Getting To the Train on Time

(OK, really it's "When the Heck Will We Get Home!")

Eric Whitenton
2007

Abstract

As with any transportation system, the Maryland MARC commuter train has occasional delays. This report outlines an easy to implement system for automatically sending text messages to users cell phones informing then of these delays. It uses a variety of widely available technologies such as PHP and XML files.

1.0 Introduction

I use the MARC commuter train. There are occasional delays due to breakdowns and bad weather. It would be useful to know the status of the trains, even when I am not near a PC. My cell phone is not web enabled, but I can receive text messages. MARC offers a service where they will email you if the train is "severely" late, but you receive nothing if the train is only "somewhat" late. Most cell phones may be sent text messages via email. However, only the first 100 to 200 characters are received, with the rest being truncated. The purpose of this project is to create a system where I may control the timing, content and formatting of the MARC train related text messages I receive.

There are, of course, some constraints. I will set this up on a PC at work and I am not allowed to have a server. I do not want the solution to be overly complex or difficult to maintain. I do not want a solution based on an immature technology still in a high state of flux.

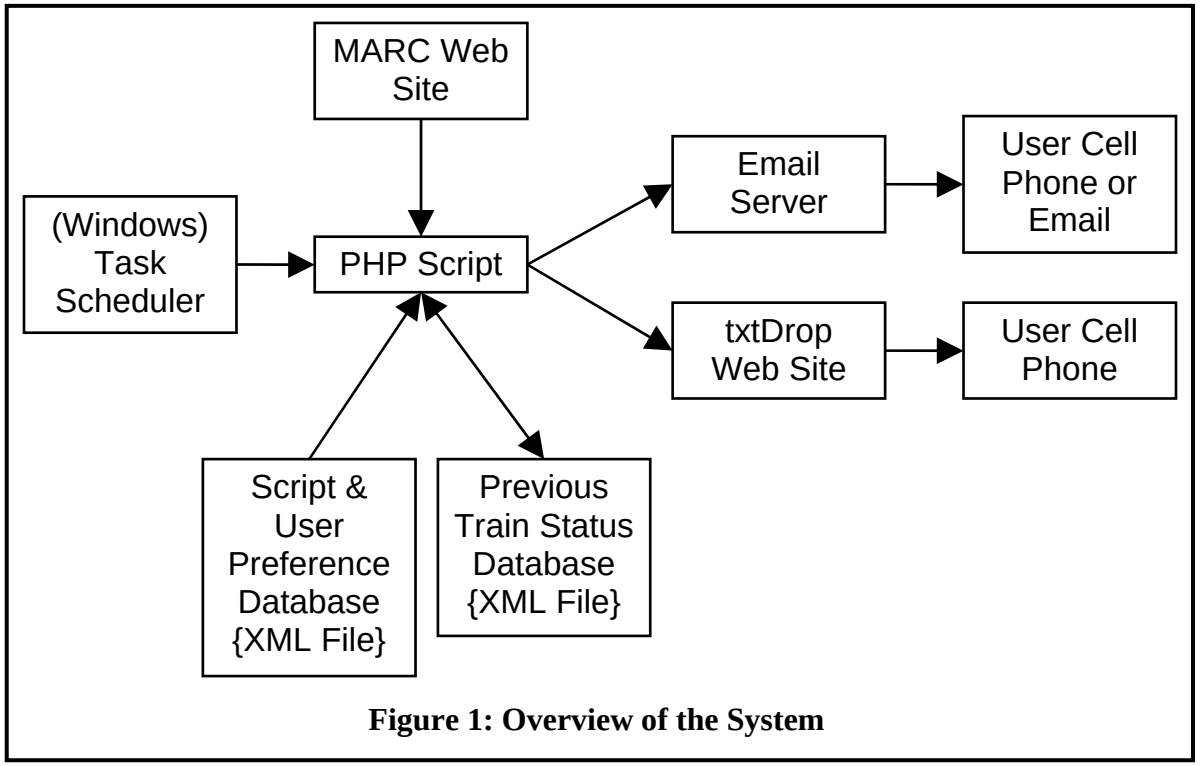


Figure 1: Overview of the System

2.0 A Solution

Figure 1 shows an overview of the system developed. My implementation of the system uses the Windows operating system, but others could be used instead. The Task Scheduler (Section 2.1) invokes a PHP Script (Section 2.2) on a regular, repeated basis. When invoked, the PHP Script reads information from XML Files (Section 2.3) regarding Script and user preferences, as well as the state of the trains the last time the Script was invoked. The Script next checks the MARC Web Site (Section 2.4) to get the current statuses of the MARC trains. For each user of the system, the Script will compare the current statuses with those the user was last informed of. If any of the statuses have changed, the user is sent a text message. This is accomplished either by sending an email (see

freesms.1888usa.com for email addresses) via an email server or by submitting a web form to a free text messaging web site (Section 2.5).

2.1 The Windows Task Scheduler

The PHP Script is written to work with even a rudimentary Task Scheduler. All that is required of the Scheduler is that it invokes the Script on a repeating basis, say every 10 minutes. The Script checks to see if users are interested in receiving text messages on the current day and time and only sends text messages to them when desired. However, the system is most efficient when the Scheduler only invokes the Script when at least one user is interested. Why should the Script be invoked on Saturday or Sunday when the trains do not even run?

Figure 2 shows screen captures from the Windows Task Scheduler where the Scheduler was setup to only invoke the Script on reasonable days and times. Figure 2A shows how tasks may be browsed, created or deleted in Windows Explorer. Once created, a dialog box may be launched

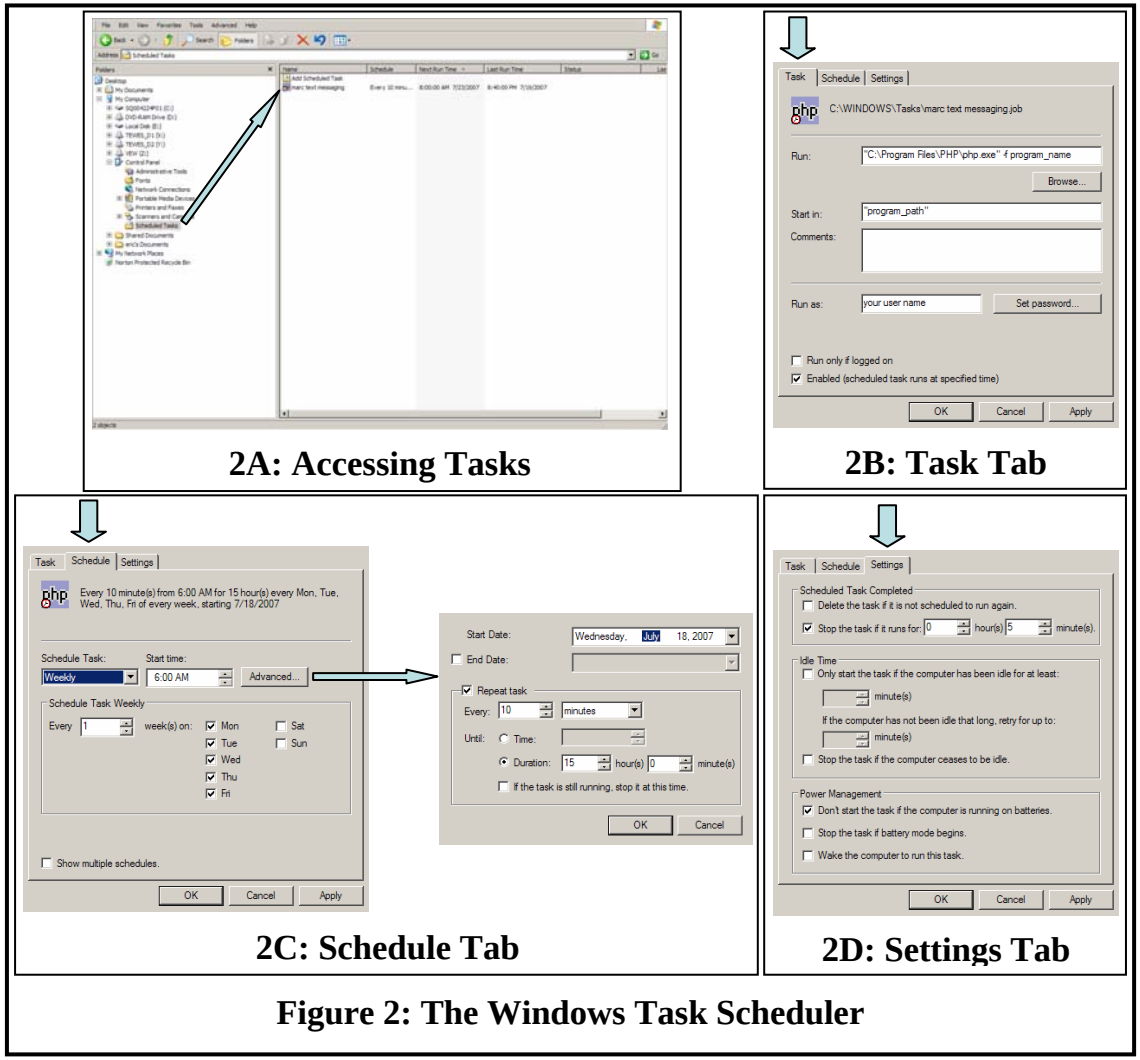


Figure 2: The Windows Task Scheduler

by double-clicking on the task icon. The behavior of the task may be controlled from this dialog box, which has three tabs. In the Task Tab (Figure 2B), the task, startup folder and user ID is defined. The Schedule Tab (Figure 2C) allows the days, time of day and frequency to be specified while the Settings Tab (Figure 2D) allows various other settings to be specified.

2.2 The PHP Script

Appendix A shows the Script. To help in understanding the logic behind the Script, a simplified algorithm is shown in Figure 3. It is fairly straight forward and was discussed briefly in Section 2.0.

```
Read user_preference database file
If current time/date is outside of range Then Exit
Read previous_train_statuses database file
Get current train statuses from the MARC web site
For each user
   If the current time/date is of interest to this user Then
      Initialize message to NULL
      For each train_of_interest to this user
         If the status has changed since the last time this user received a
         message regarding this train_of_interest Then
            Append the status for this train_of_interest to message
         End If
      Next train_of_interest
      If message is not NULL Then
         Send message to user via preferred method (txtSend or email)
      End If
   End If
Next user
Write current train statuses to previous_train_statuses database file
Exit
```

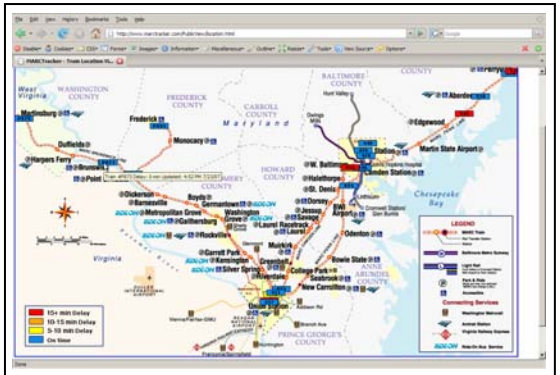
Figure 3: Simplified Algorithm behind the PHP Script

The actual Script has refinements not shown in Figure 3. For example, each user has an associated Boolean variable *active* which may be set to false to temporarily disable a user. As another example, there is a special user called “group” which defines group behavior of all of the users. The Script generally checks the preferences of the group before it checks the preferences for each individual user. If the variable *active* for user group is set to false, all users are treated as though they are inactive. This allows for easy control over the Script behavior without having to modify the preferences for each and every user.

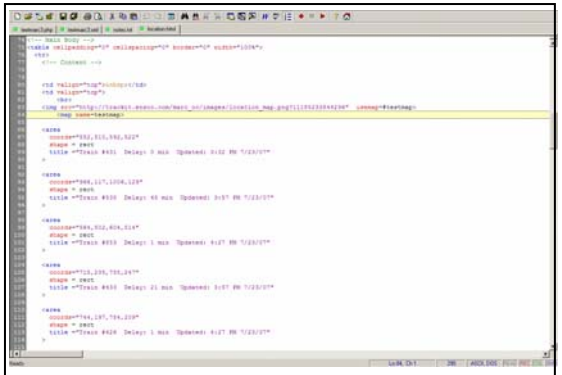
2.3 The XML Files

The first XML file is shown in Appendix B. It is used to hold relatively static information such as user preferences and program variables which do not normally change. The Script only reads this file and never writes to it. It has an embedded DTD (Document Type Definition) which defines the legal structure of the XML data. Some folks feel that DTD will eventually be replaced by XML Schema language, also known as XSD (XML Schema Definition). **If this file is modified, the second XML file should be deleted to insure the users and trains match.**

Appendix C shows the second XML file. It is both read from and written to by the Script. If it does not exist, it is created by the Script. It holds "temporary" variables used by the Script between invocations by the Scheduler. The serialize and unserialize functions in PHP are used to convert the values of the variables to and from text. **As mentioned above, if the first XML file is modified, this second file should be deleted.** It will be re-created when the Script is invoked.



4A: How it looks in a Browser



4B: The HTML Code

Figure 4: MARC Web Site

2.4 The MARC Train Web Site

Figures 4A and 4B show the browser view and the source code view of a page from the MARC train web site (www.marctracker.com/PublicView/location.html). The primary feature on the page is an image of a map with the current locations of the trains on it. When a user positions the mouse curser over a position marker for a train, text appears indicating the name and current delay for that train. By processing the text in the HTML file, information such as the train names, positions on the map (in pixels) and current delays are extracted by the Script.

5A: Page for Creating HTML Form

5B: Resulting Page Showing Created HTML Code

Figure 5: The txtDrop Web Site

2.5 The txtDrop Web Site

Figure 5A shows a page on the txtDrop web site (txtdrop.com/generate.php) which allows users to enter their phone number and generate HTML code which is a HTML form for sending a text message through the txtDrop web site. Shown in Figure 5B, this HTML code may be copied from the text box and pasted into the users own web page. In our case, a modified version of this text is used by the Script to send the text message.

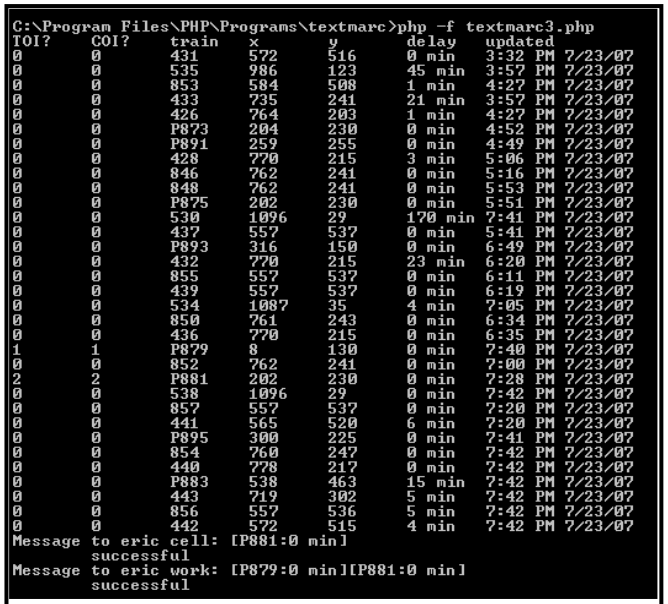


Figure 6: One Invocation of the Script

3.0 The Program in Action

Figure 6 shows the command window during one invocation of the Script. First, a table of the train statuses is output, where each train currently running generates one row of the table. The table has the following columns:

Column #	Caption	Purpose
1	TOI?	This shows the number of users for whom this is a “train of interest.” That is, the number of users who care about this train at this time.
2	COI?	This shows the number of users for whom this is a “change of interest.” That is, the number of users who need to be notified that the status of the train has changed.
3	train	The name of the train.
4	x	The x coordinate (in pixels) of the trains position on the map.
5	y	The y coordinate (in pixels) of the trains position on the map.
6	delay	The current delay for the train.
7	updated	The last time the information was last updated by MARC.

Finally, if any text messages must be sent, the user, message (currently just the train names and current delays) and whether or not the message was successfully sent is displayed for each message. The Script now exits.

4.0 Conclusions

A working system was designed, built and tested. It is a “smart client” and not a server. It supports multiple commuters and may be run on many operating systems. It is both simple and based on stable but not stagnant technology, so it should be easy to maintain.

Features which could be added in the future include: Picture messages (PHP has simple image processing functions). Have the PHP script control the task scheduler to improve efficiency (however, this makes the PHP script more OS dependent). Include more information in the text messages. Incorporate other transportation systems such as the Metro System. Create a separate program to allow changing user preferences without directly editing the XML file.

List of Appendices

Appendix A – PHP Script
Appendix B – Script and User Preference XML File
Appendix C – Previous Train Status XML File

Appendix A: PHP Script

```
<?php


 /* define functions and procedures */


 /* function to repeatedly remove and replace text until all desired text is removed */
 function clean($text_to_remove,$text_to_replace,$text)
 {
  $cleaned_text=$text;
  while(stripos($cleaned_text,$text_to_remove)):
    $cleaned_text=str_replace($text_to_remove,$text_to_replace,$cleaned_text);
  endwhile;
  return $cleaned_text;
 }

 /* function to clean extra delimiters and white spaces from file */
 function clean_page($text)
 {
  $cleaned_text=$text;
  $cleaned_text=clean("\t"  ," " ,$cleaned_text);
  $cleaned_text=clean("  "  ," " ,$cleaned_text);
  $cleaned_text=clean("\n " ,"\n",$cleaned_text);
  $cleaned_text=clean("\n\r","\n",$cleaned_text);
  $cleaned_text=clean("\n\n","\n",$cleaned_text);
  return $cleaned_text;
 }

 /* function to post form */
 /* modified from example taken from netevil.org/blog/2006/nov/http-post-from-php-without-curl */
 function do_post_request($url, $data, $optional_headers = null)
```

```
{
  $params = array
    (
    'http' => array
      (
      'method' => 'POST',
      'content' => $data
      )
    );
  if ($optional_headers !== null) {$params['http']['header'] = $optional_headers;}
  $ctx = stream_context_create($params);
  $fp = @fopen($url, 'rb', false, $ctx);
  if (!$fp)
  {
    $response = "Problem connecting to ".$url;
  }
  else
  {
    $response = @stream_get_contents($fp);
    if ($response === false)
    {
      $response = "Problem reading data from ".$url;
    }
  }
  return $response;
}


/* program start */


/* get user variables from xml file */
/* get xml file data object */
if (file_exists('textmarc3.xml'))
```

```
  {$xml = simplexml_load_file('textmarc3.xml');}
else
  {exit('Failed to find xml file.');}
/* time zone */
date_default_timezone_set((string)$xml->timezone);
/* flags to indicate which outputs should be sent to screen */
$print_table = ((string)$xml->print_table == "true");
$print_cell_message = ((string)$xml->print_cell_message == "true");
/* file to read/save status info if $program_mode is scheduled */
$program_status_file = (string)$xml->program_status_file;
/* using this page to determine current marc train status */
$marc_location_url = (string)$xml->marc_location_url;
/* $marc_location_url = "location.html"; */  /* local copy of file for testing purposes */
/* user list */
/* "group" is a special user NOT included in this list */
/* some values use "group" to describe needs of the group of all users */
$user_list = array();
foreach ($xml->user_list->user as $user)
{if ($user->handle != "group") {$user_list = array_merge($user_list, array((string)$user->handle));};}
/* true if service is active */
/* normally, user "group" is true if any users service is active */
/* however, user "group" could be made false to disable all users */
$user_service_active = array();
foreach ($xml->user_list->user as $user)
{$user_service_active = array_merge($user_service_active, array((string)$user->handle => (string)$user->user_service_active));}
/* days to check status, array of flags, Mon Tue Wed Thu Fri Sat Sun */
/* normally, user "group" is true if any user needs to check status that day */
/* however, user "group" could be made false to disable all users that day */
$days_to_check_status = array();
foreach ($xml->user_list->user as $user)
{
  $days_to_check_status_this_user=array();
  foreach ($user->days_to_check_status as $day)
```

```
   {$days_to_check_status_this_user = array_merge($days_to_check_status_this_user, array((string)$day->handle => ((string)$day->value=="true")));}
   $days_to_check_status = array_merge($days_to_check_status, array((string)$user->handle => $days_to_check_status_this_user));
  }
  /* start and stop times of day to check status, military time, hh:mm */
  /* normally, user "group" is earliest/latest time of day for any user */
  /* however, user "group" could be used to set bounds on when program is active */
  $time_of_day_to_start_checking_status = array();
  foreach ($xml->user_list->user as $user)
  {$time_of_day_to_start_checking_status = array_merge($time_of_day_to_start_checking_status, array((string)$user->handle => (string)$user->time_of_day_to_start_checking_status));}
  $time_of_day_to_stop_checking_status = array();
  foreach ($xml->user_list->user as $user)
  {$time_of_day_to_stop_checking_status = array_merge($time_of_day_to_stop_checking_status, array((string)$user->handle => (string)$user->time_of_day_to_stop_checking_status));}
  /* true if send text message via email */
  /* there is no "group" user for this variable */
  $user_contact_via_email = array();
  foreach ($xml->user_list->user as $user)
  {
   if ($user->handle != "group")
   {$user_contact_via_email = array_merge($user_contact_via_email, array((string)$user->handle => ((string)$user->user_contact_via_email=="true")));};
  }
  /* email address or txtsend code */
  /* there is no "group" user for this variable */
  $user_address = array();
  foreach ($xml->user_list->user as $user)
  {
   if ($user->handle != "group")
   {$user_address = array_merge($user_address, array((string)$user->handle => (string)$user->user_address));};
  }
  /* trains of interest */
  /* there is no "group" user for this variable */
```

```
$trains_of_interest = array();
foreach ($xml->user_list->user as $user)
{
  if ($user->handle != "group")
  {
    $trains_of_interest_this_user = array();
    foreach ($user->trains_of_interest->value as $train)
    {$trains_of_interest_this_user = array_merge($trains_of_interest_this_user, array((string)$train));};
    $trains_of_interest = array_merge($trains_of_interest, array((string)$user->handle => $trains_of_interest_this_user));
  }
}
/* finished getting user variables */

/* check if desired day and time */
$current_day=date("D");
$current_time=date("H:i");
if(($days_to_check_status["group"][$current_day]) and
   ($current_time>=$time_of_day_to_start_checking_status["group"]) and
   ($current_time<=$time_of_day_to_stop_checking_status["group"]) and
   ($user_service_active["group"])
  )
{
  /* yes, so get program status variables from XML if it exists */
  if (file_exists($program_status_file))
  {
    $xml = simplexml_load_file($program_status_file);
    /* last time program status file updated */
    $last_time_program_status_file_updated = unserialize($xml->last_time_program_status_file_updated);
    /* previous train status for each user  */
    $train_stati=unserialize($xml->train_stati);
  }
  else
  {
    /* XML program variables file does not exist, initialize default values */
```

```
  /* last time program status file updated */
  $last_time_program_status_file_updated = "";
  /* previous train status for each user  */
  foreach ($user_list as $user)
  {
    foreach ($trains_of_interest[$user] as $train)
    {$train_stati[$user][$train]="";};
  }
}
/* if new date, then note it */
$current_date=date("Ymd");
$first_marc_query_for_the_day=($current_date != $last_time_program_status_file_updated);
/* for each train of interest, clear old status if needed */
if ($first_marc_query_for_the_day)
{
  foreach ($user_list as $user)
  {
    foreach ($trains_of_interest[$user] as $train)
    {$train_stati[$user][$train]="0 min";};
  }
}
/* get text from marc web site and clean up */
do
{
  $marc_location_txt=file_get_contents($marc_location_url);
}while ($marc_location_txt=="");
$marc_location_txt=clean_page($marc_location_txt);
$start=strpos($marc_location_txt,'<map name="testmap">')+20;
if ($start<=20) {$start=strpos($marc_location_txt,'<map name=testmap>')+18;}
$stop=strpos($marc_location_txt,'</map>',$start);
$marc_location_txt=substr($marc_location_txt,$start,$stop-$start);
$marc_location_txt=clean('shape="rect" ',",$marc_location_txt);
/* get and print data for each train, one line at a time */
if ($print_table)
```

```
{
  print "TOI?"."\t";
  print "COI?"."\t";
  print "train"."\t";
  print "x"."\t";
  print "y"."\t";
  print "delay"."\t";
  print "updated"."\n";
}
foreach ($user_list as $user) {$cellphone_message[$user]="";}
$marc_location_array=explode('>',$marc_location_txt);
foreach ($marc_location_array as $marc_location_one_train)
{
  /* make sure there is data in this line */
  $marc_location_one_train=trim($marc_location_one_train);
  if ($marc_location_one_train)
  {
    /* find positions of handle characters in this line */
    $quote_1=strpos($marc_location_one_train,""                    );
    $comma_1=strpos($marc_location_one_train,','        ,$quote_1+1);
    $comma_2=strpos($marc_location_one_train,','        ,$comma_1+1);
    $comma_3=strpos($marc_location_one_train,','        ,$comma_2+1);
    $quote_2=strpos($marc_location_one_train,""        ,$comma_3+1);
    $pound_1=strpos($marc_location_one_train,'#'        ,$quote_2+1);
    $delay_1=strpos($marc_location_one_train,'Delay: '  ,$pound_1+1);
    $updat_1=strpos($marc_location_one_train,'Updated: ',$pound_1+1);
    /* determine values for this train */
    $x=strval
      ((
      floatval(substr($marc_location_one_train,$quote_1+1,$comma_1-$quote_1-1))+
      floatval(substr($marc_location_one_train,$comma_2+1,$comma_3-$comma_2-1))
      )/2);
    $y=strval
      ((
```

```
    floatval(substr($marc_location_one_train,$comma_1+1,$comma_2-$comma_1-1))+
    floatval(substr($marc_location_one_train,$comma_3+1,$quote_2-$comma_3-1))
  )/2);
$train=trim(substr($marc_location_one_train,$pound_1+1,$delay_1-$pound_1-1));
$delay=trim(substr($marc_location_one_train,$delay_1+6,$updat_1-$delay_1-6));
$updated=trim(substr($marc_location_one_train,$updat_1+8,strlen($marc_location_one_train)-$updat_1-9));
/* if this train is important to a user and there has been a change, add train to users cell phone message */
$users_where_is_a_train_of_interest=0;
$users_where_is_a_change_of_interest=0;
foreach ($user_list as $user)
{
  if( ($days_to_check_status[$user][$current_day]) and
      ($current_time>=$time_of_day_to_start_checking_status[$user]) and
      ($current_time<=$time_of_day_to_stop_checking_status[$user]) and
      ($user_service_active[$user]) and
      (in_array($train,$trains_of_interest[$user]))
    )
  {
    $users_where_is_a_train_of_interest=$users_where_is_a_train_of_interest+1;
    if ($train_stati[$user][$train] !== $delay)
    {
      $users_where_is_a_change_of_interest=$users_where_is_a_change_of_interest+1;
      $cellphone_message[$user]=$cellphone_message[$user]."[".$train.":".$delay."]";
    }
    $train_stati[$user][$train]=$delay;
  }
}
/* print values for this train */
if ($print_table)
{
  print $users_where_is_a_train_of_interest."\t";
  print $users_where_is_a_change_of_interest."\t";
  print $train."\t";
  print $x."\t";
```

```
      print $y."\t";
      print $delay."\t";
      print $updated."\n";
    }
  }
}
/* if cellphone message is not null, output */
foreach ($user_list as $user)
{
  if ($cellphone_message[$user]<>"")
  {
    if ($print_cell_message) {print "Message to ".$user.": ".$cellphone_message[$user]."\n";}
    if ($user_contact_via_email[$user])
    {
      if (mail($user_address[$user], "marc update", $cellphone_message[$user],  "From: train\r\n"))
      {$text_response="successful";}
      else
      {$text_response="failed";}
    }
    else
    {
      $text_response=do_post_request
      (
        'http://www.txtdrop.com/send.php',
        'number='.$user_address[$user].'&body='.$cellphone_message[$user]
      );
    }
    if ($print_cell_message) {print "\t".$text_response."\n";}
  }
}
/* save variables to file */
/* Could, but did not, use XMLWrite object */
$file_handle=fopen($program_status_file, 'w');
  fwrite($file_handle, '<?xml version="1.0" encoding="utf-8" standalone="yes"?>'."\n");
```

```
    fwrite($file_handle, "<variables>\n");
      fwrite($file_handle, "<last_time_program_status_file_updated>");
        fwrite($file_handle, serialize($current_date));
      fwrite($file_handle, "</last_time_program_status_file_updated>\n");
      fwrite($file_handle, "<train_stati>");
        fwrite($file_handle, serialize($train_stati));
      fwrite($file_handle, "</train_stati>\n");
    fwrite($file_handle, "</variables>\n");
  fclose($file_handle);
 }

 /* exit */
?>
```

Appendix B: Script and User Preference XML File

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE settings [
  <!ELEMENT settings (
    timezone,
    print_table,
    print_cell_message,
    program_status_file,
    marc_location_url,
    user_list
  )>
  <!ELEMENT timezone (#PCDATA)>
  <!ELEMENT print_table (#PCDATA)>
  <!ELEMENT print_cell_message (#PCDATA)>
  <!ELEMENT program_status_file (#PCDATA)>
  <!ELEMENT marc_location_url (#PCDATA)>
  <!ELEMENT user_list (user+)>
  <!ELEMENT user (
    handle,
    user_service_active,
    days_to_check_status+,
    time_of_day_to_start_checking_status,
    time_of_day_to_stop_checking_status,
    user_contact_via_email,
    user_address,
    trains_of_interest
  )>
  <!ELEMENT handle (#PCDATA)>
  <!ELEMENT user_service_active (#PCDATA)>
  <!ELEMENT days_to_check_status (handle,value)>
  <!ELEMENT time_of_day_to_start_checking_status (#PCDATA)>
  <!ELEMENT time_of_day_to_stop_checking_status (#PCDATA)>
```

```
  <!ELEMENT user_contact_via_email (#PCDATA)>
  <!ELEMENT user_address (#PCDATA)>
  <!ELEMENT trains_of_interest (value+)>
  <!ELEMENT value (#PCDATA)>
]>
<settings>
  <timezone>America/New_York</timezone>
  <print_table>true</print_table>
  <print_cell_message>true</print_cell_message>
  <program_status_file>textmarc3_status.xml</program_status_file>
  <marc_location_url>http://www.marctracker.com/PublicView/location.html</marc_location_url>
  <user_list>
    <user>
      <handle>group</handle>
      <user_service_active>true</user_service_active>
      <days_to_check_status>
        <handle>Mon</handle>
        <value>true</value>
      </days_to_check_status>
      <days_to_check_status>
        <handle>Tue</handle>
        <value>true</value>
      </days_to_check_status>
      <days_to_check_status>
        <handle>Wed</handle>
        <value>true</value>
      </days_to_check_status>
      <days_to_check_status>
        <handle>Thu</handle>
        <value>true</value>
      </days_to_check_status>
      <days_to_check_status>
        <handle>Fri</handle>
        <value>true</value>
```

```
  </days_to_check_status>
  <days_to_check_status>
    <handle>Sat</handle>
    <value>false</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Sun</handle>
    <value>false</value>
  </days_to_check_status>
  <time_of_day_to_start_checking_status>06:30</time_of_day_to_start_checking_status>
  <time_of_day_to_stop_checking_status>20:00</time_of_day_to_stop_checking_status>
  <user_contact_via_email></user_contact_via_email>
  <user_address></user_address>
  <trains_of_interest></trains_of_interest>
</user>
<user>
  <handle>eric cell</handle>
  <user_service_active>true</user_service_active>
  <days_to_check_status>
    <handle>Mon</handle>
    <value>true</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Tue</handle>
    <value>true</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Wed</handle>
    <value>true</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Thu</handle>
    <value>true</value>
  </days_to_check_status>
```

```
  <days_to_check_status>
    <handle>Fri</handle>
    <value>true</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Sat</handle>
    <value>false</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Sun</handle>
    <value>false</value>
  </days_to_check_status>
  <time_of_day_to_start_checking_status>06:30</time_of_day_to_start_checking_status>
  <time_of_day_to_stop_checking_status>20:00</time_of_day_to_stop_checking_status>
  <user_contact_via_email>false</user_contact_via_email>
  <user_address>7l2s2nm614</user_address>
  <trains_of_interest>
    <value>P880</value>
    <value>P871</value>
    <value>P881</value>
  </trains_of_interest>
</user>
<user>
  <handle>eric work</handle>
  <user_service_active>true</user_service_active>
  <days_to_check_status>
    <handle>Mon</handle>
    <value>true</value>
  </days_to_check_status>
  <days_to_check_status>
    <handle>Tue</handle>
    <value>true</value>
  </days_to_check_status>
  <days_to_check_status>
```

```
      <handle>Wed</handle>
      <value>true</value>
    </days_to_check_status>
    <days_to_check_status>
      <handle>Thu</handle>
      <value>true</value>
    </days_to_check_status>
    <days_to_check_status>
      <handle>Fri</handle>
      <value>true</value>
    </days_to_check_status>
    <days_to_check_status>
      <handle>Sat</handle>
      <value>false</value>
    </days_to_check_status>
    <days_to_check_status>
      <handle>Sun</handle>
      <value>false</value>
    </days_to_check_status>
    <time_of_day_to_start_checking_status>18:30</time_of_day_to_start_checking_status>
    <time_of_day_to_stop_checking_status>20:00</time_of_day_to_stop_checking_status>
    <user_contact_via_email>true</user_contact_via_email>
    <user_address>e@email.com</user_address>
    <trains_of_interest>
      <value>P879</value>
      <value>P881</value>
    </trains_of_interest>
  </user>
 </user_list>
</settings>
```

Appendix C: Previous Train Statuses XML File

<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<variables>
<last_time_program_status_file_updated>s:8:"20070723";</last_time_program_status_file_updated>
<train_stati>a:2:{s:9:"eric cell";a:3:{s:4:"P880";s:0:"";s:4:"P871";s:0:"";s:4:"P881";s:5:"0 min";}s:9:"eric work";a:2:{s:4:"P879";s:5:"0 min";s:4:"P881";s:5:"0 min";}}</train_stati>
</variables>