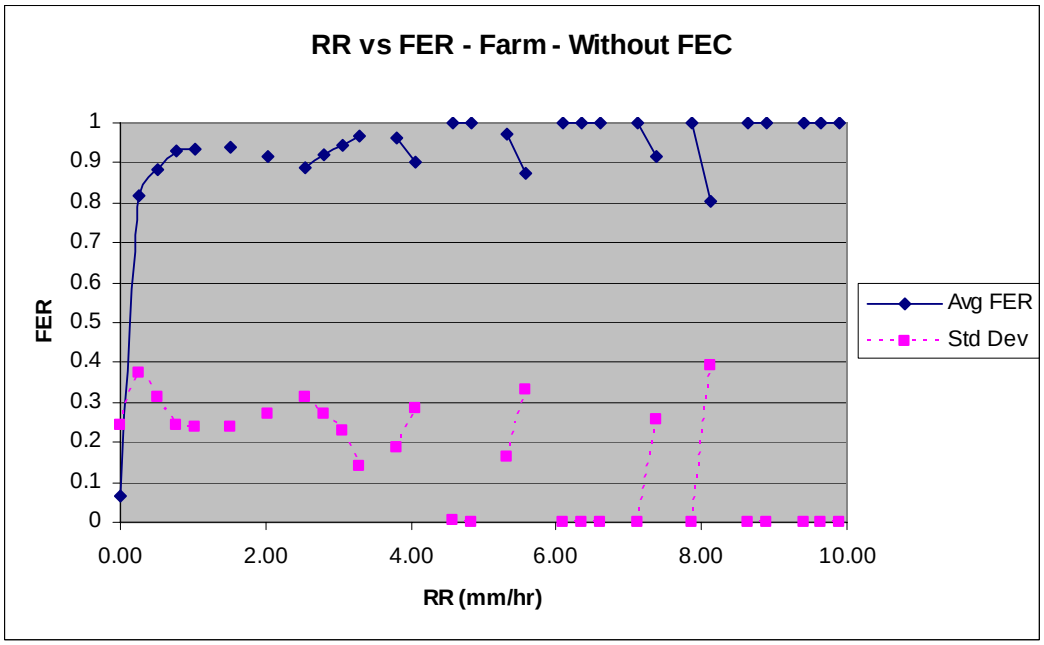


Figure 13

Figures 12 and 13 show the performance of the radio link without FEC at the Nichols and Farm locations, respectively. Each point corresponds to the average FER for all points in that particular RR bin. For example, the first point corresponds to all points with RRs of 0.00 mm/hr. The next point corresponds to all points with RRs larger than 0.00 mm/hr but less than 0.25 mm/hr, and so on. This plot shows a dramatic increase in FER as the RR increases to anything larger than 0.00 mm/hr.

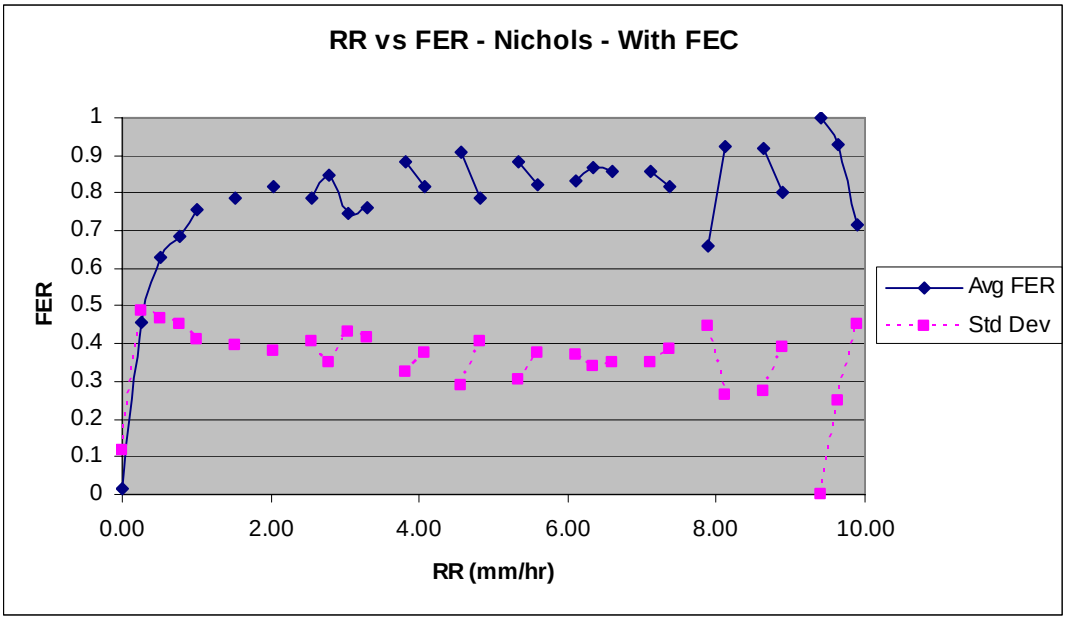


Figure 14

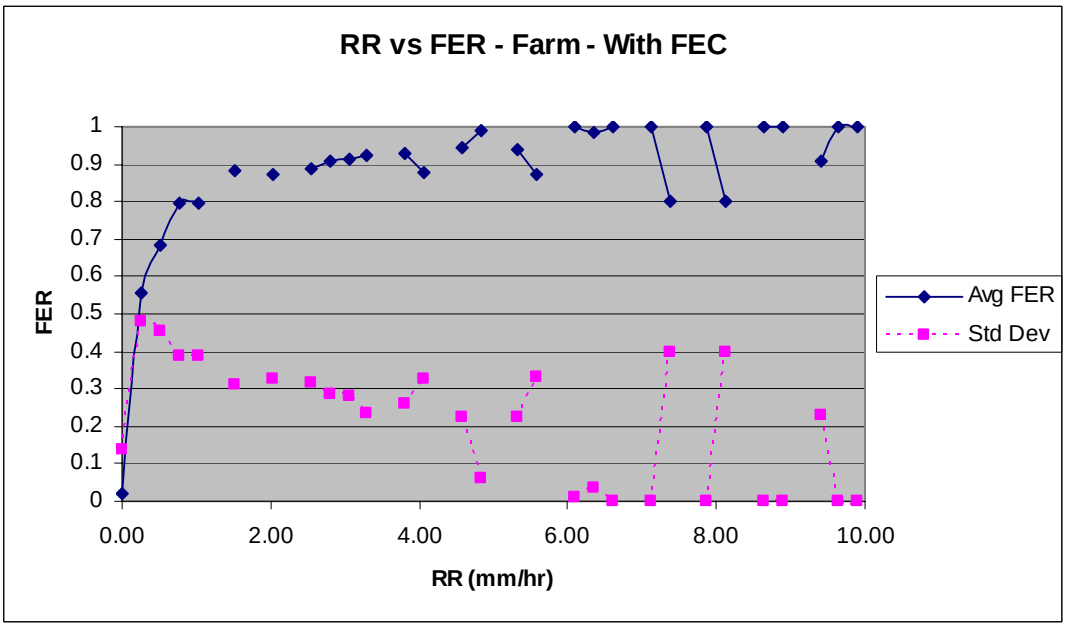


Figure 15

Figures 14 and 15 show the same locations as the two plots before, except this data comes from the radio that uses FEC. As can be seen in these two figures, this enhancement makes a visible improvement in the FER as a function of the RR. The radio without FEC had higher FER rates than the radio with FEC for the first couple RR bins.

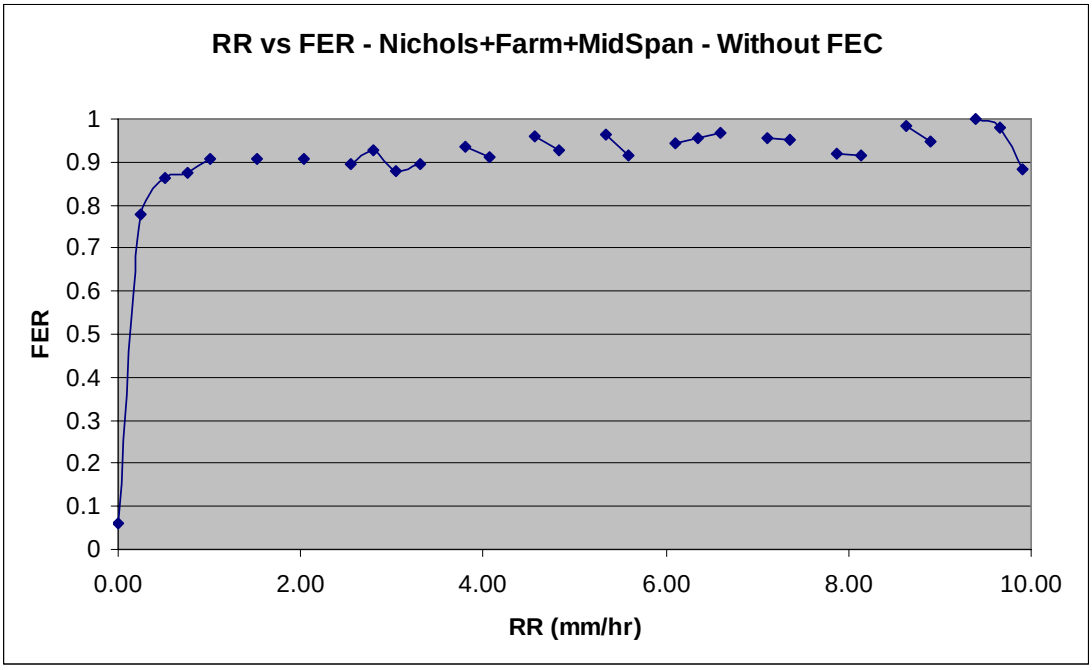


Figure 16

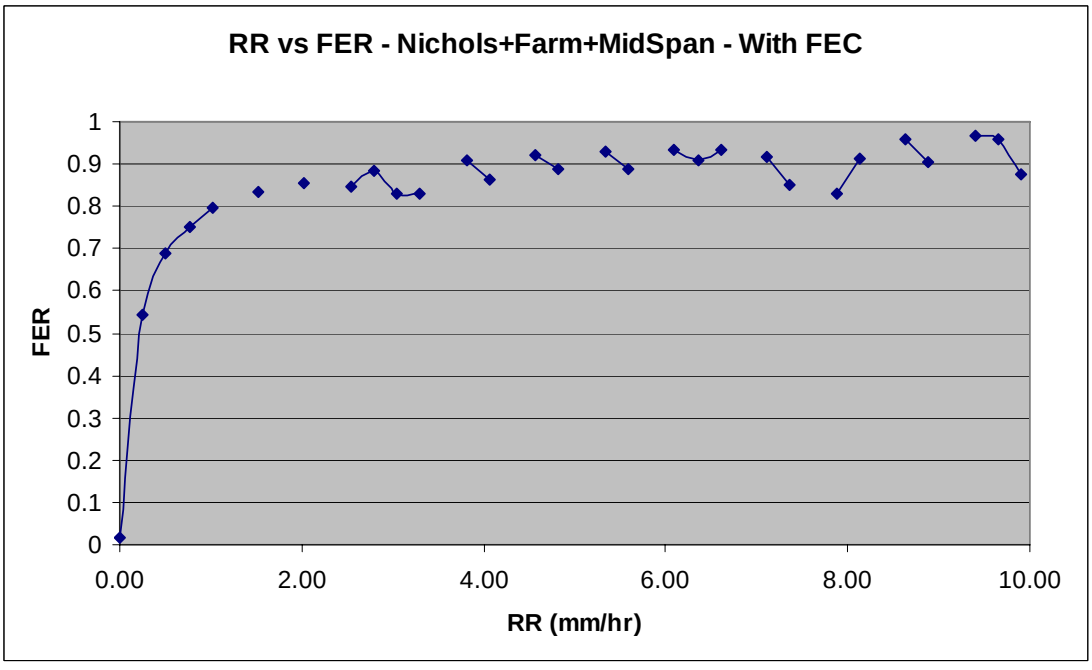


Figure 17

Figures 16 and 17 show the averages for the two radios over all three locations. It is clearly evident here that the radio with FEC outperformed the other radio. Significantly high FERs result from even the slightest rain in both cases, however.

An important note to make is that different locations do result in different FER/RR plots. This can easily be seen when comparing Figure 12 with Figure 13 or when

comparing Figure 14 with Figure 15. This difference is to be expected because weather measurements taken at different locations from the radio link site will introduce large time differences into the results.

The standard deviations (indicated by the dashed lines above) are not abnormally high and decrease to zero for some of the high RR points as there are few samples included in these bins.

5 Conclusions and Future Work

The RR/FER analysis results shown in the previous sections were somewhat unexpected. Almost any degree of precipitation showed high FERs, but it would be expected (from Figure 2) that small levels of precipitation would still leave these links operational [4]. Figure 2 shows that theoretically, it would be expected that the link would not become unavailable (with a FER greater than 10^{-6}) unless the rain rate was at least 1 mm/hr. The results gathered from this research indicate that the link becomes unavailable with a smaller amount of precipitation. Figures 17 and 18 show that the link condition deteriorates to a level of almost complete inoperability when the rain rate reaches about 1 mm/hr. The empirical results seem to indicate that for this particular link setup and geographic location, almost any amount of precipitation renders these high frequency links essentially unavailable.

The two radio types had significantly different levels of performance. For almost every amount of precipitation, the radio link with FEC had lower FERs than the radio link without FEC, and this would be expected. The collection location of the weather data used to compare with the link data also had an impact on the RR/FER plots.

There are several possible explanations for why the RR/FER plots showed higher FERs than the theoretical results. First off, the weather data is measured from single locations, and this is not completely representative of the weather all along the length of the radio link. Secondly, the differences with time in the matched up FER and RR points may have resulted in higher FERs for lower RRs than was actually present. There were timing offsets between the weather collection instruments and the FER measurement equipment. Another possibility is that the assumptions and numbers used in the link budget analysis are incorrect. Perhaps the theoretical model that we are comparing our data to is not representative of our link arrangement. And finally, there is a possibility that the analysis method used (where an average FER is calculated for a small range of RRs) is the source of this discrepancy. It would have been interesting to complete the analysis using the opposite procedure: averaging the RRs for a small range of FERs.

There were many lessons learned during the work on this research project. First and foremost, it would have been wise to have implemented the database from the very start. If the data collection, organization, and storage had been fully automated from the very beginning, many hours of unnecessary work could have been avoided. Secondly, it would have been more convenient and precise if the weather data had been collected on the same timing interval and in time synchronization as the link performance data. If all the equipment was time synchronized and therefore these two types of data samples had been collected at the exact same time, comparison and analysis of the RR and FER data would have been much easier and straightforward. And perhaps a final suggestion would have been to have collected data for a longer period of time. If, say, an entire year's worth of weather and link performance data had been collected, perhaps the results would have been slightly different. It would have been interesting to see if the changing of the four seasons had any impact on the overall performance of the links.

6 References

[1] T. Utsunomiya and M. Sekine, “Rain Attenuation at Millimeter and Submillimeter Wavelengths”, International Journal of Infrared and Millimeter Waves, vol. 26, no. 6, pp. 905-920, 2005.

[2] “Vaisala Weather Transmitter WXT510: The Most Essential of Weather” Vaisala Instruments Catalog, 2008, pp 130-134.

[3] R. K. Crane, “Prediction of Attenuation by Rain” IEEE Transactions of Communications, vol 28, issue 9, pp 1717-1733, 1980.

[4] Bharatwajan Raman, “A Synthetic Storm Modeling Technique for Millimeter Wave Band Mesh Networks”, Master of Science Thesis, University of Kansas, 2008.

[5] “MySQL 6.0 Reference Manual”, 2008 from: http://dev.mysql.com/doc/refman/6.0/en/index.html.

7 Appendix

Table A1. Link Availability for Varying Weather Events

Weather Event/Comment	Start date	Start time	End date	End time	Without FEC (%)	With FEC (%)	Availability Threshold
All observations	6/30/2007	0:00:00	2/4/2008	23:59:59	38.64	94.69	0.000001
Humidity effect	6/30/2007	0:00:00	2/4/2008	9:00:00	23.79	94.38	0.000001
Humidity effect	6/30/2007	9:00:00	2/4/2008	22:00:00	48.48	94.88	0.000001
Rain Event	7/23/2007	10:30:00	7/23/2007	11:30:00	0	17.6	0.000001
Rain Event	7/23/2007	10:30:00	7/23/2007	11:30:00	0	18.7	0.0001
Rain Event	7/23/2007	10:30:00	7/23/2007	11:30:00	11	19.8	0.01
Rain Event	7/27/2007	20:00:00	7/27/2007	21:45:00	8.9	68.1	0.000001
Rain Event	7/27/2007	20:00:00	7/27/2007	21:45:00	51.6	73.9	0.0001
Rain Event	7/27/2007	20:00:00	7/27/2007	21:45:00	56	75.1	0.01
Rain Event	7/28/2007	3:30:00	7/28/2007	4:00:00	27.6	93.67	0.000001
Rain Event	7/29/2007	11:40:00	7/29/2007	12:05:00	0	10.2	0.000001
Rain Event	8/2/2007	9:55:00	8/2/2007	10:05:00	0	0	0.000001
Rain Event	8/2/2007	9:55:00	8/2/2007	10:05:00	0	13.3	0.01
Rain Event	8/9/2007	0:00:00	8/9/2007	1:00:00	1.1	15.3	0.000001
Rain Event	8/9/2007	0:00:00	8/9/2007	1:00:00	8.8	16.5	0.01
Rain Event	8/16/2007	18:00:00	8/16/2007	23:00:00	8.5	56	0.000001
Rain Event	8/16/2007	18:00:00	8/16/2007	23:00:00	53.1	62.4	0.01
Rain Event	8/19/2007	6:00:00	8/19/2007	7:00:00	1.1	85.7	0.000001
Rain Event	8/19/2007	6:00:00	8/19/2007	7:00:00	72.5	87.9	0.01
Rain Event	8/24/2007	4:00:00	8/24/2007	5:00:00	0	56	0.000001
Rain Event	8/24/2007	4:00:00	8/24/2007	5:00:00	43.9	59.3	0.01
Rain Event	9/5/2007	8:30:00	9/5/2007	12:30:00	0	99.7	0.000001
Rain Event	9/7/2007	1:00:00	9/7/2007	3:00:00	0	1	0.000001
Rain Event	9/7/2007	1:00:00	9/7/2007	3:00:00	2.1	3.8	0.01
Rain Event	9/10/2007	16:30:00	9/10/2007	16:30:00	0	84.4	0.000001
Rain Event	9/10/2007	16:30:00	9/10/2007	16:30:00	77.8	88.9	0.01
Rain Event	9/16/2007	12:40:00	9/16/2007	13:00:00	0	30	0.000001
Rain Event	9/16/2007	12:40:00	9/16/2007	13:00:00	16.7	33.3	0.01
Rain Event	9/18/2007	14:20:00	9/18/2007	16:30:00	0	3.5	0.000001
Rain Event	9/18/2007	14:20:00	9/18/2007	16:30:00	4.5	8.5	0.01
Rain Event-Heavy	9/19/2007	19:30:00	9/19/2007	20:00:00	0	0	0.000001
Rain Event-Heavy	9/19/2007	19:30:00	9/19/2007	20:00:00	0	0	0.01
Power disruption	9/20/2007	12:45:00	9/20/2007	15:00:00	0	0	0.000001
Rain Event	9/21/2007	20:30:00	9/21/2007	21:00:00	8.6	34.7	0.000001
Rain Event	9/21/2007	20:30:00	9/21/2007	21:00:00	34.7	43.5	0.01
Rain Event	9/25/2007	8:00:00	9/25/2007	12:00:00	0	92.8	0.000001
Rain Event	9/25/2007	8:00:00	9/25/2007	12:00:00	87.3	97.2	0.01
Rain Event	9/30/2007	15:00:00	9/30/2007	16:00:00	2.2	36.3	0.000001
Rain Event	9/30/2007	15:00:00	9/30/2007	16:00:00	36.2	38.5	0.01
Rain Event	10/2/2007	12:30:00	10/2/2007	13:45:00	0	0	0.000001
Rain Event	10/2/2007	12:30:00	10/2/2007	13:45:00	0	2.6	0.01
Rain Event	10/7/2007	14:00:00	10/7/2007	20:00:00	0.2	77.1	0.000001
Rain Event	10/7/2007	14:00:00	10/7/2007	20:00:00	64.5	81.7	0.01
Rain Event	10/13/2007	1:00:00	10/13/2007	12:00:00	0	48.1	0.000001

Weather Event/Comment	Start date	Start time	End date	End time	Without FEC (%)	With FEC (%)	Availability Threshold
Rain Event	10/17/2007	5:00:00	10/17/2007	5:00:00	0.13	48.9	0.000001
Rain Event	10/22/2007	0:00:00	10/22/2007	8:30:00	0.13	22.45	0.000001
Rain Event	11/21/2007	0:00:00	11/21/2007	23:59:59	0	96.4	0.000001
Low Cloud deck	11/23/2007	5:00:00	11/23/2007	9:00:00	0	100	0.000001
Snow event	12/6/2007	9:00:00	12/6/2007	14:30:00	Radio not operational	99.6	0.000001
Mix Precipitation	12/22/2007	10:30:00	12/22/2007	23:59:59	33.4	94.9	0.000001
Fog	12/27/2007	1:00:00	12/27/2007	8:00:00	0	93.5	0.000001
Snow event	12/28/2007	1:00:00	12/28/2007	8:00:00	0	99.0431	0.000001
Snow event	12/28/2007	1:00:00	12/28/2007	8:00:00	0	99.8405	0.0001
Snow event	12/28/2007	1:00:00	12/28/2007	8:00:00	19.3	100	0.001
Snow event	12/28/2007	1:00:00	12/28/2007	8:00:00	100	100	0.01
Haze/reduced visibility	12/29/2007	10:00:00	12/29/2007	12:00:00	39.7	93.3	0.000001
Fog/reduced visibility	12/29/2007	22:00:00	12/29/2007	23:59:59	2.2	100	0.000001
Humidity event >90%	1/4/2008	8:00:00	1/4/2008	17:00:00	97.4	100	0.000001
Rain/snow event	1/8/2008	11:40:00	1/8/2008	13:30:00	0	74.5	0.000001
Rain/snow event	1/10/2008	0:07:00	1/10/2008	0:07:00	28.7	95.1	0.000001
Snow Event	1/16/2008	12:00:00	1/16/2008	18:00:00	0	96.3	0.000001
Snow Event	1/17/2008	1:00:00	1/17/2008	4:00:00	3.7	98.9	0.000001
Snow Event	1/17/2008	1:00:00	1/17/2008	4:00:00	84.4	99.6	0.0001
Snow Event	1/17/2008	1:00:00	1/17/2008	4:00:00	100	100	0.01
Snow Event	1/18/2008	10:30:00	1/18/2008	11:30:00	0	100	0.000001
Snow Event	1/18/2008	10:30:00	1/18/2008	11:30:00	16.7	100	0.00001
Snow Event	1/18/2008	10:30:00	1/18/2008	11:30:00	100	100	0.0001
Snow Event	1/25/2008	6:00:00	1/25/2008	8:00:00	10.6	100	0.000001
Snow Event	1/25/2008	6:00:00	1/25/2008	8:00:00	98.9	100	0.0001
Snow Event	1/29/2008	10:00:00	1/29/2008	14:00:00	0	100	0.000001
Snow Event	1/29/2008	10:00:00	1/29/2008	14:00:00	89.7	100	0.001
Rain Event	2/3/2008	9:15:00	2/3/2008	14:30:00	8	18.6	0.000001
Rain Event	2/3/2008	9:15:00	2/3/2008	14:30:00	39.5	37.4	0.0001
Rain Event	2/3/2008	9:15:00	2/3/2008	14:30:00	69.3	64.9	0.01

7.1 Source Code

parseRain.pl – Adds the raw weather data to the database

```
#Filename: parseRain.pl
#Description: Parses the raw data files located in the base_dir and adds the data
   to the MySQL database

#!/usr/bin/perl -w
use warnings;
use strict;
use DBI;

sub add_wxt510_data($$);
sub add_rr_data($$);
sub check_before($$$$);


#------ Initialization and DB Setup ------#

#flag used to update last_update field in DB after each data set (0 or 1)
my $update_flag = 0;  #don't update after each data set

#location of raw weather data files
my $base_dir = "/projects/sprintmwb/Weather_Station_Data/";

#create the DB Connector
my $username = 'sprintmwb';
my $password = 'sprintmwb';
my $database = 'sprintmwb';
my $hostname = 'fiasco.ittc.ku.edu';
#db handle:
my $dbh = DBI->connect("dbi:mysql:database=$database;host=$hostname;", $username,
   $password) or die "Couldn't connect to database: " . DBI->errstr;

#----------------------------------------#


#------ Collect Array Of Files To Parse ------#

my @files_to_parse;

#Create list of directories (locations)
my @locations = (
   #{dir => 'Farm_S-DL-PF', name => 'Farm',},
   {dir => 'Mid-Span_S-MIDDL-SB', name => 'Mid_Span',},
   {dir => 'Nichols_S-DL-NH', name => 'Nichols',},
   {dir => 'NorthEast_S-NEDL-SB', name => 'North_East',},
   {dir => 'SouthEast_S-SEDL-SB', name => 'South_East',} );

print "Files that will be parsed and added to the DB:\n";

foreach my $location_ref (@locations) #loop through the location directories
{
   my $dir = $location_ref->{dir};
   my $location = $location_ref->{name};

   #print "$dir\n";  #print all locations
```

```
   #my $current_date = `date +%Y_%b_%d`;

   #check last date in DB for each location
   my $sql_query = "SELECT MAX(date) FROM sprintmwb.wxt510_raw WHERE location =
   '$location'"; # the SQL query
   my $select = $dbh->prepare($sql_query) or die "Couldn't prepare statement: " .
   $dbh->errstr;
   $select->execute() or die "Couldn't execute statement: " . $select->errstr;
   my @sql_return = $select->fetchrow_array;

   $select->finish;

   my $last_update = $sql_return[0];  #last_update field now has timestamp from DB

   #print "Last Date: $last_update\n";  #print the last_update dates from the DB

   #Create list of all files for the location
   opendir(DIR_HANDLE, "$base_dir$dir/") || die "Could not open location
   directory";
   my @filenames = grep(/^\d\d\d\d_\w\w\w_\d\d_\S+_WXT510.dat$/,
   readdir(DIR_HANDLE));  #only include valid WXT510.dat files
   @filenames = sort(@filenames);  #sort files in chronological order
   closedir(DIR_HANDLE);

   foreach my $filename (@filenames) #loop through the days in the location
   directory
   {
      #print "  $filename\n";  #print all files

      #use function to determine if the file needs to be added to the DB
      my @temp = split(/_/,$filename);  #split line into individual values
      my $year = $temp[0];
      my $month = $temp[1];
      my $day = $temp[2];

      if(check_before($last_update, $year, $month, $day))
      {
         push(@files_to_parse, "$base_dir$dir/$filename");  #add this file to the
   DB

         #print "$base_dir$dir/$filename\n";  #print all the filenames with
   directory paths
         print "$dir/$filename\n";  #print all the filenames with sub directory
      }
   }
}


my $num_of_files_to_parse = @files_to_parse;  #find how many files we have to
   parse
my $num_of_files_parsed = 0;  #counter that keeps track of how many files have
   been parsed
my $percent_mult = 1;
my $percent_complete = 0;

#-------------------------------------------#


#Tests a single file
#my @TEST_FILE_TO_PARSE = ("$files_to_parse[20]");
```

```
#print "Loading: $files_to_parse[20]\n";


#------ Parse Data Files and Add Data to DB ------#

print "Beginning Data Insertion...\n";

#Upload files with timestamp after last_update field
foreach my $filename (@files_to_parse)
{
   open(FILE_HANDLE, "<", "$filename") || die "Could not open file";

   my @data =  <FILE_HANDLE>;  #array now contains all data from the file

   my ($date, $time, $location);
   my $count = 1;  #line counter

   close(FILE_HANDLE);

   #parse the data file
   foreach my $line (@data)
   {

      if ($count == 1)  #first line, gather location
      {
         my @temp = split(/,/,$line);  #split line into individual values

         my $loc = $temp[1];
         $loc =~ s/"//g;  #remove the quotes around the location

         #print "Location: $loc\n";  #print the location

         #if($loc eq 'S-DL-PF') #Farm
         #{
         #   $location = 'Farm';
         #}
         if($loc eq 'S-MIDDL-SB') #Mid_Span
         {
            $location = 'Mid_Span';
         }
         elsif($loc eq 'S-DL-NH') #Nichols
         {
            $location = 'Nichols';
         }
         elsif($loc eq 'S-NEDL-SB') #North_East
         {
            $location = 'North_East';
         }
         elsif($loc eq 'S-SEDL-SB') #South_East
         {
            $location = 'South_East';
         }
         else
         {
            print "Error: Bad Location Type\n";
            print "  Filename: $filename\n";
         }
      }
```

```
   elsif (($count > 4) && ($line =~ m/\d\d\d\d-\d\d-\d\d \d\d:\d\d:\d\d/))
#check if the line is a valid data line
   ##elsif (($count > 4) && ($line =~ /\S/))  #check if the line is a non-blank
data line
   {
      my @temp = split(/,/,$line);  #split line into individual values

      $temp[0] =~ s/"//g;  #remove the quotes around the timestamp

      my @timestamp = split(/ /,$temp[0]);  #grab date and time from timestamp
      my $date = $timestamp[0];
      my $time = $timestamp[1];

      #print "Date: $date\n";
      #print "Time: $time\n";

      if ((!$date) || (!$time))  #insure that we have a valid timestamp
            {
                  print STDERR "ERROR WITH TIMESTAMP!  Date: $date  Time:
$time  Location: $location\n";
        print STDERR "  Filename: $filename\n";
            }


      #$last_update = "$date $time";

            #if($update_flag == 1)
            #{
                  #update last_update timestamp in the DB
      #   update_timestamp($dbh, $last_update);
            #}


      #Raw data grabbed from .dat file
      my %data_row;

      $data_row{"date"} = $date;
      $data_row{"time"} = $time;
      $data_row{"location"} = $location;
      $data_row{"wind_dir_min"} = $temp[4];
      $data_row{"wind_dir_avg"} = $temp[5];
      $data_row{"wind_dir_max"} = $temp[6];
      $data_row{"wind_speed_min"} = $temp[7];
      $data_row{"wind_speed_avg"} = $temp[8];
      $data_row{"wind_speed_max"} = $temp[9];
      $data_row{"air_temp"} = $temp[10];
      $data_row{"rel_humidity"} = $temp[11];
      $data_row{"air_pressure"} = $temp[12];
      $data_row{"rain_amount"} = $temp[13];
      $data_row{"rain_duration"} = $temp[14];  # these were
      $data_row{"rain_intensity"} = $temp[15];  # switched
      $data_row{"hail_amount"} = $temp[16];
      $data_row{"hail_duration"} = $temp[17];  # these were
      $data_row{"hail_intensity"} = $temp[18];  # switched
      $data_row{"filename"} = $filename;  #used for debugging

      my %rr_data = (date => $date, time => $time, location =>
$data_row{"location"}, rr => $data_row{"rain_intensity"}, filename =>
$filename);  #create hash of rr data
```

```
            #Add the wxt510 data to database!
            add_wxt510_data($dbh, \%data_row);

            #Add the rr data to the database!
            add_rr_data($dbh, \%rr_data);

         }
         #else{}  #otherwise the data is garbage

         $count = $count + 1;

      }  #end of looping through lines


      # Display percentage complete text
      #$num_of_files_parsed += 1; #increment the parsed files counter
      #$percent_complete = int(100 * ($num_of_files_parsed /
      $num_of_files_to_parse));

      #if($percent_complete >= ($percent_mult*10))
      #{
      #   print "\n$percent_complete% complete\n";  #display percentage complete
      #   $percent_mult += 1;
      #}


}  #end of looping through files


#close connection with database
$dbh->disconnect;

print "\nData Insertion Complete!\n";

#-------------------------------------------------#


#----- Subroutine Definitions -----#

#function to add wxt510 data to database
#expects 2 arguments, the data base handle, and a reference to the data hash to be
   added to the DB
sub add_wxt510_data($$)
{
   my $dbh = shift;
   my $data_ref = shift;
   my %data = %{$data_ref};

   # the SQL insertion statement
   my $sql_query1 = 'INSERT INTO sprintmwb.wxt510_raw
   VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)';

   my $select = $dbh->prepare($sql_query1) or die "Couldn't prepare statement: "
   . $dbh->errstr;
   $select->execute($data{'date'}, $data{'time'}, $data{'location'},
   $data{'wind_dir_min'},
      $data{'wind_dir_avg'}, $data{'wind_dir_max'}, $data{'wind_speed_min'},
   $data{'wind_speed_avg'},
      $data{'wind_speed_max'}, $data{'air_temp'}, $data{'rel_humidity'},
   $data{'air_pressure'},
```

```
      $data{'rain_amount'}, $data{'rain_duration'}, $data{'rain_intensity'},
   $data{'hail_amount'},
      $data{'hail_duration'}, $data{'hail_intensity'}); #or print "Couldn't
   execute statement: 'INSERT INTO sprintmwb.wxt510_raw'\n  Filename:
   $data{'filename'}\n";

   $select->finish;
}


#function to add rr data to database
#expects 2 arguments, the data base handle, and a reference to the data hash to be
   added to the DB
sub add_rr_data($$)
{
   my $dbh = shift;
   my $data_ref = shift;
   my %data = %{$data_ref};

   # the SQL insertion statement
   my $sql_query1 = 'INSERT INTO sprintmwb.rr (date, time, location, rr_raw)
   VALUES(?,?,?,?)';

   my $select = $dbh->prepare($sql_query1) or die "Couldn't prepare statement: "
   . $dbh->errstr;
   $select->execute($data{'date'}, $data{'time'}, $data{'location'},
   $data{'rr'}); #or print "Couldn't execute statement: 'INSERT INTO sprintmwb.rr
   (date, time, location, rr_raw)'\n  Filename: $data{'filename'}\n";

   $select->finish;
}


#function to decide if the first date is before the second
#returns 1 if last_update is less recent than the given date, 0 otherwise
#expects four arguments:
# $last_update (ex: '2008-04-09')
# $year - (ex: '2007')
# $month - (ex: 'Jun')
# $day - (ex: '05')
sub check_before($$$$)
{
   my $last_update = shift;
   my @temp = split(/-/,$last_update);
   my $year_DB = $temp[0];
   my $month_DB = $temp[1];
   my $day_DB = $temp[2];

   my $year = shift;

   my $month_string = shift;
   my $month;

   if($month_string eq 'Jan') #Jan = '01'
   {
      $month = '01';
   }
   elsif($month_string eq 'Feb') #Feb = '02'
   {
      $month = '02';
```

```
}
elsif($month_string eq 'Mar') #Mar = '03'
{
   $month = '03';
}
elsif($month_string eq 'Apr') #Apr = '04'
{
   $month = '04';
}
elsif($month_string eq 'May') #May = '05'
{
   $month = '05';
}
elsif($month_string eq 'Jun') #Jun = '06'
{
   $month = '06';
}
elsif($month_string eq 'Jul') #Jul = '07'
{
   $month = '07';
}
elsif($month_string eq 'Aug') #Aug = '08'
{
   $month = '08';
}
elsif($month_string eq 'Sep') #Sep = '09'
{
   $month = '09';
}
elsif($month_string eq 'Oct') #Oct = '10'
{
   $month = '10';
}
elsif($month_string eq 'Nov') #Nov = '11'
{
   $month = '11';
}
elsif($month_string eq 'Dec') #Dec = '12'
{
   $month = '12';
}
else
{
   print "\nError: Bad Month Type!!!!!\n\n";
}

my $day = shift;

#Convert strings to numbers
$year_DB = $year_DB + 0;
$month_DB = $month_DB + 0;
$day_DB = $day_DB + 0;
$year = $year + 0;
$month = $month + 0;
$day = $day + 0;


my $result = 0;

if($year_DB < $year)
```

```
   {
      $result = 1;
   }
   elsif($year_DB == $year)
   {
      if($month_DB < $month)
      {
         $result = 1;
      }
      elsif($month_DB == $month)
      {
         if($day_DB < $day)
         {
            $result = 1;
         }
         elsif($day_DB == $day)
         {
            $result = 1;  #dates are same, CHECK TIME?
         }
      }
   }

   return $result;

}

#---------------------------------#
```

parseFER.pl – Adds the raw link performance data to the database

```
#Filename: parseFer.pl
#Description: Parses the raw data files located in the base_dir and adds the data
   to the MySQL database


#!/usr/bin/perl -w
use warnings;
use strict;
use DBI;

sub add_smartbits_data($$);
sub add_fer_data($$);
sub check_before($$$$);


#------ Initialization and DB Setup ------#

#location of raw FER data files
my $base_dir = "/projects/sprintmwb/smartbits_reports/daily_reports/";

#create the DB Connector
my $username = 'sprintmwb';
my $password = 'sprintmwb';
my $database = 'sprintmwb';
my $hostname = 'fiasco.ittc.ku.edu';
#db handle:
my $dbh = DBI->connect("dbi:mysql:database=$database;host=$hostname;", $username,
   $password) or die "Couldn't connect to database: " . DBI->errstr;

#my $current_date = `date +%Y_%b_%d`;

#check last date in DB for smartbits_raw table
my $sql_query = "SELECT MAX(date) FROM sprintmwb.smartbits_raw"; # the SQL query
my $select = $dbh->prepare($sql_query) or die "Couldn't prepare statement: " .
   $dbh->errstr;
$select->execute() or die "Couldn't execute statement: " . $select->errstr;
my @sql_return = $select->fetchrow_array;

$select->finish;

my $last_update = $sql_return[0];  #last_update field now has timestamp from DB

#print "Last Date: $last_update\n";  #print the last_update dates from the DB

#----------------------------------------#


#------ Collect Array Of Files To Parse ------#

my @files_to_parse;

#Create list of all files for the location
opendir(DIR_HANDLE, "$base_dir") || die "Could not open raw smartbits data
   directory";
my @sub_dirs = grep(/^\d\d\d\d_\d\d$/, readdir(DIR_HANDLE));  #get list of valid
   raw data directories
@sub_dirs = sort(@sub_dirs);  #sort directories in chronological order
```

```
closedir(DIR_HANDLE);

foreach my $sub_dir (@sub_dirs) #loop through the month directories
{
   #print "$sub_dir\n";  #print all sub directories

   #Create list of all files in the month
   opendir(DIR_HANDLE, "$base_dir$sub_dir/") || die "Could not open directory2";
   my @filenames = grep(/^\d\d.csv$/, readdir(DIR_HANDLE));  #grab only valid raw
   data .csv files
   @filenames = sort(@filenames);  #sort files in chronological order
   closedir(DIR_HANDLE);

   #grab the year and month from the subdirectory
   #my @temp = split(/_/,$sub_dir);  #split subdirectory into year and month
   #my $year = $temp[0];
   #my $month = $temp[1];

   foreach my $filename (@filenames) #loop through the days in the month directory
   {
      #print "  $filename\n";  #print all files
      #print "$sub_dir/$filename\n";  #print all the filenames with sub directory

      #use function to determine if the file needs to be added to the DB
      #my @temp = split(/\./,$filename);  #split filename into date and file
   extension
      #my $day = $temp[0];
      $last_update =~ m/(\d\d\d\d)-(\d\d)-(\d\d)/;

      #if(check_before($last_update, $year, $month, $day))
      #if(("$sub_dir" ge "$1_$2" or "$filename" ge "$3.cvs") or ("$sub_dir" gt
   "$1_$2" and "$3" gt '28'))
      if(("$sub_dir" ge "$1_$2" and "$filename" ge "$3.cvs") or ("$sub_dir" gt
   "$1_$2" and "$3" gt '28'))
      {
         push(@files_to_parse, "$base_dir$sub_dir/$filename");  #add this file to
   the DB

         #print "$base_dir$sub_dir/$filename\n";  #print all the filenames with
   directory paths
         #print "$sub_dir/$filename\n";  #print all the filenames with sub
   directory
      }
   }
}


my $num_of_files_to_parse = @files_to_parse;  #find how many files we have to
   parse
my $num_of_files_parsed = 0;  #counter that keeps track of how many files have
   been parsed
my $percent_mult = 1;
my $percent_complete = 0;

#-------------------------------------------#


#Tests a single file
#my @TEST_FILE_TO_PARSE = ("$files_to_parse[60]");
#print "Loading: $files_to_parse[60]\n";
```

```
#------ Parse Data Files and Add Data to DB ------#

#print "Beginning Data Insertion...\n";

#Upload files with dates after the last_update
foreach my $filename (@files_to_parse)
{
   #open the new data file
   open(FILE_HANDLE, "<", "$filename") || die "Could not open file";

   my @data =  <FILE_HANDLE>;  #array now contains all data from the file

   my ($date, $time);
   my $out_flag = 1;  #flag used to switch between outbound and inbound

   close(FILE_HANDLE);

   #parse the data file
   foreach my $line (@data)
   {
         if ($line =~ m/(\d\d:\d\d:\d\d) (\d\d\/\d\d\/\d\d\d\d)/)  #check if the
   line contains the date and time
         {
               $time = $1;
               $date = $2;

               #print "\nTIME: $time\n";  #FOR TESTING
               #print "DATE: $date\n\n";  #FOR TESTING

               #need to reverse the date components
         my @temp2 = split(/\//,$date);
         $date = "$temp2[2]/$temp2[0]/$temp2[1]";

               if ((!$temp2[2]) || (!$temp2[0]) || (!$temp2[1]))
               {
                     print STDERR "ERROR WITH DATE!  Date: $date  Time:
   $time\n";
               }

         }

      elsif ($line =~ m/^(Totals|Loea|BW|Bridgewave)/)  #check if the line is one
   of the data lines
      {
         my @temp = split(/,/,$line);  #split line into the individual data values

         #Raw data grabbed from .csv file
         my %data_row;

         $data_row{"date"} = $date;
         $data_row{"time"} = $time;
         #$data_row{"type"} = ?  This is filled in later
         $data_row{"packet_length"} = $temp[1];
         $data_row{"crc_size"} = $temp[2];
         $data_row{"flow_load"} = $temp[3];  #warning: the string "N/A" is present
   in types 1, 2, and 3
         $data_row{"tx_stream"} = $temp[4];
         $data_row{"transmitted"} = $temp[5];
```

```
      $data_row{"expected"} = $temp[6];
      $data_row{"received"} = $temp[7];
      $data_row{"lost"} = $temp[8];
      $data_row{"in_sequence"} = $temp[9];
      $data_row{"out_of_sequence"} = $temp[10];
      $data_row{"min_latency"} = $temp[11];
      $data_row{"max_latency"} = $temp[12];
      $data_row{"avg_latency"} = $temp[13];
      $data_row{"sf_min_latency"} = $temp[14];
      $data_row{"sf_max_latency"} = $temp[15];
      $data_row{"sf_avg_latency"} = $temp[16];
      $data_row{"std_deviation"} = $temp[17];

      #Latency Bucket data
      $data_row{"lb_num"} = "NULL";
      #$data_row{"lb_5_0"} = $temp[18];
      # .
      # .
      # .
      #$data_row{"lb_300_7g"} = $temp[33];


      # calculate FER
      my $fer;

      #prevent divide_by_zero error
      if ($data_row{"transmitted"} < 1)
      {
         $fer = 1.0;  #SHOULD THIS POINT BE DISCARDED?
      }
      else
      {
         $fer = ($data_row{"lost"}/$data_row{"transmitted"});
      }


      #------ Decide which data line it is ------#
      if ($line =~ m/^Totals/)  #Totals data (type 1)
      {
         $data_row{"type"} = 1;
      }

      elsif ($line =~ m/^Loea Group/)  #Loea Group data (type 2)
      {
         $data_row{"type"} = 2;
      }

      elsif (($line =~ m/^BW Group/) || ($line =~ m/^Bridgewave Group/))
#Bridgewave Group data (type 3)
      {
         $data_row{"type"} = 3;
      }

      elsif ($line =~ m/^Loea 2:1-1/)  #Loea Outbound and Inbound data
      {
         if ($out_flag == 1)  #outbound  (type 4)
         {
            $data_row{"type"} = 4;
            $out_flag = 0;
         }
```

```
            else  #inbound  (type 5)
            {
               $data_row{"type"} = 5;
               $out_flag = 1;
            }
         }

         elsif (($line =~ m/^BW 1:2-1/) || ($line =~ m/^Bridgewave 1:2-1/))
   #Bridgewave Outbound and Inbound data
         {
            if ($out_flag == 1)  #outbound  (type 6)
            {
               $data_row{"type"} = 6;
               $out_flag = 0;
            }
            else  #inbound  (type 7)
            {
               $data_row{"type"} = 7;
               $out_flag = 1;
            }
         }

         my %fer_data = (date => $date, time => $time, type => $data_row{"type"},
   fer => $fer);  #create hash of fer data

         #Add the smartbits data to database!
         add_smartbits_data($dbh, \%data_row);

         #Add the fer data to the database!
         add_fer_data($dbh, \%fer_data);

          }

   }  #end of looping through lines


   # Display percentage complete text
   #$num_of_files_parsed += 1; #increment the parsed files counter
   #$percent_complete = int(100 * ($num_of_files_parsed /
   $num_of_files_to_parse));
  #
  # if($percent_complete >= ($percent_mult*10))
  # {
  #    print "$percent_complete% complete\n";  #display percentage complete
  #    $percent_mult += 1;
  # }


}  #end of looping through files


#close connection with database
$dbh->disconnect;

#print "\nData Insertion Complete!\n";

#-------------------------------------------------#


#----- Subroutine Definitions -----#
```

```
#function to add smartbits data to database
#expects 2 arguments, the data base handle, and a reference to the data hash to be
   added to the DB
sub add_smartbits_data($$)
{
   my $dbh = shift;
   my $data_ref = shift;
   my %data = %{$data_ref};

   # the SQL insertion statement
   my $sql_query1 = 'INSERT INTO sprintmwb.smartbits_raw
   VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)';

   my $select = $dbh->prepare($sql_query1) or die "Couldn't prepare statement: "
   . $dbh->errstr;
   $select->execute($data{'date'}, $data{'time'}, $data{'type'},
   $data{'packet_length'},
      $data{'crc_size'}, $data{'flow_load'}, $data{'tx_stream'},
   $data{'transmitted'},
      $data{'expected'}, $data{'received'}, $data{'lost'}, $data{'in_sequence'},
      $data{'out_of_sequence'}, $data{'min_latency'}, $data{'max_latency'},
   $data{'avg_latency'},
      $data{'sf_min_latency'}, $data{'sf_max_latency'}, $data{'sf_avg_latency'},
      $data{'std_deviation'}, $data{'lb_num'}); #or die "Couldn't execute
   statement: " . $select->errstr;

   $select->finish;
}


#function to add fer data to database
#expects 2 arguments, the data base handle, and a reference to the data hash to be
   added to the DB
sub add_fer_data($$)
{
   my $dbh = shift;
   my $data_ref = shift;
   my %data = %{$data_ref};

   # the SQL insertion statement
   my $sql_query1 = 'INSERT INTO sprintmwb.fer (date, time, type, fer_raw)
   VALUES(?,?,?,?)';

   my $select = $dbh->prepare($sql_query1) or die "Couldn't prepare statement: "
   . $dbh->errstr;
   $select->execute($data{'date'}, $data{'time'}, $data{'type'}, $data{'fer'});
   #or die "Couldn't execute statement: " . $select->errstr;

   $select->finish;
}


#function to decide if the first date is before the second
#returns 1 if last_update is less recent than the given date, 0 otherwise
#expects four arguments:
# $last_update (ex: '2008-04-09')
# $year - (ex: '2007')
# $month - (ex: '09')
```

```
# $day - (ex: '05')
sub check_before($$$$)
{
   my $last_update = shift;
   my @temp = split(/-/,$last_update);
   my $year_DB = $temp[0];
   my $month_DB = $temp[1];
   my $day_DB = $temp[2];

   my $year = shift;
   my $month = shift;
   my $day = shift;

   #Convert strings to numbers
   $year_DB = $year_DB + 0;
   $month_DB = $month_DB + 0;
   $day_DB = $day_DB + 0;
   $year = $year + 0;
   $month = $month + 0;
   $day = $day + 0;


   my $result = 0;

   if($year_DB < $year)
   {
      $result = 1;
   }
   elsif($year_DB == $year)
   {
      if($month_DB < $month)
      {
         $result = 1;
      }
      elsif($month_DB == $month)
      {
         if($day_DB < $day)
         {
            $result = 1;
         }
         elsif($day_DB == $day)
         {
            $result = 1;  #dates are same, CHECK TIME?
         }
      }
   }

   return $result;

}

#---------------------------------#
```

RRvsFER.pl – Matches up the RR and FER data points and puts this into the database

```
#Filename: RRvsFER.pl
#Description: Parses the RR and FER data and puts the data into a table matching
   RRs with FERs


#!/usr/bin/perl -w
use warnings;
use strict;
use DBI;

sub add_data($$);


#------ Initialization and DB Setup ------#

#create the DB Connector
my $username = 'sprintmwb';
my $password = 'sprintmwb';
my $database = 'sprintmwb';
my $hostname = 'fiasco.ittc.ku.edu';
#db handle:
my $dbh = DBI->connect("dbi:mysql:database=$database;host=$hostname;", $username,
   $password) or die "Couldn't connect to database: " . DBI->errstr;

#----------------------------------------#


#------ Collect FER data ------#

#print "Beginning Data Insertion...\n";

#types of link
my @types = ('6');   #JUST BW FOR NOW
#location of weather
my @locations = ('Farm','Mid_Span','Nichols','North_East','South_East');

#check last date in DB for smartbits_raw table
my $sql_query_smart = "SELECT MAX(date) FROM sprintmwb.smartbits_raw"; # the SQL
   query
my $select_smart = $dbh->prepare($sql_query_smart) or die "Couldn't prepare
   statement: " . $dbh->errstr;
$select_smart->execute() or die "Couldn't execute statement: " . $select_smart-
   >errstr;
my @sql_return_smart = $select_smart->fetchrow_array;
$select_smart->finish;

my $last_update_smartbits = $sql_return_smart[0];  #last_update field now has
   timestamp from DB

#check last date in DB for rr_vs_fer table
my $sql_query_rr = "SELECT MAX(date) FROM sprintmwb.rr_vs_fer"; # the SQL query
my $select_rr = $dbh->prepare($sql_query_rr) or die "Couldn't prepare statement: "
   . $dbh->errstr;
$select_rr->execute() or die "Couldn't execute statement: " . $select_rr->errstr;
my @sql_return_rr = $select_rr->fetchrow_array;
$select_rr->finish;
```

```
my $last_update_rr_vs_fer = $sql_return_rr[0];  #last_update field now has
   timestamp from DB

#date range to work with
my $start_date = $last_update_rr_vs_fer;
my $end_date = $last_update_smartbits;


foreach my $type (@types)
{

   foreach my $location (@locations)
   {

      #print "Parsing data of type $type and location $location...\n";

      my @days_to_parse;
      my $day;

      #Get the days we are looking for and stick into an array
      my $date_query = "SELECT date FROM sprintmwb.rr WHERE date > '$start_date'
   AND date <= '$end_date' AND location = '$location' GROUP BY date";
      my $date_select = $dbh->prepare($date_query) or die "Couldn't prepare
   statement: " . $dbh->errstr;
      $date_select->execute(); #or die "Couldn't execute statement: " .
   $date_select->errstr;

      my $num_of_days = 0;  #store the number of days in search

      while ($day = $date_select->fetchrow_hashref() )
      {
         push(@days_to_parse, $day->{date});
         $num_of_days = $num_of_days + 1;
      }


      my $num_of_days_to_parse = @days_to_parse;  #find how many days we have to
   parse
      #my $num_of_days_parsed = 0;  #counter that keeps track of how many days
   have been parsed
      #my $percent_mult = 1;
      #my $percent_complete = 0;

      if($num_of_days != $num_of_days_to_parse)
      {
         print "Number of days ERROR!\n";
      }


      foreach my $date (@days_to_parse)
      {
         #obtain FER data
         my $sql_query = "SELECT date, time, fer_raw FROM sprintmwb.fer WHERE date
   = '$date' AND type = '$type'"; # the SQL query
         my $select = $dbh->prepare($sql_query) or die "Couldn't prepare
   statement: " . $dbh->errstr;
         $select->execute(); #or die "Couldn't execute statement: " . $select-
   >errstr;

         my $sql_return;
```

```
      my $sql_return_rr;
      my ($new_time, $old_time, $new_fer, $old_fer);
      my $rr;
      my $sql_query_rr = "SELECT rr_raw FROM sprintmwb.rr WHERE date = '$date'
AND time >= ? AND time < ? AND location = '$location'"; # the SQL query

      #grab first data point
      #$sql_return = $select->fetchrow_hashref();
      #$new_time = $sql_return->{time};  #time from entry in DB
      #$new_fer = $sql_return->{fer_raw};  #fer from entry in DB

      #print "First line of data: $date, $new_time, $new_fer\n";

      #loop through all the FER points
      while ( $sql_return = $select->fetchrow_hashref() )
      {
         #Reset old values to be old "new" values
         $old_time = $new_time;
         $old_fer = $new_fer;

         #get next set of "new" values
         $new_time = $sql_return->{time};  #time from entry in DB
         $new_fer = $sql_return->{fer_raw};  #fer from entry in DB

         #grab the RR point with old_time <= time < new_time
         my $select_rr = $dbh->prepare_cached($sql_query_rr) or die "Couldn't
prepare statement: " . $dbh->errstr;
         $select_rr->execute($old_time, $new_time); #or die "Couldn't execute
statement: " . $select_rr->errstr;

         #check if a valid RR point was found in the timeframe
         if ( $sql_return_rr = $select_rr->fetchrow_hashref() )
         {
            $rr = $sql_return_rr->{rr_raw};

            #now add rr and fer to rr_vs_fer table
            my %data_row = (date => $date, time => $old_time, link_type =>
$type, location => $location, rr => $rr, fer => $old_fer);  #create hash of
data to be added

            #Add the data to database!
            add_data($dbh, \%data_row);
         }
         #else{ }  #do nothing, ignore the FER point

         $select_rr->finish;
      }

      $select->finish;


      # Display percentage complete text
      #$num_of_days_parsed += 1; #increment the parsed files counter
      #$percent_complete = int(100 * ($num_of_days_parsed /
$num_of_days_to_parse));

      #if($percent_complete >= ($percent_mult*10))
      #{
      #   print "$percent_complete% complete\n";  #display percentage complete
      #   $percent_mult += 1;
```

```
         #}

      }

   }#end of looping through locations

}#end of loopsing through types


#close connection with database
$dbh->disconnect;

#print "\nData Insertion Complete!\n";

#-------------------------------------------------#


#----- Subroutine Definitions -----#

#function to add smartbits data to database
#expects 2 arguments, the data base handle, and a reference to the data hash to be
   added to the DB
sub add_data($$)
{
   my $dbh = shift;
   my $data_ref = shift;
   my %data = %{$data_ref};

   # the SQL insertion statement
   my $sql_query1 = 'INSERT INTO sprintmwb.rr_vs_fer VALUES(?,?,?,?,?,?)';

   my $select = $dbh->prepare_cached($sql_query1) or die "Couldn't prepare
   statement: " . $dbh->errstr;
   $select->execute($data{'date'}, $data{'time'}, $data{'link_type'},
   $data{'location'}, $data{'rr'}, $data{'fer'}); #or die "Couldn't execute
   statement: " . $select->errstr;

   $select->finish;
}

#---------------------------------#
```

outputRRvsFER.pl – Sorts the matched RR/FER data into histogram bins for plotting

```
#Filename: outputRRvsFER.pl
#Description: Divides the RRvsFER data into an n-bin histogram and then outputs
   this
#Conversion Note: 1 inch = 25.4 mm


#!/usr/bin/perl -w
use warnings;
use strict;
use DBI;


#------ Initialization and DB Setup ------#

#create the DB Connector
my $username = 'sprintmwb';
my $password = 'sprintmwb';
my $database = 'sprintmwb';
my $hostname = 'fiasco.ittc.ku.edu';
#db handle:
my $dbh = DBI->connect("dbi:mysql:database=$database;host=$hostname;", $username,
   $password) or die "Couldn't connect to database: " . DBI->errstr;

my $output_filename = "output_20.csv";
#create output file handle
open (my $out, ">",  $output_filename) or die "Can't open $output_filename!";

#---------------------------------------#


#------ Calculate and Output Histogram Bucket values ------#

my @link_types = ('4','6');
my @locations = ('Nichols','Farm','Mid_Span');

foreach my $link_type (@link_types)
{
   foreach my $location (@locations)
   {

      print $out "\n-----------------------------------------\n";
      print $out "Link Type: $link_type, Location: $location\n";
      print $out "-----------------------------------------\n\n";


      #Define the histogram parameters
      my $delta_r = 0.010;
      my $high_rr = 0.000;
      my $low_rr = $high_rr - $delta_r;
      my $max_rr = 1.000;


      my $average_fer;
      my $std_dev;
      my $std_dev_sum = 0.00000000;
      my $num_of_points = 0;
      my $fer_sum = 0.00000000;
```

```
   print $out "RR bin, Average FER, Std Dev, Number of Points\n";

   while($high_rr <= $max_rr)
   {
      $num_of_points = 0;
      $fer_sum = 0.00000000;
      $std_dev_sum = 0.00000000;

      #get data for the histogram
      my $sql_query = "SELECT rr, fer FROM sprintmwb.rr_vs_fer WHERE link_type
= '$link_type' AND location = '$location' AND rr > ? AND rr <= ?"; # the SQL
query
      my $select = $dbh->prepare_cached($sql_query) or die "Couldn't prepare
statement: " . $dbh->errstr;
      $select->execute("$low_rr", "$high_rr") or die "Couldn't execute
statement: " . $select->errstr;

      my $sql_return;

      #loop through all the FER points
      while ( $sql_return = $select->fetchrow_hashref() )
      {
         $fer_sum += $sql_return->{fer};
         $num_of_points += 1;
      }

      $select->finish;

      if($num_of_points > 0)
      {
         #calculate average value for fer
         $average_fer = $fer_sum / $num_of_points;

         #------------ calculate the std dev for fer ------------#
         $num_of_points = 0;

         #get data for the histogram
         my $sql_query = "SELECT rr, fer FROM sprintmwb.rr_vs_fer WHERE
link_type = '$link_type' AND location = '$location' AND rr > ? AND rr <= ?"; #
the SQL query
         my $select = $dbh->prepare_cached($sql_query) or die "Couldn't prepare
statement: " . $dbh->errstr;
         $select->execute("$low_rr", "$high_rr") or die "Couldn't execute
statement: " . $select->errstr;

         my $sql_return;

         #loop through all the FER points
         while ( $sql_return = $select->fetchrow_hashref() )
         {
            $std_dev_sum += (($average_fer - $sql_return->{fer}) ** 2);
            $num_of_points += 1;
         }

         $select->finish;

         $std_dev = sqrt($std_dev_sum / $num_of_points);

         #-------------------------------------------------------#
```

```
         print $out "> $high_rr, $average_fer, $std_dev, $num_of_points\n";
#print the results
      }
      else  #no data points in the bin
      {
         print $out "> $high_rr, 0.0, 0.0, $num_of_points\n";  #print the
results
      }

      $low_rr += $delta_r;
      $high_rr += $delta_r;

   }

   $num_of_points = 0;
   $fer_sum = 0.00000000;
   $std_dev_sum = 0.00000000;

   #get data for the last bucket (RR > max_rr)
   my $sql_query = "SELECT rr, fer FROM sprintmwb.rr_vs_fer WHERE link_type =
'$link_type' AND location = '$location' AND rr > ?"; # the SQL query
   my $select = $dbh->prepare_cached($sql_query) or die "Couldn't prepare
statement: " . $dbh->errstr;
   $select->execute($max_rr) or die "Couldn't execute statement: " . $select-
>errstr;

   my $sql_return;

   #loop through all the FER points
   while ( $sql_return = $select->fetchrow_hashref() )
   {
      $fer_sum += $sql_return->{fer};
      $num_of_points += 1;
   }

   $select->finish;

   if($num_of_points > 0)
   {
      #calculate average value for fer
      $average_fer = $fer_sum / $num_of_points;

      #------------ calculate the std dev for fer ------------#
      $num_of_points = 0;

      #get data for the last bucket (RR > max_rr)
      my $sql_query = "SELECT rr, fer FROM sprintmwb.rr_vs_fer WHERE link_type
= '$link_type' AND location = '$location' AND rr > ?"; # the SQL query
      my $select = $dbh->prepare_cached($sql_query) or die "Couldn't prepare
statement: " . $dbh->errstr;
      $select->execute($max_rr) or die "Couldn't execute statement: " .
$select->errstr;

      my $sql_return;

      #loop through all the FER points
      while ( $sql_return = $select->fetchrow_hashref() )
      {
         $std_dev_sum += (($average_fer - $sql_return->{fer}) ** 2);
```

```
            $num_of_points += 1;
         }

         $select->finish;

         $std_dev = sqrt($std_dev_sum / $num_of_points);

         #-----------------------------------------------------#

         print $out "> $max_rr, $average_fer, $std_dev, $num_of_points\n";  #print
   the results
      }
      else  #no data points in the bin
      {
         print $out "> $max_rr, 0.0, 0.0, $num_of_points\n";  #print the results
      }

   } #end of looping through locations

} #end of looping through types

#Histogram Buckets:
#RR = 0.0
#0.0 < RR <= 0.1
#0.1 < RR <= 0.2
#0.2 < RR <= 0.3
#0.3 < RR <= 0.4
#0.4 < RR <= 0.5
#0.5 < RR <= 0.6
#0.6 < RR <= 0.7
#0.7 < RR <= 0.8
#0.8 < RR <= 0.9
#0.9 < RR <= 1.0
#RR > 1.0


#close connection with database
$dbh->disconnect;

close($out);

print "\nExecution Complete!  Output sent to: $output_filename\n";

#-------------------------------------------------#
```

index.html – Main page for SprintMWB website

```
<!-- Main webpage for SprintMWB Website -->
<!-- Filename: index.html -->

<html>

<head>
   <title>Sprint MWB Project</title>
   <link href='../format.css' rel="stylesheet" type="text/css" />
   <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>

<img src='/images/ittc_logo.jpg'>
<br /><br />

<H1>Weather Station Data</H1><br />

<table width="750">
 <tr>
   <td>
   <a href="cgi-bin/site.cgi?site=Farm"><B>Pendleton Farm</B></a></td>
   <td>
   <a href="cgi-bin/site.cgi?site=Mid_Span"><B>Mid Span</B></a></td>
   <td>
   <a href="cgi-bin/site.cgi?site=Nichols"><B>Nichols</B></a></td>
   <td>
   <a href="cgi-bin/site.cgi?site=North_East"><B>North East</B></a></td>
   <td>
   <a href="cgi-bin/site.cgi?site=South_East"><B>South East</B></a></td>
  </tr>
</table>

<br /><br />
<H1>Calculators</H1><br />

<table width="750">
 <tr>
  <td>
  <a href="cgi-bin/fer.php"><B>FER Calculator</B></a></td>
  <td>
  <a href="cgi-bin/rr.php"><B>RR Calculator</B></a></td>
  <td></td>
  </tr>
</table>

<br /><br />

<h1>Plotter</h1><br />

<table width="750">
 <tr>
  <td>
  <a href="cgi-bin/r_plot.php"><b>RR/FER Plotter</b></a></td>
 </tr>
</table>
```

```
<br /><br />

<h1>Downtime Log</h1><br />

<table width="750">
 <tr>
  <td>
  <a href="cgi-bin/downtime_log.php"><b>Update Downtime Log</b></a></td>
 </tr>
</table>

<br /><br />

<table width="750" border="0">
  <tr>
    <td style="text-align:center; font-size:9pt; font-style:italic">
        <!--<img src="images/v_blue_line.gif" height="2" width="750"><br /><br />-
   ->
      <img src="../images/long_line.gif" height="10" width="750"><br
   /><br />

    Copyright 2006 University of Kansas, KU Center for Research, Inc.<br />
    </td>
  </tr>
</table>

</body>
</html>
```

format.css – Formatting details for all pages in the SprintMWB website collection

```
/* Formatting Cascading Style Sheet, written by Jake Foiles */
/* Filename: format.css */

body {
   font-family: arial, verdana, "Times New Roman";
   font-size: 0.8em;
}

fieldset {
   padding-top: 0.7em;
   padding-bottom: 1.2em;
   padding-left: 1.0em;
   padding-right: 1.0em;
   width: 700px;
}

legend {
   padding-top: .6em;
   padding-bottom: .6em;
   padding-left: .8em;
   padding-right: .8em;
}

td {
   padding-top: 0.2em;
   padding-bottom: 0.2em;
   padding-left: 0.3em;
   padding-right: 0.3em;
}

h6 {
   margin-bottom: 0.5em;
   margin-top: 1.5em;
}

#radio_button {
         float: left;
         width: 6em;
}

.downtime{
   color: red;
}
```

fer.php – Input webpage for SprintMWB Availability Calculator

```
<!-- Input webpage for SprintMWB Availability Calculator -->
<!-- Filename: fer.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Link Availabilty Calculator</title>
         <link href='../format.css' rel="stylesheet" type="text/css" />
         <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=UTF-8">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Link Availability Calculator</h2>

      <form action="calc_avail.php" method="post">

         <fieldset>
            <legend>Enter Time Interval </legend>

            <table>
               <tr>
                  <td>
                  <label for="start_date">Start Date:</label>
                  <input type="text" name="start_date" id="start_date"
   value="2007-06-30" />
                  </td>
                  <td>
                  <label for="end_date">End Date:</label>
                  <input type="text" name="end_date" id="end_date" value="2008-02-
   04" />
                  </td>
                                    <td>yyyy-mm-dd</td>
               </tr>
               <tr>
                  <td>
                  <label for="start_time">Start Time:</label>
                  <input type="text" name="start_time" id="start_time"
   value="00:00:00" />
                  </td>
                  <td>
                  <label for="end_time">End Time:</label>
                  <input type="text" name="end_time" id="end_time"
   value="23:59:59" />
                  </td>
                                    <td>hh[0-23]:mm[0-59]:ss[0-59]</td>
               </tr>
            </table>

                     <p><strong>Example:</strong> To select the link availabilty
   for the time interval of July 10, 2007 to July 17, 2007
                              between the hours of 8:30am and 8:30pm, you would
   enter the following values:</p>
                     <table>
```

```
            <tr>
               <td>
                                    Start Date: 2007-07-10
                                 </td>
                                 <td>
                                    End Date: 2007-07-17
                                 </td>
            </tr>
                           <tr>
               <td>
                                    Start Time: 08:30:00
                                 </td>
                                 <td>
                                    End Time: 16:30:00
                                 </td>
            </tr>
         </table>

                     <p>With a threshold of 0.000001 you should see these
results:<br /><br />
                           Radio Type: Both <br />
                           Start Date: 2007-07-10 <br />
                           End Date: 2007-07-17 <br />
                           Start Time: 08:30:00 <br />
                           End Time: 16:30:00 <br />
                           Def of Avail: 0.000001 <br /><br />
                           <strong>Without FEC Availability = 85.0000%
</strong><br /><br />
                           <strong>With FEC Availability = 99.8432% </strong><br
/></p>


                     <h6>*All times are local Central Time</h6>

      </fieldset>


      <fieldset>
         <legend>Define "Availability"</legend>

         <label for="def_avail">FER < </label>
         <input type="text" name="def_avail" id="def_avail" value="0.000001" />

                     <p>Here, link availability is defined as the percentage of
time that the link had a Frame Error Rate (FER)
                           of less than a specified threshold. The basic unit
being measured is over an approximately 40 sec interval.
                           During this interval, the number of frames lost is
divided by the number of frames transmitted to determine
                           a link availability for that 40 sec interval. On
average, about 3,348,000 frames of size 512 bytes are transmitted during
                           every 40 sec time interval.  This data is then used
to determine the link availability over longer time intervals.
                           For example, to find the percentage of time that the
link FER is less than 10<sup>-6</sup>, you would enter: FER < 0.000001. </p>

      </fieldset>


      <fieldset>
```

```
         <legend>Select Radio Type</legend>

         <table>
                        <tr>
                              <td>
                              <label id="radio_button"
for="radio_vendor">Without FEC</label>
                              <input type="radio" name="radio_vendor"
id="radio_vendor" value="Loea" />
                              </td>
                        </tr>
                        <tr>
                              <td>
                              <label id="radio_button"
for="radio_vendor">With FEC</label>
                              <input type="radio" name="radio_vendor"
id="radio_vendor" value="BW" />
                              </td>
                        </tr>
                        <tr>
                              <td>
                              <label id="radio_button"
for="radio_vendor">Both</label>
                              <input type="radio" name="radio_vendor"
id="radio_vendor" value="Both" checked="checked" />
                              </td>
                        </tr>
                  </table>

      </fieldset>

      <fieldset>
         <legend>Select Output</legend>

         <table>
                        <tr>
                              <td>
                              <label id="radio_button"
for="output_type">Number</label>
                              <input type="radio" name="output_type"
id="output_type" value="number" checked="checked" />
                              </td>
                        </tr>
                        <tr>
                              <td>
                              <label id="radio_button"
for="output_type">Graph</label>
                              <input type="radio" name="output_type"
id="output_type" value="graph" />
                               NOTE: Date range must include at least 4 days.
                              </td>
                        </tr>
                  </table>

      </fieldset>

      <p><input type="submit" value="Calculate Availability" /></p>

   </form>
```

```
      <img src="../images/long_line.gif" height="10" width="720" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

rr.php – Input webpage for SprintMWB Rain Rate Calculator

```
<!-- Input webpage for SprintMWB Rain Rate Calculator -->
<!-- Filename: rr.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Rain Rate Calculator</title>
         <link href='../format.css' rel="stylesheet" type="text/css" />
         <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=UTF-8">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Rain Rate Calculator</h2>

      <form action="calc_rr.php" method="post">

         <fieldset>
            <legend>Enter Time Interval </legend>

            <table>
               <tr>
                  <td>
                  <label for="start_date">Start Date:</label>
                  <input type="text" name="start_date" id="start_date"
   value="2007-09-01" />
                  </td>
                  <td>
                  <label for="end_date">End Date:</label>
                  <input type="text" name="end_date" id="end_date" value="2008-02-
   04" />
                  </td>
                                    <td>yyyy-mm-dd</td>
               </tr>
               <tr>
                  <td>
                  <label for="start_time">Start Time:</label>
                  <input type="text" name="start_time" id="start_time"
   value="00:00:00" />
                  </td>
                  <td>
                  <label for="end_time">End Time:</label>
                  <input type="text" name="end_time" id="end_time"
   value="23:59:59" />
                  </td>
                                    <td>hh[0-23]:mm[0-59]:ss[0-59]</td>
               </tr>
            </table>

                     <p><strong>Example:</strong> To select the rain rate for
   the time interval of October 10, 2007 to November 10, 2007
                              between the hours of 8:30am and 8:30pm, you would
   enter the following values:</p>
                     <table>
```

```
            <tr>
               <td>
                                    Start Date: 2007-10-10
                                 </td>
                                 <td>
                                    End Date: 2007-11-10
                                 </td>
            </tr>
                           <tr>
               <td>
                                    Start Time: 08:30:00
                                 </td>
                                 <td>
                                    End Time: 16:30:00
                                 </td>
            </tr>
         </table>

                     <p>With a threshold of 0.01 you should see these
results:<br /><br />
                           Location: All <br />
                           Start Date: 2007-10-10 <br />
                           End Date: 2007-11-10 <br />
                           Start Time: 08:30:00 <br />
                           End Time: 16:30:00 <br />
                           Rain Rate Threshold: 0.01 <br /><br />
                           <strong>Farm Rain Rate = 0.6986% </strong><br /><br
/>
            <strong>Mid-Span Rain Rate = 0.5383% </strong><br /><br />
            <strong>Nichols Rain Rate = 1.2970% </strong><br /><br />
            <strong>North East Rain Rate = 0.4992% </strong><br /><br />
                           <strong>South East Rain Rate = 0.4644% </strong><br
/></p>


                     <h6>*All times are local Central Time</h6>

      </fieldset>


      <fieldset>
         <legend>Define the Rain Rate Threshold (in/hr)</legend>

         <label for="def_rr">Rain >= </label>
         <input type="text" name="def_rr" id="def_rr" value="0.01" />

                     <p>The Rain Rate Threshold is the cutoff value that will be
used to determine what percentage of time had a
            rain rate of larger than this value.  The smallest amount above
zero is 0.01 inches/hour.</p>

      </fieldset>


      <fieldset>
         <legend>Select Location</legend>

         <table>
                           <tr>
                                 <td>
```

```
                                    <label id="radio_button"
for="location">Farm</label>
                                    <input type="radio" name="location"
id="location" value="Farm" />
                                    </td>
                              </tr>
                              <tr>
                                    <td>
                                    <label id="radio_button" for="location">Mid-
Span</label>
                                    <input type="radio" name="location"
id="location" value="Mid-Span" />
                                    </td>
                              </tr>
                  <tr>
                                    <td>
                                    <label id="radio_button"
for="location">Nichols</label>
                                    <input type="radio" name="location"
id="location" value="Nichols" />
                                    </td>
                              </tr>
                  <tr>
                                    <td>
                                    <label id="radio_button" for="location">North
East</label>
                                    <input type="radio" name="location"
id="location" value="North East" />
                                    </td>
                              </tr>
                  <tr>
                                    <td>
                                    <label id="radio_button" for="location">South
East</label>
                                    <input type="radio" name="location"
id="location" value="South East" />
                                    </td>
                              </tr>
                              <tr>
                                    <td>
                                    <label id="radio_button" for="location">All
5</label>
                                    <input type="radio" name="location"
id="location" value="All" checked="checked" />
                                    </td>
                              </tr>
                        </table>

            </fieldset>

            <fieldset>
               <legend>Select Output</legend>

               <table>
                              <tr>
                                    <td>
                                    <label id="radio_button"
for="output_type">Number</label>
                                    <input type="radio" name="output_type"
id="output_type" value="number" checked="checked" />
```

```
                                    </td>
                                </tr>
                                <tr>
                                    <td>
                                    <label id="radio_button"
   for="output_type">Graph</label>
                                    <input type="radio" name="output_type"
   id="output_type" value="graph" />
                  NOTE: Date range must include at least 4 days.
                                    </td>
                                </tr>
                            </table>

         </fieldset>

         <p><input type="submit" value="Calculate Rain Rate" /></p>

      </form>


      <img src="../images/long_line.gif" height="10" width="720" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

r_plot.php – Input webpage for SprintMWB RR vs FER Plotter

```
<!-- Input webpage for SprintMWB RR vs FER plotter -->
<!-- Filename: r_plot.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Rain and FER Plotter</title>
         <link href='../format.css' rel="stylesheet" type="text/css" />
         <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=UTF-8">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Rain and FER Plotter</h2>

      <form action="calc_rr_plot.php" method="post">

         <fieldset>
            <legend>Enter Time Interval </legend>

            <table>
               <tr>
                  <td>
                  <label for="start_date">Date:</label>
                  <input type="text" name="start_date" id="start_date"
   value="2007-10-21" />
                  </td>
                  <!--
                  <td>
                  <label for="end_date">End Date:</label>
                  <input type="text" name="end_date" id="end_date" value="2008-02-
   04" />
                  </td>
                  -->
                  <td>yyyy-mm-dd</td>
                                    <td></td>
               </tr>
               <tr>
                  <td>
                  <label for="start_time">Start Time:</label>
                  <input type="text" name="start_time" id="start_time"
   value="00:00:00" />
                  </td>
                  <td>
                  <label for="end_time">End Time:</label>
                  <input type="text" name="end_time" id="end_time"
   value="23:59:59" />
                  </td>
                                    <td>hh[0-23]:mm[0-59]:ss[0-59]</td>
               </tr>
            </table>

            <p>A rain rate and FER plot will be generated for the day that you
   specify</p>
```

```
                  <h6>*All times are local Central Time</h6>

      </fieldset>

      <fieldset>
         <legend>Select Location</legend>

         <table>
                           <tr>
                                 <td>
                                 <label id="radio_button"
for="location">Farm</label>
                                 <input type="radio" name="location"
id="location" value="Farm" />
                                 </td>
                           </tr>
                           <tr>
                                 <td>
                                 <label id="radio_button" for="location">Mid-
Span</label>
                                 <input type="radio" name="location"
id="location" value="Mid-Span" />
                                 </td>
                           </tr>
            <tr>
                                 <td>
                                 <label id="radio_button"
for="location">Nichols</label>
                                 <input type="radio" name="location"
id="location" value="Nichols" />
                                 </td>
                           </tr>
            <tr>
                                 <td>
                                 <label id="radio_button" for="location">North
East</label>
                                 <input type="radio" name="location"
id="location" value="North East" />
                                 </td>
                           </tr>
            <tr>
                                 <td>
                                 <label id="radio_button" for="location">South
East</label>
                                 <input type="radio" name="location"
id="location" value="South East" />
                                 </td>
                           </tr>
            <tr>
                                 <td>
                                 <label id="radio_button" for="location">All
5</label>
                                 <input type="radio" name="location"
id="location" value="All" checked="checked" />
                                 </td>
                           </tr>
                  </table>

      </fieldset>
```

```
      <fieldset>
         <legend>Select Radio Type</legend>

         <table>
                           <tr>
                                 <td>
                                 <label id="radio_button" for="type">Without
   FEC</label>
                                 <input type="radio" name="type" id="type"
   value="4" checked="checked" />
                                 </td>
                           </tr>
                           <tr>
                                 <td>
                                 <label id="radio_button" for="type">With
   FEC</label>
                                 <input type="radio" name="type" id="type"
   value="6" />
                                 </td>
                           </tr>
                           <!--
            <tr>
                                 <td>
                                 <label id="radio_button"
   for="radio_vendor">Both</label>
                                 <input type="radio" name="radio_vendor"
   id="radio_vendor" value="Both" checked="checked" />
                                 </td>
                           </tr>
            -->
                  </table>

      </fieldset>

      <p><input type="submit" value="Generate RR and FER Plots" /></p>

   </form>


   <img src="../images/long_line.gif" height="10" width="720" />
   <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

  </body>
</html>
```

downtime_log.php – Main page for the SprintMWB downtime log

```
<!-- Main page for the SprintMWB downtime log -->
<!-- Filename: downtime_log.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Downtime Log Form</title>
         <link href='../format.css' rel="stylesheet" type="text/css" />
         <meta HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=UTF-8">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Downtime Log Form</h2>

      <form action="show_log.php" method="post">

         <fieldset>
            <legend>View Current Downtime Log</legend>

            <p><input type="submit" value="View Log" /></p>

         </fieldset>

      </form>

      <br />

      <form action="add_log.php" method="post">

         <fieldset>
            <legend>Add New Downtime Log Entry</legend>

            <table>
               <tr>
                  <td>
                  <label id="radio_button" for="date">Date:</label>
                  <input type="text" name="date" id="date" />
                  </td>
                  <td>yyyy-mm-dd</td>
               </tr>
            </table>

            <br />

            <table>
               <tr>
                  <td>
                  <label id="radio_button" for="radio_vendor">Without FEC</label>
                  <input type="radio" name="radio_vendor" id="radio_vendor"
   value="Loea" />
                  </td>
               </tr>
               <tr>
```

```
                  <td>
                  <label id="radio_button" for="radio_vendor">With FEC</label>
                  <input type="radio" name="radio_vendor" id="radio_vendor"
   value="BW" />
                  </td>
               </tr>
               <tr>
                  <td>
                  <label id="radio_button" for="radio_vendor">Both</label>
                  <input type="radio" name="radio_vendor" id="radio_vendor"
   value="Both" checked="checked" />
                  </td>
               </tr>
            </table>

            <br />

            <table>
               <tr>
                  <td>
                  <label id="radio_button" for="comments">Comments:</label>
                  <input type="text" name="comments" id="comments" />
                  </td>
                  <td>(optional)</td>
               </tr>
            </table>

            <p><input type="submit" value="Add New Entry" /></p>

         </fieldset>

      </form>

      <br />
      <br />

      <img src="../images/long_line.gif" height="10" width="720" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

calc_avail.php – Processes the Availability calculator input and either generates a numerical result or passes it on to be graphed.

```
<!-- Processes the Availability calculator input and either generates a -->
<!--   numerical result or passes it on to be graphed. -->
<!-- Filename: calc_avail.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Link Availabilty Calculator</title>
         <link href="../format.css" rel="stylesheet" type="text/css" />
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Link Availability Calculator</h2>


<?php

   include ("connect.inc");
   include ("functions.inc");

   $start_date = $_POST['start_date'];
   $end_date = $_POST['end_date'];
   $start_time = $_POST['start_time'];
   $end_time = $_POST['end_time'];
   $def_of_avail = $_POST['def_avail'];
   $radio_vendor = $_POST['radio_vendor'];
   $output_type = $_POST['output_type'];

   if($radio_vendor == 'Loea')
   {
      $type = 4;
   }
   elseif($radio_vendor == 'BW')
   {
      $type = 6;
   }

   $radio_conversion = convert_radio($radio_vendor);

   print "Radio Type: $radio_conversion <br />";
   print "Start Date: $start_date <br />";
   print "End Date: $end_date <br />";
   print "Start Time: $start_time <br />";
   print "End Time: $end_time <br />";
   print "Def of Avail: $def_of_avail <br /><br />";

   //Check if any days in the submitted range have a downtime log
   if($radio_vendor == 'Both')
   {
      $downtime_query = "SELECT date, radio_vendor, comments FROM downtime_log
   WHERE date >= '$start_date' AND date <= '$end_date'";
```

```
}
else
{
   $downtime_query = "SELECT date, radio_vendor, comments FROM downtime_log
WHERE date >= '$start_date' AND date <= '$end_date' AND (radio_vendor =
'$radio_vendor' OR radio_vendor = 'Both')";
}

$downtime_result = mysql_query($downtime_query) or die('Query failed: ' .
mysql_error());

$num_of_days = 0;  //store the number of days in search

while ($day = mysql_fetch_row($downtime_result))
{
   $days[] = "$day[0]";
   $radio_vendors[] = "$day[1]";
   $comments[] = "$day[2]";
   $num_of_days = $num_of_days + 1;
}

//clear up all sql stuff and memory
mysql_free_result($downtime_result);

if($num_of_days > 0)  //there were downtime logs for one or more days...
{
   print "Your query's date range has the following downtime logs:<br />";

   $count = 0;
   foreach($days as $one_day)
   {
      $radio_conversion = convert_radio($radio_vendors[$count]);

      print "<span class=\"downtime\">$one_day - $radio_conversion radio(s)
were down.</span><br />";  //Can add the downtime_log comments to this line if
needed

      $count = $count + 1;
   }

   print "<br />";
}


//check if the day has valid data (just checks whether the date has data for BW
OR Loea)
$validity_query = "SELECT date FROM fer WHERE date >= '$start_date' AND date <=
'$end_date' AND time >= '$start_time' AND time <= '$end_time' AND (type = '6'
OR type = '4') GROUP BY date";
$validity_result = mysql_query($validity_query) or die('Query failed: ' .
mysql_error());
$num_rows = mysql_num_rows($validity_result);

//clear up all sql stuff and memory
mysql_free_result($validity_result);

if($num_rows < 1)  //the specified day has no valid data in the DB
{
   print "<span class=\"downtime\">The date range $start_date - $end_date has
no valid data, try a different range.</span><br />";
```

```
   print "<br />";
}

else  //The user's input checks out fine, plot the graphs!
{

   if($output_type == 'number') // display numerical calculation
   {

      if($radio_vendor == 'Both')
      {
         // BW
         $query_total_bw = "SELECT COUNT(*) FROM fer WHERE date >=
'$start_date' AND date <= '$end_date' AND time >= '$start_time' AND time <=
'$end_time' AND type = '6'";
         $result_total_bw = mysql_query($query_total_bw) or die('Query failed:
' . mysql_error());
         $query_avail_bw = "SELECT COUNT(*) FROM fer WHERE date >=
'$start_date' AND date <= '$end_date' AND time >= '$start_time' AND time <=
'$end_time' AND type = '6' AND fer_raw < '$def_of_avail'";
         $result_avail_bw = mysql_query($query_avail_bw) or die('Query failed:
' . mysql_error());

         // Loea
         $query_total_l = "SELECT COUNT(*) FROM fer WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
type = '4'";
         $result_total_l = mysql_query($query_total_l) or die('Query failed: '
. mysql_error());
         $query_avail_l = "SELECT COUNT(*) FROM fer WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
type = '4' AND fer_raw < '$def_of_avail'";
         $result_avail_l = mysql_query($query_avail_l) or die('Query failed: '
. mysql_error());


         // calculate availability
         // BW
         $num_total_row_bw = mysql_fetch_row($result_total_bw);
         $num_avail_row_bw = mysql_fetch_row($result_avail_bw);
         // Loea
         $num_total_row_l = mysql_fetch_row($result_total_l);
         $num_avail_row_l = mysql_fetch_row($result_avail_l);


         $availability_bw = $num_avail_row_bw[0]/$num_total_row_bw[0];
         $availability_l = $num_avail_row_l[0]/$num_total_row_l[0];

         print "<h4>Without FEC Availability =
";printf("%01.4f",100*$availability_l);echo"% </h4>";
         print "<h4>With FEC Availability =
";printf("%01.4f",100*$availability_bw);echo"% </h4>";

         // Free memory
         mysql_free_result($result_total_bw);
         mysql_free_result($result_avail_bw);
         mysql_free_result($result_total_l);
         mysql_free_result($result_avail_l);
```

```
      }
      else
      {
         // Perform SQL queries
         $query_total = "SELECT COUNT(*) FROM fer WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
type = '$type'";
         $result_total = mysql_query($query_total) or die('Query failed: ' .
mysql_error());

         $query_avail = "SELECT COUNT(*) FROM fer WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
type = '$type' AND fer_raw < '$def_of_avail'";
         $result_avail = mysql_query($query_avail) or die('Query failed: ' .
mysql_error());

         // Calculate the Availability
         $num_total_row = mysql_fetch_row($result_total);
         $num_avail_row = mysql_fetch_row($result_avail);

         $availability = $num_avail_row[0]/$num_total_row[0];

         print "<h4>Availability = ";printf("%01.4f",100*$availability);echo"%
</h4>";

         // Free result memory
         mysql_free_result($result_total);
         mysql_free_result($result_avail);
      }

   }
   elseif($output_type == 'graph') // graph availability
   {
       // Perform SQL queries
      //$query_rows = "SELECT COUNT(*) FROM fer WHERE date >= '$start_date' AND
date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND type
= '$type' GROUP BY date";
      //$result_rows = mysql_query($query_rows) or die('Query failed: ' .
mysql_error());
      //$num_of_rows = mysql_num_rows($result_rows);
      // Free result memory
      //mysql_free_result($result_rows);

      //print "$num_of_rows<br />";

      if($radio_vendor != 'Both')  //print a single graph
      {
         echo "<img
src=\"graph.php?start_date=$start_date&end_date=$end_date&start_time=$start_ti
me&end_time=$end_time&def_avail=$def_of_avail&radio_vendor=$radio_vendor&outpu
t_type=$output_type\">\n";
         print "<br />";
      }

      elseif($radio_vendor == 'Both')  //print both vendor radio graphs
      {
         echo "<img
src=\"graph.php?start_date=$start_date&end_date=$end_date&start_time=$start_ti
me&end_time=$end_time&def_avail=$def_of_avail&radio_vendor=Loea&output_type=$o
utput_type\">\n";
```

```
            print "<br />";
            echo "<img
   src=\"graph.php?start_date=$start_date&end_date=$end_date&start_time=$start_ti
   me&end_time=$end_time&def_avail=$def_of_avail&radio_vendor=BW&output_type=$out
   put_type\">\n";
            print "<br />";
         }
      }
   }

   // Close DB connection
   mysql_close($dbh);

?>


      <img src="../images/long_line.gif" height="10" width="750" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

calc_rr.php – Processes the RR calculator input and either generates a numerical result or passes it on to be graphed.

```
<!-- Processes the RR calculator input and either generates a numerical -->
<!--   result or passes it on to be graphed. -->
<!-- Filename: calc_rr.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Rain Rate Calculator</title>
         <link href="../format.css" rel="stylesheet" type="text/css" />
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Rain Rate Calculator</h2>


<?php

   //include ("graph_rr.php");
   //include ("jpgraph_1.19/src/jpgraph.php");
   //include ("jpgraph_1.19/src/jpgraph_scatter.php");
   include ("connect.inc");

   $start_date = $_POST['start_date'];
   $end_date = $_POST['end_date'];
   $start_time = $_POST['start_time'];
   $end_time = $_POST['end_time'];
   $def_of_rr = $_POST['def_rr'];
   $location = $_POST['location'];
   $output_type = $_POST['output_type'];

   //Assign locations in the form that the DB expects
   if($location == 'Farm')
   {
      $loc = 'Farm';
   }
   elseif($location == 'Mid-Span')
   {
      $loc = 'Mid_Span';
   }
   elseif($location == 'Nichols')
   {
      $loc = 'Nichols';
   }
   elseif($location == 'North East')
   {
      $loc = 'North_East';
   }
   elseif($location == 'South East')
   {
      $loc = 'South_East';
   }
```

```
print "Location: $location <br />";
print "Start Date: $start_date <br />";
print "End Date: $end_date <br />";
print "Start Time: $start_time <br />";
print "End Time: $end_time <br />";
print "Rain Rate Threshold: $def_of_rr <br /><br />";

if($output_type == 'number') // display numerical calculation
{

   if($location == 'All')
   {
      // Farm
      $query_total_farm = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'Farm'";
      $result_total_farm = mysql_query($query_total_farm) or die('Query failed:
' . mysql_error());
      $query_rain_farm = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'Farm' AND rr_raw >= '$def_of_rr'";
      $result_rain_farm = mysql_query($query_rain_farm) or die('Query failed: '
. mysql_error());

      // Mid-Span
      $query_total_mid = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'Mid_Span'";
      $result_total_mid = mysql_query($query_total_mid) or die('Query failed: '
. mysql_error());
      $query_rain_mid = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'Mid_Span' AND rr_raw >= '$def_of_rr'";
      $result_rain_mid = mysql_query($query_rain_mid) or die('Query failed: ' .
mysql_error());

      // Nichols
      $query_total_nic = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'Nichols'";
      $result_total_nic = mysql_query($query_total_nic) or die('Query failed: '
. mysql_error());
      $query_rain_nic = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'Nichols' AND rr_raw >= '$def_of_rr'";
      $result_rain_nic = mysql_query($query_rain_nic) or die('Query failed: ' .
mysql_error());

      // North East
      $query_total_ne = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'North_East'";
      $result_total_ne = mysql_query($query_total_ne) or die('Query failed: ' .
mysql_error());
      $query_rain_ne = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date' AND
date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'North_East' AND rr_raw >= '$def_of_rr'";
      $result_rain_ne = mysql_query($query_rain_ne) or die('Query failed: ' .
mysql_error());
```

```
      // South East
      $query_total_se = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date'
AND date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'South_East'";
      $result_total_se = mysql_query($query_total_se) or die('Query failed: ' .
mysql_error());
      $query_rain_se = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date' AND
date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = 'South_East' AND rr_raw >= '$def_of_rr'";
      $result_rain_se = mysql_query($query_rain_se) or die('Query failed: ' .
mysql_error());


      // calculate availability
      // Farm
      $num_total_row_farm = mysql_fetch_row($result_total_farm);
      $num_rain_row_farm = mysql_fetch_row($result_rain_farm);
      // Mid-Span
      $num_total_row_mid = mysql_fetch_row($result_total_mid);
      $num_rain_row_mid = mysql_fetch_row($result_rain_mid);
      // Nichols
      $num_total_row_nic = mysql_fetch_row($result_total_nic);
      $num_rain_row_nic = mysql_fetch_row($result_rain_nic);
      // North East
      $num_total_row_ne = mysql_fetch_row($result_total_ne);
      $num_rain_row_ne = mysql_fetch_row($result_rain_ne);
      // South East
      $num_total_row_se = mysql_fetch_row($result_total_se);
      $num_rain_row_se = mysql_fetch_row($result_rain_se);


      $rain_percent_farm = $num_rain_row_farm[0]/$num_total_row_farm[0];
      $rain_percent_mid = $num_rain_row_mid[0]/$num_total_row_mid[0];
      $rain_percent_nic = $num_rain_row_nic[0]/$num_total_row_nic[0];
      $rain_percent_ne = $num_rain_row_ne[0]/$num_total_row_ne[0];
      $rain_percent_se = $num_rain_row_se[0]/$num_total_row_se[0];

      print "<h4>Farm Rain Rate =
";printf("%01.4f",100*$rain_percent_farm);echo"% </h4>";
      print "<h4>Mid-Span Rain Rate =
";printf("%01.4f",100*$rain_percent_mid);echo"% </h4>";
      print "<h4>Nichols Rain Rate =
";printf("%01.4f",100*$rain_percent_nic);echo"% </h4>";
      print "<h4>North East Rain Rate =
";printf("%01.4f",100*$rain_percent_ne);echo"% </h4>";
      print "<h4>South East Rain Rate =
";printf("%01.4f",100*$rain_percent_se);echo"% </h4>";

      // Free memory
      mysql_free_result($result_total_farm);
      mysql_free_result($result_rain_farm);
      mysql_free_result($result_total_mid);
      mysql_free_result($result_rain_mid);
      mysql_free_result($result_total_nic);
      mysql_free_result($result_rain_nic);
      mysql_free_result($result_total_ne);
      mysql_free_result($result_rain_ne);
      mysql_free_result($result_total_se);
      mysql_free_result($result_rain_se);
```

```
   }
   else
   {
      // Perform SQL queries
      $query_total = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date' AND
date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = '$loc'";
      $result_total = mysql_query($query_total) or die('Query failed: ' .
mysql_error());

      $query_rain = "SELECT COUNT(*) FROM rr WHERE date >= '$start_date' AND
date <= '$end_date' AND time >= '$start_time' AND time <= '$end_time' AND
location = '$loc' AND rr_raw >= '$def_of_rr'";
      $result_rain = mysql_query($query_rain) or die('Query failed: ' .
mysql_error());

      // Calculate the Availability
      $num_total_row = mysql_fetch_row($result_total);
      $num_rain_row = mysql_fetch_row($result_rain);

      $rain_percent = $num_rain_row[0]/$num_total_row[0];

      print "<h4>Rain Rate = ";printf("%01.4f",100*$rain_percent);echo"%
</h4>";

      // Free result memory
      mysql_free_result($result_total);
      mysql_free_result($result_rain);
   }

}
elseif($output_type == 'graph') // graph availability
{

   if($location != 'All')  //print a single graph
   {
      //$inputs = array
($start_date,$end_date,$start_time,$end_time,$def_of_rr,$location);
      //daily($inputs);

      echo "<img
src=\"graph_rr.php?start_date=$start_date&end_date=$end_date&start_time=$start
_time&end_time=$end_time&def_rr=$def_of_rr&location=$loc\">\n";
      print "<br />";
   }

   else  //print all location graphs
   {
      echo "<img
src=\"graph_rr.php?start_date=$start_date&end_date=$end_date&start_time=$start
_time&end_time=$end_time&def_rr=$def_of_rr&location=Farm\">\n";
      print "<br />";
      echo "<img
src=\"graph_rr.php?start_date=$start_date&end_date=$end_date&start_time=$start
_time&end_time=$end_time&def_rr=$def_of_rr&location=Mid_Span\">\n";
      print "<br />";
      echo "<img
src=\"graph_rr.php?start_date=$start_date&end_date=$end_date&start_time=$start
_time&end_time=$end_time&def_rr=$def_of_rr&location=Nichols\">\n";
```

```
         print "<br />";
         echo "<img
   src=\"graph_rr.php?start_date=$start_date&end_date=$end_date&start_time=$start
   _time&end_time=$end_time&def_rr=$def_of_rr&location=North_East\">\n";
         print "<br />";
         echo "<img
   src=\"graph_rr.php?start_date=$start_date&end_date=$end_date&start_time=$start
   _time&end_time=$end_time&def_rr=$def_of_rr&location=South_East\">\n";
         print "<br />";
      }
   }


   // Close DB connection
   mysql_close($dbh);

?>


      <img src="../images/long_line.gif" height="10" width="750" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

calc_rr_plot.php – Processes the RR vs FER plotter input and passes it on to be graphed.

```
<!-- Processes the RR vs FER plotter input and passes it on to be graphed. -->
<!-- Filename: calc_rr_plot.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Rain and FER Plotter</title>
         <link href="../format.css" rel="stylesheet" type="text/css" />
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Rain and FER Plotter</h2>


<?php

   include ("connect.inc");
   include ("functions.inc");

   $start_date = $_POST['start_date'];
   //$end_date = $_POST['end_date'];
   $start_time = $_POST['start_time'];
   $end_time = $_POST['end_time'];
   $location = $_POST['location'];
   $type = $_POST['type'];

 if($type == '4')
 {
  $radio_type = 'Loea';
 }
 elseif($type == '6')
 {
  $radio_type = 'BW';
 }

   //Assign locations in the form that the DB expects
   if($location == 'Farm')
   {
      $loc = 'Farm';
   }
   elseif($location == 'Mid-Span')
   {
      $loc = 'Mid_Span';
   }
   elseif($location == 'Nichols')
   {
      $loc = 'Nichols';
   }
   elseif($location == 'North East')
   {
      $loc = 'North_East';
```

```
}
elseif($location == 'South East')
{
   $loc = 'South_East';
}

$radio_conversion = convert_radio($radio_type);

print "Location: $location <br />";
print "Radio Type: $radio_conversion <br />";
print "Start Date: $start_date <br />";
//print "End Date: $end_date <br />";
print "Start Time: $start_time <br />";
print "End Time: $end_time <br /><br />";


//Check if the submitted day is in the downtime log
if($radio_vendor == 'Both')
{
   $downtime_query = "SELECT date, radio_vendor, comments FROM downtime_log
WHERE date='$start_date'";
}
else
{
   $downtime_query = "SELECT date, radio_vendor, comments FROM downtime_log
WHERE date='$start_date' AND (radio_vendor = '$radio_vendor' OR radio_vendor =
'Both')";
}

$downtime_result = mysql_query($downtime_query) or die('Query failed: ' .
mysql_error());

$num_rows = mysql_num_rows($downtime_result);



if($num_rows > 0)  //the specified day has a downtime log
{
   $day = mysql_fetch_row($downtime_result);
   $day = "$day[0]";
   $radio_vendors = "$day[1]";
   $comments = "$day[2]";

   //clear up all sql stuff and memory
   mysql_free_result($downtime_result);

   print "Your query's date range has the following downtime log:<br />";
   $radio_conversion = convert_radio($radio_vendors);
   print "<span class=\"downtime\">$day - $radio_conversion radio(s) were
down.</span><br />";  //Can add the downtime_log comments to this line if
needed
   print "<br />";
}


else  //the specified day has no downtime log
{

   //check if the day has valid data
   if($location == 'All')
```

```
   {
      $validity_query = "SELECT date FROM rr_vs_fer WHERE date='$start_date'
AND link_type = '$type'";
   }
   else
   {
      $validity_query = "SELECT date FROM rr_vs_fer WHERE date='$start_date'
AND location = '$loc' AND link_type = '$type'";
   }
   $validity_result = mysql_query($validity_query) or die('Query failed: ' .
mysql_error());
   $num_rows = mysql_num_rows($validity_result);

   //clear up all sql stuff and memory
   mysql_free_result($validity_result);

   if($num_rows < 1)  //the specified day has no valid data in the DB
   {
      print "<span class=\"downtime\">The date $start_date has no valid data,
try a different date.</span><br />";
      print "<br />";
   }

   else  //The user's input checks out fine, plot the graphs!
   {

      //graph rain plots
      if($location != 'All')  //print a single graph
      {
         //$inputs = array
($start_date,$end_date,$start_time,$end_time,$def_of_rr,$location);
         //daily($inputs);

         echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=$loc&type=$type\">\n";
         print "<br />";
         echo "<img
src=\"graph_fer_plot.php?start_date=$start_date&start_time=$start_time&end_tim
e=$end_time&location=$loc&type=$type\">\n";
         print "<br />";
      }

      else  //print all graphs
      {
         //RR
         echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=Nichols&type=$type\">\n";
         print "<br />";
         //FER
         echo "<img
src=\"graph_fer_plot.php?start_date=$start_date&start_time=$start_time&end_tim
e=$end_time&location=Nichols&type=$type\">\n";
         print "<br />";

         echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=Farm&type=$type\">\n";
         print "<br />";
```

```
         echo "<img
src=\"graph_fer_plot.php?start_date=$start_date&start_time=$start_time&end_tim
e=$end_time&location=Farm&type=$type\">\n";
         print "<br />";

         echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=Mid_Span&type=$type\">\n";
         print "<br />";
         echo "<img
src=\"graph_fer_plot.php?start_date=$start_date&start_time=$start_time&end_tim
e=$end_time&location=Mid_Span&type=$type\">\n";
         print "<br />";

         echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=North_East&type=$type\">\n";
         print "<br />";
         echo "<img
src=\"graph_fer_plot.php?start_date=$start_date&start_time=$start_time&end_tim
e=$end_time&location=North_East&type=$type\">\n";
         print "<br />";

         echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=South_East&type=$type\">\n";
         print "<br />";
         echo "<img
src=\"graph_fer_plot.php?start_date=$start_date&start_time=$start_time&end_tim
e=$end_time&location=South_East&type=$type\">\n";
         print "<br />";

         //print both radio graphs
      }

   }

}

//graph fer plots
//if($location != 'All')  //print a single graph
//{
   //$inputs = array
($start_date,$end_date,$start_time,$end_time,$def_of_rr,$location);
   //daily($inputs);

//   echo "<img
src=\"graph_rr_plot.php?start_date=$start_date&start_time=$start_time&end_time
=$end_time&location=$loc\">\n";
//   print "<br />";
//}

//else  //print both radio graphs
//{


//}


// Close DB connection
```

```
   mysql_close($dbh);

?>

      <img src="../images/long_line.gif" height="10" width="750" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

show_log.php – Displays the downtime log entries in a table format.

```
<!-- Displays the downtime log entries in table format. -->
<!-- Filename: show_log.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Downtime Log</title>
         <link href="../format.css" rel="stylesheet" type="text/css" />
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Downtime Log</h2>

      <br />


<?php

   include ("connect.inc");
   include ("functions.inc");

   //Print out all downtime log entries
   $downtime_query = "SELECT date, radio_vendor, comments FROM downtime_log ORDER
   BY date";
   $downtime_result = mysql_query($downtime_query) or die('Query failed: ' .
   mysql_error());

   $num_of_days = 0;  //store the number of days in search

   while ($day = mysql_fetch_row($downtime_result))
   {
      $days[] = "$day[0]";
      $radio_vendors[] = "$day[1]";
      $comments[] = "$day[2]";
      $num_of_days = $num_of_days + 1;
   }

   //clear up all sql stuff and memory
   mysql_free_result($downtime_result);

   if($num_of_days > 0)  //there were downtime logs for one or more days...
   {
      print "<table>";
      print "<caption>Downtime Log Entries:</caption>";

      print "<tr>";
      print "<th>Date</th>";
      print "<th>Radio(s)</th>";
      print "<th>Comments</th>";
      print "</tr>";

      $count = 0;
      foreach($days as $one_day)
```

```
      {
         $radio_conversion = convert_radio($radio_vendors[$count]);
         print "<tr>";
         print "<td>$one_day</td>";
         print "<td>$radio_conversion</td>";
         print "<td>$comments[$count]</td>";
         print "</tr>";
         $count = $count + 1;
      }

      print "</table>";
      print "<br />";
   }
   else
   {
      print "No downtime log entries found<br />";
   }


?>

      <br />

      <img src="../images/long_line.gif" height="10" width="750" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

add_log.php – Adds a downtime log entry to the database.

```
<!-- Adds a downtime log entry to the Database -->
<!-- Filename: add_log.php -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>
   <head>
      <title>Sprint MWB Link Availabilty Calculator</title>
         <link href="../format.css" rel="stylesheet" type="text/css" />
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </head>

   <body>
      <a href="http://www.ittc.ku.edu"><img src='../images/ittc_logo.jpg' /></a>

      <h2>Sprint MWB Link Availability Calculator</h2>

      <br />


<?php

   include ("functions.inc");

   $username = 'sprintmwb';
   $password = 'sprintmwb';
   $database = 'sprintmwb';
   $hostname = 'fiasco.ittc.ku.edu';

   // Connect and select the database
   $dbh = mysql_connect($hostname, $username, $password) or die('Could not
   connect: ' . mysql_error());
   mysql_select_db($database) or die('Could not select database');

   //Grab user input from HTML form
   $date = $_POST['date'];
   $radio = $_POST['radio_vendor'];
   $comments = $_POST['comments'];

   //check if the date already has a downtime log
   $validity_query = "SELECT date FROM downtime_log WHERE date = '$date'";
   $validity_result = mysql_query($validity_query) or die('Query failed: ' .
   mysql_error());
   $num_rows = mysql_num_rows($validity_result);

   //clear up all sql stuff and memory
   mysql_free_result($validity_result);

   if($num_rows < 1)  //the specified day has no downtime log entry
   {

      if (($comments == '') || ($comments == NULL))
      {
         $insert_query = "INSERT INTO sprintmwb.downtime_log
   VALUES('$date','$radio',NULL)";
      }
      else
```

```
      {
         $insert_query = "INSERT INTO sprintmwb.downtime_log
   VALUES('$date','$radio','$comments')";
      }
      mysql_query($insert_query) or die('Query failed: ' . mysql_error());

      print "<p>The entry for $date was successfully added to the Downtime
   Log</p>";

   }
   else  //the specified day already has a downtime log entry
   {
      print "<p>$date already has a downtime log entry, please select a different
   date.</p>";
   }


?>

      <br />

      <img src="../images/long_line.gif" height="10" width="750" />
      <h6><em>Copyright 2006 University of Kansas, KU Center for Research,
   Inc.</em></h6>

   </body>
</html>
```

graph.php – Plots the link availability.

```
<?php

   //Plots the link availability
   //Filename: graph.php

   include ("jpgraph_1.19/src/jpgraph.php");
   include ("jpgraph_1.19/src/jpgraph_scatter.php");
   include ("connect.inc");
   include ("functions.inc");

   $start_date = $_REQUEST['start_date'];
   $end_date = $_REQUEST['end_date'];
   $start_time = $_REQUEST['start_time'];
   $end_time = $_REQUEST['end_time'];
   $def_of_avail = $_REQUEST['def_avail'];
   $radio_vendor = $_REQUEST['radio_vendor'];
   $output_type = $_REQUEST['output_type'];

   // buy doing this with this simple if statement we are not limited
   // to two vendors, we can add as many as we want by just changing the type
   function.
   if($output_type == 'graph') // graph availability
   {
      $type = type($radio_vendor);
      $inputs = array
   ($start_date,$end_date,$start_time,$end_time,$def_of_avail,$radio_vendor,$type
   );
      daily($inputs);
   }



//Start of functions

//graph function
//inputs: fer values    array $values
//   days         array $days
//   name of radio scalar $radio
//      num of days in search $num_of_days
//return: image to be displayed
//NOTE: This function uses jpgraph_1.19 to do the image
//      creation
function graph ($values,$days,$radio,$num_of_days) {


 if($num_of_days < 10) {
   $scaler = 24.0;
 }else {
   $scaler = 12.0;
 }

 if($num_of_days > '4') {
  $x_axis_size = intval($num_of_days * $scaler); //scale x_axis depending on
   number of days
  $graph = new Graph($x_axis_size,'400','auto');
  //$graph = new Graph('1500','400','auto');
  $graph->SetMargin(50,25,25,90);  //SetMargin(x_left, x_right, y_top, y_bottom)
  $graph->SetScale("textlin");
```

```
  $graph->SetShadow();

  // Specify X and Y labels
  $graph->xaxis->SetTickLabels($days);
  $graph->xaxis->SetLabelAngle(90);
  $graph->xaxis->SetFont(FF_FONT1,FS_NORMAL,8);
  $graph->xaxis->title->Set("Date");
  $graph->xaxis->SetTitlemargin(50);
  $graph->yaxis->title->Set("%" );
  $graph->yaxis->SetTitlemargin(25);

  // Specify Graph title and fonts
  $radio_conversion = convert_radio($radio);
  $graph->title->Set("Link Availability - $radio_conversion");
  $graph->title->SetFont(FF_FONT1,FS_BOLD);
  $graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
  $graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);

  //Specify type of graph to create
  $sp1 = new ScatterPlot($values);

  //Add graph to stack and output image
  $graph->Add($sp1);
  $graph->Stroke();
 }else {
   print "Error Not enough data points to graph properly<br>";
   print "Please return to <a href=\"https://weather.ittc.ku.edu/cgi-
   bin/fer.php\">weather.ittc.ku.edu</a><br>";
   print "and resubmit your request.<br>";
 }

}


//daily function
//inputs: search parameters array $inputs
//
//return: no return values
//NOTE: this function calls the graphing functions
function daily ($inputs) {

   //Get values from array
   $start_date = $inputs[0];
   $end_date = $inputs[1];
   $start_time = $inputs[2];
   $end_time = $inputs[3];
   $def_of_avail = $inputs[4];
   $radio_vendor = $inputs[5];
   $type = $inputs[6];

   // Get the days we are looking for and stick into an array
   $date_query = "SELECT date FROM fer WHERE date >= '$start_date' AND date <=
   '$end_date' AND type = '$type' GROUP BY date";
   $date_result = mysql_query($date_query) or die('Query failed: ' .
   mysql_error());

   $num_of_days = 0;  //store the number of days in search

   while ($day = mysql_fetch_row($date_result))
```

```
   {
      $days[] = "$day[0]";
      $num_of_days = $num_of_days + 1;
   }

   //clear up all sql stuff and memory
   mysql_free_result($date_result);

   // loop through the specified days and get all fer values
   foreach($days as $one_day)
   {
      // Perform SQL queries
      $query_total = "SELECT COUNT(*) FROM fer WHERE date = '$one_day' AND time >=
   '$start_time' AND time <= '$end_time' AND type = $type";
      $result_total = mysql_query($query_total) or die('Query failed: ' .
   mysql_error());

      $query_avail = "SELECT COUNT(*) FROM fer WHERE date = '$one_day' AND time >=
   '$start_time' AND time <= '$end_time' AND type = $type AND fer_raw <
   '$def_of_avail'";
      $result_avail = mysql_query($query_avail) or die('Query failed: ' .
   mysql_error());

      // Calculate the Availability
      $num_total_row = mysql_fetch_row($result_total);
      $num_avail_row = mysql_fetch_row($result_avail);

      $availability = $num_avail_row[0]/$num_total_row[0];

      //put values into an array
      $values[] = round(100*$availability, 4);

      mysql_free_result($result_total);
      mysql_free_result($result_avail);
   }

   // graph results !
   graph($values,$days,$radio_vendor,$num_of_days);

   // Close DB connection
   mysql_close($dbh);

}

// function type
// inputs: name of vendor scalar $vendor
//
// retrun: returns numerical value of vendor
// description: function returns numerical value
//   of vendor that has been assigned in database
//   to make sure to get the correct link values
function type ($vendor)
{
   if($vendor == 'Loea')
   {
      return(4);
   }
   elseif($vendor == 'BW')
   {
      return(6);
```

```
    }
}
?>
```

graph_rr.php – Plots the RR calculator output.

```
<?php

   //Plots the RR calculator output
   //Filename: graph_rr.php

   //include ("graph_rr.php");
   include ("jpgraph_1.19/src/jpgraph.php");
   include ("jpgraph_1.19/src/jpgraph_scatter.php");
   include ("connect.inc");

   $start_date = $_REQUEST['start_date'];
   $end_date = $_REQUEST['end_date'];
   $start_time = $_REQUEST['start_time'];
   $end_time = $_REQUEST['end_time'];
   $def_of_rr = $_REQUEST['def_rr'];
   $location = $_REQUEST['location'];

   $inputs = array
   ($start_date,$end_date,$start_time,$end_time,$def_of_rr,$location);
   daily($inputs);


//Start of functions

//graph function
//inputs: fer values    array $values
//   days        array $days
//   name of radio scalar $radio
//      num of days in search $num_of_days
//return: image to be displayed
//NOTE: This function uses jpgraph_1.19 to do the image
//      creation
function graph ($values,$days,$location,$num_of_days) {

 $x_axis_size = intval($num_of_days * 11.0); //scale x_axis depending on number of
   days

 $graph = new Graph($x_axis_size,'400','auto');
 //$graph = new Graph('1500','400','auto');
 $graph->SetMargin(50,25,25,90);  //SetMargin(x_left, x_right, y_top, y_bottom)
 $graph->SetScale("textlin");

 $graph->SetShadow();

 // Specify X and Y labels
 $graph->xaxis->SetTickLabels($days);
 $graph->xaxis->SetLabelAngle(90);
 $graph->xaxis->SetFont(FF_FONT1,FS_NORMAL,8);
 $graph->xaxis->title->Set("Date");
 $graph->xaxis->SetTitlemargin(50);
 $graph->yaxis->title->Set("%" );
 $graph->yaxis->SetTitlemargin(25);

 // Specify Graph title and fonts
 $graph->title->Set("Rain Rate - $location");
 $graph->title->SetFont(FF_FONT1,FS_BOLD);
 $graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
 $graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);
```

```
 //Specify type of graph to create
 $sp1 = new ScatterPlot($values);

 //Add graph to stack and output image
 $graph->Add($sp1);
 $graph->Stroke();
}


//daily function
//inputs: search parameters array $inputs
//
//return: no return values
//NOTE: this function calls the graphing functions
function daily ($inputs) {

   //Get values from array
   $start_date = $inputs[0];
   $end_date = $inputs[1];
   $start_time = $inputs[2];
   $end_time = $inputs[3];
   $def_of_rr = $inputs[4];
   $location = $inputs[5];

   // Get the days we are looking for and stick into an array
   $date_query = "SELECT date FROM rr WHERE date >= '$start_date' AND date <=
   '$end_date' AND location = '$location' GROUP BY date";
   $date_result = mysql_query($date_query) or die('Query failed: ' .
   mysql_error());

   $num_of_days = 0;  //store the number of days in search

   while ($day = mysql_fetch_row($date_result))
   {
      $days[] = "$day[0]";
      $num_of_days = $num_of_days + 1;
   }

   //clear up all sql stuff and memory
   mysql_free_result($date_result);

   // loop through the specified days and get all fer values
   foreach($days as $one_day)
   {
      // Perform SQL queries
      $query_total = "SELECT COUNT(*) FROM rr WHERE date = '$one_day' AND time >=
   '$start_time' AND time <= '$end_time' AND location = '$location'";
      $result_total = mysql_query($query_total) or die('Query failed: ' .
   mysql_error());

      $query_rain = "SELECT COUNT(*) FROM rr WHERE date = '$one_day' AND time >=
   '$start_time' AND time <= '$end_time' AND location = '$location' AND rr_raw >=
   '$def_of_rr'";
      $result_rain = mysql_query($query_rain) or die('Query failed: ' .
   mysql_error());

      // Calculate the Availability
      $num_total_row = mysql_fetch_row($result_total);
      $num_rain_row = mysql_fetch_row($result_rain);
```

```
      $rain_percent = $num_rain_row[0]/$num_total_row[0];

      //put values into an array
      $values[] = round(100*$rain_percent, 4);

      mysql_free_result($result_total);
      mysql_free_result($result_rain);
   }

   // graph results !
   graph($values,$days,$location,$num_of_days);

   // Close DB connection
   mysql_close($dbh);

}


?>
```

graph_rr_plot.php – Plots the matched RR data.

```
<?php

   //Plots out the matched RR data
   //Filename: graph_rr_plot.php

   //include ("graph_rr.php");
   include ("jpgraph_1.19/src/jpgraph.php");
   include ("jpgraph_1.19/src/jpgraph_scatter.php");
   include ("connect.inc");
   include ("functions.inc");

   $start_date = $_REQUEST['start_date'];
   //$end_date = $_REQUEST['end_date'];
   $start_time = $_REQUEST['start_time'];
   $end_time = $_REQUEST['end_time'];
   $location = $_REQUEST['location'];
   $type = $_REQUEST['type'];

   $inputs = array ($start_date,$start_time,$end_time,$location,$type);
   daily($inputs);


//Start of functions

//graph function
//inputs: fer values    array $values
//   days        array $days
//   name of radio scalar $radio
//      num of days in search $num_of_days
//return: image to be displayed
//NOTE: This function uses jpgraph_1.19 to do the image
//      creation
function graph ($values,$times,$location,$type,$start_date,$num_of_times) {

 if($type == '4')
 {
  $radio_type = 'Loea';
 }
 elseif($type == '6')
 {
  $radio_type = 'BW';
 }

 $x_axis_size = intval($num_of_times * 9.5); //scale x_axis depending on number of
   days

 $graph = new Graph($x_axis_size,'400','auto');
 //$graph = new Graph('1500','400','auto');
 $graph->SetMargin(60,25,25,90);  //SetMargin(x_left, x_right, y_top, y_bottom)
 $graph->SetScale("textlin");

 $graph->SetShadow();

 // Specify X and Y labels
 $graph->xaxis->SetTickLabels($times);
 $graph->xaxis->SetLabelAngle(90);
 $graph->xaxis->SetFont(FF_FONT1,FS_NORMAL,8);
 $graph->xaxis->title->Set("Time");
```

```
 $graph->xaxis->SetTitlemargin(50);
 $graph->yaxis->title->Set("RR (in/hr)" );
 $graph->yaxis->SetTitlemargin(40);

 // Specify Graph title and fonts
 $radio_conversion = convert_radio($radio_type);
 $graph->title->Set("Rain Rate - $location - $radio_conversion - $start_date");
 $graph->title->SetFont(FF_FONT1,FS_BOLD);
 $graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
 $graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);

 //Specify type of graph to create
 $sp1 = new ScatterPlot($values);

 //Add graph to stack and output image
 $graph->Add($sp1);
 $graph->Stroke();
}


//daily function
//inputs: search parameters array $inputs
//
//return: no return values
//NOTE: this function calls the graphing functions
function daily ($inputs) {

   //Get values from array
   $start_date = $inputs[0];
   //$end_date = $inputs[1];
   $start_time = $inputs[1];
   $end_time = $inputs[2];
   $location = $inputs[3];
   $type = $inputs[4];

   // Get the times we are looking for and stick into an array
   $time_query = "SELECT time FROM rr_vs_fer WHERE date = '$start_date' AND time
   >= '$start_time' AND time <= '$end_time' AND location = '$location' AND
   link_type = '$type' GROUP BY time";
   $time_result = mysql_query($time_query) or die('Query failed: ' .
   mysql_error());

   $num_of_times = 0;  //store the number of times in search

   while ($time = mysql_fetch_row($time_result))
   {
      $times[] = "$time[0]";
      $num_of_times = $num_of_times + 1;
   }

   // Get the rr's we are looking for and stick into an array
   $rr_query = "SELECT rr FROM rr_vs_fer WHERE date = '$start_date' AND time >=
   '$start_time' AND time <= '$end_time' AND location = '$location' AND link_type
   = '$type' GROUP BY time";
   $rr_result = mysql_query($rr_query) or die('Query failed: ' . mysql_error());

   $num_of_times2 = 0;  //store the number of times in search

   while ($value = mysql_fetch_row($rr_result))
   {
```

```
      $values[] = round($value[0], 2);
      $num_of_times2 = $num_of_times2 + 1;
   }

   mysql_free_result($time_result);
   mysql_free_result($rr_result);

   // graph results !
   graph($values,$times,$location,$type,$start_date,$num_of_times);

   // Close DB connection
   mysql_close($dbh);

}


?>
```

graph_fer_plot.php – Plots the matched FER data.

```
<?php

   //Plots out the matched FER data
   //Filename: graph_fer_plot.php

   //include ("graph_rr.php");
   include ("jpgraph_1.19/src/jpgraph.php");
   include ("jpgraph_1.19/src/jpgraph_scatter.php");
   include ("connect.inc");
   include ("functions.inc");

   $start_date = $_REQUEST['start_date'];
   //$end_date = $_REQUEST['end_date'];
   $start_time = $_REQUEST['start_time'];
   $end_time = $_REQUEST['end_time'];
   $location = $_REQUEST['location'];
   $type = $_REQUEST['type'];

   $inputs = array ($start_date,$start_time,$end_time,$location,$type);
   daily($inputs);


//Start of functions

//graph function
//inputs: fer values    array $values
//   days        array $days
//   name of radio scalar $radio
//      num of days in search $num_of_days
//return: image to be displayed
//NOTE: This function uses jpgraph_1.19 to do the image
//      creation
function graph ($values,$times,$location,$type,$start_date,$num_of_times) {

 if($type == '4')
 {
  $radio_type = 'Loea';
 }
 elseif($type == '6')
 {
  $radio_type = 'BW';
 }

 $x_axis_size = intval($num_of_times * 9.5); //scale x_axis depending on number of
   days

 $graph = new Graph($x_axis_size,'400','auto');
 //$graph = new Graph('1500','400','auto');
 $graph->SetMargin(80,25,25,80);  //SetMargin(x_left, x_right, y_top, y_bottom)
 $graph->SetScale("textlin");

 $graph->SetShadow();

 // Specify X and Y labels
 $graph->xaxis->SetTickLabels($times);
 $graph->xaxis->SetLabelAngle(90);
 $graph->xaxis->SetFont(FF_FONT1,FS_NORMAL,8);
 $graph->xaxis->title->Set("Time");
```

```
 $graph->xaxis->SetTitlemargin(50);
 $graph->yaxis->title->Set("FER" );
 $graph->yaxis->SetTitlemargin(40);

 // Specify Graph title and fonts
 $radio_conversion = convert_radio($radio_type);
 $graph->title->Set("FER - $location - $radio_conversion - $start_date");
 $graph->title->SetFont(FF_FONT1,FS_BOLD);
 $graph->xaxis->title->SetFont(FF_FONT1,FS_BOLD);
 $graph->yaxis->title->SetFont(FF_FONT1,FS_BOLD);

 //Specify type of graph to create
 $sp1 = new ScatterPlot($values);

 //Add graph to stack and output image
 $graph->Add($sp1);
 $graph->Stroke();
}


//daily function
//inputs: search parameters array $inputs
//
//return: no return values
//NOTE: this function calls the graphing functions
function daily ($inputs) {

   //Get values from array
   $start_date = $inputs[0];
   //$end_date = $inputs[1];
   $start_time = $inputs[1];
   $end_time = $inputs[2];
   $location = $inputs[3];
   $type = $inputs[4];

   // Get the times we are looking for and stick into an array
   $time_query = "SELECT time FROM rr_vs_fer WHERE date = '$start_date' AND time
   >= '$start_time' AND time <= '$end_time' AND location = '$location' AND
   link_type = '$type' GROUP BY time";
   $time_result = mysql_query($time_query) or die('Query failed: ' .
   mysql_error());

   $num_of_times = 0;  //store the number of times in search

   while ($time = mysql_fetch_row($time_result))
   {
      $times[] = "$time[0]";
      $num_of_times = $num_of_times + 1;
   }

   // Get the fer's we are looking for and stick into an array
   $rr_query = "SELECT fer FROM rr_vs_fer WHERE date = '$start_date' AND time >=
   '$start_time' AND time <= '$end_time' AND location = '$location' AND link_type
   = '$type' GROUP BY time";
   $rr_result = mysql_query($rr_query) or die('Query failed: ' . mysql_error());

   $num_of_times2 = 0;  //store the number of times in search

   while ($value = mysql_fetch_row($rr_result))
   {
```

```
      $values[] = round($value[0], 7);
      $num_of_times2 = $num_of_times2 + 1;
   }

   mysql_free_result($time_result);
   mysql_free_result($rr_result);

   // graph results !
   graph($values,$times,$location,$type,$start_date,$num_of_times);

   // Close DB connection
   mysql_close($dbh);

}


?>
```