A GIS Enabled Air Dispersion Modeling Tool for Emergency Management

Stephen D. Jakala
Department of Resource Analysis, Saint Mary's University of Minnesota, Minneapolis, MN, 55404

Keywords: GIS, Air Dispersion Modeling, Atmospheric Dispersion Modeling, Chemical Plume Analysis

Abstract

This paper documents the importance of GIS enabled air dispersion modeling for use in Emergency Management operations and outlines the steps taken to design and build a GIS enabled air dispersion modeling tool for ESRI's ArcGIS software. The tool contains report generating functionality that has the ability to analyze the area affected by the plume and create a summary report on the people and resources that are in harms way. The paper also provides a sample case study on the analysis of an accidental chemical release scenario.

Introduction

Air dispersion models are computer tools that use mathematical equations to simulate how air pollutants disperse in the atmosphere. Air dispersion models are used to estimate or to predict the downwind concentration of chemical air pollutants from sources such as industrial plants, vehicular traffic, chemical storage facilities, and accidental chemical spills.

Air dispersion modeling has evolved greatly since the initial model development in the early twentieth century. At that time, air emissions from industrial and mobile sources were substantially unregulated and the dispersion process was not well understood (Westbrook, 1999). By the 1950's, scientists were examining the dispersion process to predict atomic bomb fall out. In the 1960's and 1970's, two pioneering scientists, F. Pasquill and F.A. Gifford, developed basic dispersion curves that could be employed in modeling. Another scientist, G.A. Briggs developed equations to describe emissions plume behavior known as "the Briggs Equations" which were widely used and helped advance scientific research in the area of air dispersion modeling. In the past, computer resources were limited and dispersion modeling calculations were often completed manually or by using relatively crude computer code (Westbrook, 1999).

Improvements in technology have greatly increased the speed and accuracy of today's air dispersion models and they have found use in many different areas from ensuring regulatory compliance under the Clean Air Act to assessing human exposure to natural, accidental, and intentional chemical releases. It is the latter of the two examples that has made air dispersion modeling such an important tool for emergency managers everywhere.

Emergency management responsibilities are centered on assessing, analyzing, managing, and protecting the public through emergency planning and coordination of emergency services operations in major emergencies and disasters such as chemical release scenarios (Bacon, 2000). Recently, emergency managers have realized the benefit of leveraging air dispersion modeling with the modeling ability, analysis, and display functionalities of Geographic Information Systems (GIS). Because of the intrinsic spatial components of air dispersion modeling, it was inevitable that these two technologies would be used in conjunction with each other to give an overall situational awareness previously unattainable in either standalone system. In fact, combining these two software platforms into an easy to use interface was the inspiration for the tool and analysis created in the course of this research project.

Using GIS enabled air dispersion modeling, emergency managers are able to display a chemical plume footprint, which represents an overhead view of the chemical plume and is typically symbolized by a polygon or group of polygons representing different exposure rates at different areas within the plume, on top of other relevant GIS data. Types of analysis that can be performed using this methodology include demographic, transportation, and critical infrastructure analysis. This data can be used to generate information such as how many people are affected by the chemical release or which roads need to be closed to isolate the affected area or which businesses need to be notified about the situation.

GIS enabled air dispersion modeling provides chemical release information in a convenient and easily understandable fashion, which can be quickly distributed to necessary parties involved in critical emergency service operations (Hunt, 2005).

The purpose of this paper was to research and document the benefits of GIS enabled air dispersion modeling tools for emergency managers, document the steps taken to create a GIS enabled air dispersion modeling tool for the Scott County, Minnesota Emergency Management Department, and to use the tool to analyze a hypothetical chemical release situation and its impact on county peoples, transportation, and other critical infrastructure elements.

Background

There are many different types of potential chemical release scenarios that can occur, and emissions can originate from various source types that have different levels of urgency but these can all be broken down to three major modeling activities. These three major modeling activities are (1) contingency modeling, (2) short term site assessment modeling, and (3) natural, accidental, or intentional release modeling (Turpin, 2004).

Contingency modeling is a planning activity that is used to provide possible downwind concentrations for specific chemicals and emission rates, which may be encountered at a potential chemical release site. An example of this type of modeling would be to model the most likely outcome of a chemical release using an average of historical weather data for the study area. This type of modeling is commonly used for training scenarios and for developing emergency response plans.

Short-term site assessment modeling is used to calculate chemical concentrations, which have occurred over periods of a year or less. This type of model is most often used for risk assessments after a release has occurred. An example of this is type of modeling would be to model a release that occurred two weeks prior to determine the area that was affected by the plume to be able to choose the best locations for soil sampling.

The last type of modeling and probably the most critical to emergency managers is natural, accidental, or intentional release modeling. Regardless of the motive, this type of release modeling is performed soon after a chemical release occurs or is discovered and is intended to provide immediate results. These types of models perform the best when real-time data is available to be able to model current ground conditions. These models provide worst-case scenario results and supply data for emergency managers to take immediate action. An example of this type of modeling scenario would be if a chemical storage tank ruptured and a large amount of toxic chemical was released in to the atmosphere. The data from the accidental release model could be used to notify or evacuate people in the affected area (Turpin, 2004). This is also the modeling scenario that was used for a case study in the course of this research paper.

Due to the diverse factors that can affect plume behavior, many different air dispersion models have been devised over the years. Special models exist to address plume movement in valleys, around mountains, over water, and near shorelines. Other models address short-term impacts for chemicals in different physical states. These models are used to complete dispersion modeling for accidental chemical spills. Separate models that calculate chemical concentrations due to sources emitting into the lee side of structures also exist (Westbrook, 1999).

The air dispersion model that was chosen for this project is one of the more common air dispersion models used in emergency management. It is a free software tool created by the United States Environmental Protection Agency (EPA) called the ALOHA air model (Areal Locations of Hazardous Atmospheres). ALOHA contains a database of approximately 1,000 common chemicals and it can predict the atmospheric dispersion rate and direction of chemical releases from broken pipes, ruptured tanks, puddles or direct chemical sources (Tomaszewski, 2003). The model estimates pollutant concentrations downwind from the source of a spill, taking into consideration the toxological and physical characteristics of the spilled material. ALOHA uses two separate dispersion models, a Gaussian model and a heavy gas model. The Gaussian model describes movement and spread of a neutrally buoyant gas, which is approximately the same density as air. The heavy gas dispersion calculations are derived from the Dense Gas Dispersion Model (DEGADIS) model, which was developed in part by the U.S. EPA (Chakraborty and Armstrong, 1994).

The main reason the ALOHA model was selected for this project is because it is one of the most widely accepted models available and its open architecture allows integration with other platforms such as ESRI’s ArcGIS technology. The EPA even offers a suite of free import tools that enable importation of ALOHA plume footprints directly into ESRI's ArcMap GIS software (Chakraborty and Armstrong, 1994). Once the plume footprint is imported into the GIS software all of the analysis capability of the GIS software is available to perform on the plume. The information that is obtained from this analysis is important for emergency managers, but because of the general lack of GIS training for emergency management personnel, this level of implementation is hard to realize, especially in time critical situations. Thus, some GIS departments have created custom applications that automate this process.

There is also a small market of third party vendors that have created GIS software extensions and standalone software that automate the process of importing plumes from various air dispersion models. Many of these applications provide the added benefit of integrating real-time weather data into the models equations and also conduct statistical analysis on the area affected by the plume and report this information in easy to understand printable reports.

Methods

Tool Creation

The ALOHA air dispersion model was selected as the preferred modeling tool to use for this project because of its open architecture and easy integration with ESRI's ArcGIS software. The ALOHA software is a standalone application that allows a user to enter specific variables for a chemical release scenario, the tool then models how the toxic gas cloud will disperse throughout the atmosphere.

The GIS enabled air dispersion modeling tool's requirements are as follows. First, the tool needs to be able to be integrated with ESRI's ArcGIS software. This means that the user needs to be able to click a point on a map within ArcMap to define the location of the release point. The tool needs to be able to determine the real-time weather conditions, such as wind speed and direction, at the point of origin. This information will also be needed later as an input for ALOHA's calculations. After the user defines the point of origin in the map, the user then needs to define ALOHA's required variables such as chemical type and chemical source. This information along with the real-time weather data is transferred to the ALOHA program for analysis. After the analysis is complete the result is a plume footprint file which can be displayed in ALOHA's primitive map interface. This plume footprint then needs to be converted into an ESRI shapefile data format and imported into ArcGIS.

Once the data is imported into the GIS software, analysis can be conducted. The tool then needs to perform overlay analysis to select data from the base map information such as points of critical infrastructure, population and demographic data, and anything else deemed useful. This data needs to be presented in an easy to create summary document, which can be easily distributed during a chemical release event.

This tool will streamline the process of analyzing an ALOHA plume footprint in ArcGIS. The reporting feature will allow Emergency Managers to share information with others in a fast and convenient fashion.

Converting ALOHA's .PAS file into a Shapefile

The first and most important part of creating this tool was determining how to convert the plume footprint file created by ALOHA into a shapefile. Figure 1 displays an example of a typical ALOHA footprint. As shown in Figure 1, the legend has different values that represent chemical concentration at different areas of the plume. The area immediately adjacent and down wind from the point of release has the highest chemical concentration.

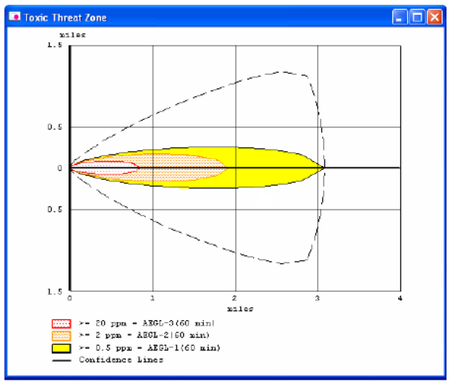


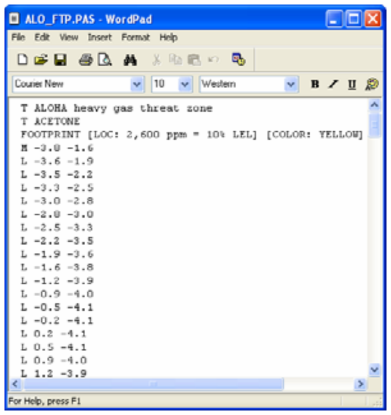
Figure 1. Typical ALOHA footprint as displayed in ALOHA.

The footprint file is always saved as ALO_FTP.PAS in the ALOHA file directory. This file is a temporary file that is created only after ALOHA has performed its analysis and while the ALOHA program is open. As soon as the program is closed, the ALO_FTP.PAS file is deleted.

The .PAS file is a text file that defines the attributes and X, Y locations of each of the vertices that make up the polygons of a plume footprint (Figure 2). Each line in the .PAS file is preceded by a character that defines the type of data that will follow. The two main types of data that are represented in the .PAS file are X, Y locations and attribute data. For example, the character "L" would precede an X, Y location, and the word "FOOTPRINT" would precede attribute information for a specific polygon.

In order to create a shapefile from the data contained in the .PAS file, the first step was to parse the data to extract the critical elements. The tool reads the .PAS file line by line and looks for the all the lines that start with an "M" which are the starting and ending points of polygons, or an "L" which are the other vertices that make up a polygon. The tool also evaluates other indications in the data that designate where the attribute information is located. Figure 3 presents the code for this extraction.

As illustrated in Figure 3, the X, Y data in the .PAS file has values such as 0.5, -4.1. It is easy to see that the plume is not going to be placed in the correct location because the user-defined point of origin has not been taken into consideration. In order to ensure that the plume plots in the correct location, the



```
T ALOHA heavy gas threat zone
T ACETONE
FOOTPRINT [LOC: 2,600 ppm = 10% LEL] [COLOR: YELLOW]
M -3.8 -1.6
L -3.6 -1.9
L -3.5 -2.2
L -3.3 -2.5
L -3.0 -2.8
L -2.8 -3.0
L -2.5 -3.3
L -2.2 -3.5
L -1.9 -3.6
L -1.6 -3.8
L -1.2 -3.9
L -0.9 -4.0
L -0.5 -4.1
L -0.2 -4.1
L 0.2 -4.1
L 0.5 -4.1
L 0.9 -4.0
L 1.2 -3.9
```

Figure 2. Contents of a typical .PAS file.

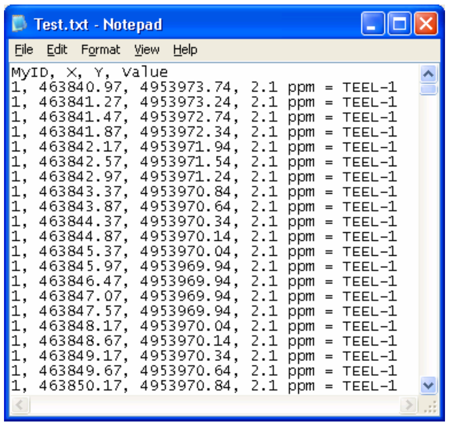
application needs to add the X, Y location of the user-defined point of origin to each X, Y value. This information is collected when the user clicks on the map after selecting the tool button. For example, after the user-defined point of origin was added to each X, Y value in the .PAS file, the X,

```
FF = FreeFile
Open "C:\ALOHA\ALO_FTP.PAS" For Input As #FF                    '** Opens up the .PAS file for use as the FreeFile feed

Dim fs As Object
Set fs = CreateObject("Scripting.FileSystemObject")
Set a = fs.CreateTextFile("C:\ALOHA\Test.txt", True)
a.WriteLine ("MyID, X, Y, Value")                               '** Creates a new text file called Test.txt and adds MyID, X, Y, Value to the
                                                                "** first line. These will be the field headings in the new table.
Dim i As Integer
i = 0

Dim f As String
Dim d As String
Dim g As String

Do While Not EOF(FF)                                            "** Do while its not the end of the file

  Line Input #FF, LineFromFile

       f = Split(LineFromFile, " ")(0)

       If f = "FOOTPRINT" Then                                  "** Parse the .PAS file to look for the word FOOTPRINT. This is the Value.
         d = Split(LineFromFile, ":")(1)
         g = Split(d, "]")(0)
       End If

       If Mid(LineFromFile, 1, 2) = "M " Then                   "** Parse the .PAS file to look for lines in the .PAS file that start with M.
                                                                '** An M indicates the start of a polygon.

            i = i + 1                                           '**Becomes the MyID value. All vertices of a polygon have the same ID.
                                                                '** This value changes once the code loops through to a new letter M.

           Dim pXY As String
           pXY = Mid(LineFromFile, 2)
           pXY = Trim(pXY)                                      '** Splits the concatenated X and Y values out of the string but are still
                                                                '** separated by a space.
           Dim pXY1 As String
           pXY1 = Split(pXY, " ")(0)                            '** Gets the X value from the variable pXY

           Dim pXY2 As String
           pXY2 = Split(pXY, " ")(1)                            '** Gets the Y value from the variable pXY

           a.WriteLine (i & ", " & pXY1 + pX & ", " _
            & pXY2 + pY & "," & g)                              '** Writes the MyID, X, Y, and Value data to the .txt. Also adds the lat/long
                                                                '**from the users map click.
       End If

       If Mid(LineFromFile, 1, 2) = "L " Then                   '**Parse the .PAS file to look for lines in the .PAS file that start with L. An L
                                                                '** indicates a vertex in a polygon.

            'Dim pXY As String                                  '** Splits the concatenated X and Y values out of the string but are still
            pXY = Mid(LineFromFile, 2)                          '** separated by a space.
            pXY = Trim(pXY)

           'Dim pXY1 As String
           pXY1 = Split(pXY, " ")(0)                            '** Gets the X value from the variable pXY

           'Dim pXY2 As String
            pXY2 = Split(pXY, " ")(1)                           '** Gets the Y value from the variable pXY

           a.WriteLine (i & ", " & pXY1 + pX & ", " _
            & pXY2 + pY & "," & g)                              '** Writes the MyID, X, Y, and Value data to the .txt. The actual lat/long taken
       End If                                                   '**from the map coordinates are added to the X, Y values in this step (pX and
                                                                '**pY).  This allows the plume to display in the location that the user clicked in
                                                                "**the map.
Loop
```

Figure 3. Code sample showing how the application parses the .PAS file to pull out an ID, X, Y, and Value and writes it to a text file which can then be used to build a table.

Y data would have values such as 462432.5, -4953664.1. Because the same numbers are added to each X, Y location, the shape of each polygon will be maintained thus ensuring that the polygon that is created appears in the correct location.

The goal of parsing the data is to create a new text file that contains the data needed to build a shapefile. It is important to determine where a polygon starts and stops in order to create a shapefile that has complete polygons and correct attributes along with the correct set of X, Y locations. Luckily, the .PAS file signifies when a polygon starts and stops by preceding the data with the character “M.” Figure 4 provides an example of a parsed .PAS file. It is a coma delimited text file with a feature ID, an X field, a Y field, and an attribute value field.

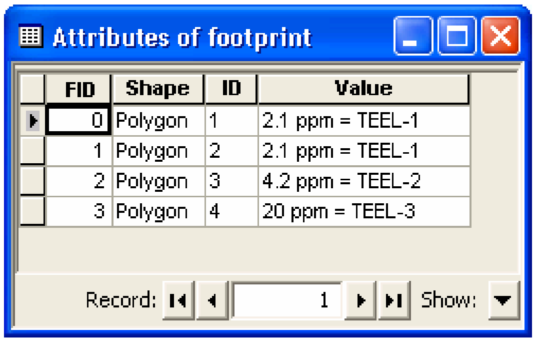


```
MyID, X, Y, Value
1, 463840.97, 4953973.74, 2.1 ppm = TEEL-1
1, 463841.27, 4953973.24, 2.1 ppm = TEEL-1
1, 463841.47, 4953972.74, 2.1 ppm = TEEL-1
1, 463841.87, 4953972.34, 2.1 ppm = TEEL-1
1, 463842.17, 4953971.94, 2.1 ppm = TEEL-1
1, 463842.57, 4953971.54, 2.1 ppm = TEEL-1
1, 463842.97, 4953971.24, 2.1 ppm = TEEL-1
1, 463843.37, 4953970.84, 2.1 ppm = TEEL-1
1, 463843.87, 4953970.64, 2.1 ppm = TEEL-1
1, 463844.37, 4953970.34, 2.1 ppm = TEEL-1
1, 463844.87, 4953970.14, 2.1 ppm = TEEL-1
1, 463845.37, 4953970.04, 2.1 ppm = TEEL-1
1, 463845.97, 4953969.94, 2.1 ppm = TEEL-1
1, 463846.47, 4953969.94, 2.1 ppm = TEEL-1
1, 463847.07, 4953969.94, 2.1 ppm = TEEL-1
1, 463847.57, 4953969.94, 2.1 ppm = TEEL-1
1, 463848.17, 4953970.04, 2.1 ppm = TEEL-1
1, 463848.67, 4953970.14, 2.1 ppm = TEEL-1
1, 463849.17, 4953970.34, 2.1 ppm = TEEL-1
1, 463849.67, 4953970.64, 2.1 ppm = TEEL-1
1, 463850.17, 4953970.84, 2.1 ppm = TEEL-1
```

Figure 4. Text file created from a parsed .PAS file.

The next step is to use ArcObjects code to create an empty polygon shapefile and add it to the ArcGIS project. Next a table is built from the text file that was created in the previous process (Figure 4). The application loops through the table adding each X, Y point to the empty polygon. After all of the points are added, ArcObjects code is used to select all of the points of each feature and create polygons from the X, Y points. A detailed description of the code used in this process is provided in Figure 6.

The end result of this process is a shapefile with the appropriate attributes corresponding to the correct polygons (Figure 5). The last step involved in converting the .PAS file to a shapefile is adding the correct symbology. Using ArcObjects, a UniqueValueRenderer was set up to symbolize the footprint file similarly to how the footprint appeared in the ALOHA software including a transparency. The next task was to figure out how to get the real-time weather data and other attributes into ALOHA.



FID	Shape	ID	Value
0	Polygon	1	2.1 ppm = TEEL-1
1	Polygon	2	2.1 ppm = TEEL-1
2	Polygon	3	4.2 ppm = TEEL-2
3	Polygon	4	20 ppm = TEEL-3

Figure 5. Attribute table of footprint.

Real-Time Weather Data

One of the most important features of air dispersion modeling software for modeling accidental releases is the ability to obtain and use real-time weather data. This is an important feature to add to the project because it streamlines the modeling process and adds reliability to the results. For this project, there were two main options for obtaining real-time weather data. The first option was to use NOAA’s (National Oceanic and Atmospheric Administration’s) National Weather Service XML (Extensible Markup Language) Feeds of Current

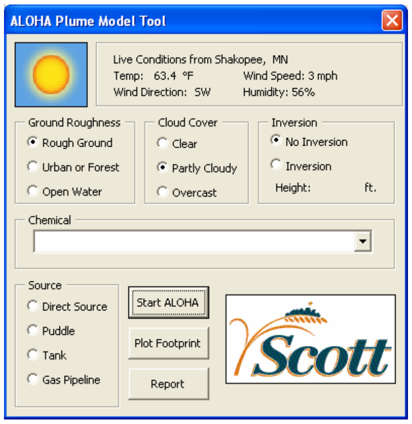
```
Call CreateShapefile("C:\ALOHA", "footprint")                    '** Calls a function that creates a shapefile with the correct format.

Call AddaShapefile("C:\ALOHA", "footprint", 0)                   '** Calls a function that adds the empty shapefile to the map so it
                                                                 '** can be built.
Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument

Dim pfLayer As IFeatureLayer
Set pfLayer = pMxDoc.FocusMap.Layer(0)                           '** Sets the focus to the new shapefile. It's the first one in the TOC.

Dim pTable As ITable
Set pTable = CreateTableFromText("C:\ALOHA", "Test.txt")         '** Calls a function that builds a table from the text file that was
If pTable Is Nothing Then                                        '** previously created.
   Exit Sub
End If

Dim pCursor As ICursor, pCursor2 As ICursor
Set pCursor = pTable.Search(Nothing, True)                       '** Creates a cursor that will loop through the new table.

Dim pDataStats As IDataStatistics
Set pDataStats = New DataStatistics
Set pDataStats.Cursor = pCursor
pDataStats.Field = "MyID" 'this is the unique id for each polygon   '** Sets the unique value of the table to the MyID field
Dim theValues As IEnumVariantSimple
Set theValues = pDataStats.UniqueValues


 Dim theID As Variant, pFilter As IQueryFilter, pRow As IRow, _  '** Sets up the new polygon and the tools needed to populate it.
pPolygon As IPointCollection
Dim pPoint As IPoint, pFeature As IFeature
Set pFilter = New QueryFilter
Dim theDamage As String
Dim pRecNum As String

theValues.Reset                                                  '** Sets the enumerator to the top of the table.
theID = theValues.Next                                           '** Gets the first MyID value in the table.
Do While theID <> ""
   pRecNum = theID
   Set pPolygon = New Polygon                                    '** Creates a new polygon.
   pFilter.WhereClause = "MyID = " & theID                       '** Selects all the MyID's with value 1 (then loops).
   Set pCursor2 = pTable.Search(pFilter, True)
   Set pRow = pCursor2.NextRow
   theDamage = pRow.Value(pRow.Fields.FindField("Value"))        '** Gets the Value attribute of the polygon.


    Do While Not pRow Is Nothing
      Set pPoint = New Point
      pPoint.PutCoords pRow.Value(pRow.Fields.FindField("X")),  _   '** Gets the X and Y attributes from the table for the record and
     pRow.Value(pRow.Fields.FindField("Y"))                      '** creates a point. This process is repeated for each MyID value.
      pPolygon.AddPoint pPoint
      Set pRow = pCursor2.NextRow
   Loop

   If pPolygon.PointCount >= 4 Then                              '** If a set of points has more then 4 points in it, then create a
                                                                 '** polygon from the points and assign it a ID and Value.
      Set pFeature = pfLayer.FeatureClass.CreateFeature
      Set pFeature.Shape = pPolygon
      pFeature.Value(pFeature.Fields.FindField("ID")) = theID
      pFeature.Value(pFeature.Fields.FindField("Value")) = theDamage
      pFeature.Store
   End If

   theID = theValues.Next
Loop                                                             '** Loop through and create another polygon. After all of the
                                                                 '** polygons are created, they are merged and symbolized.
```

Figure 6. This code sample shows how the application creates a table from the text file that was created in Code Sample 1 and builds a shapefile from it.

Weather Conditions. This option had a few shortfalls though. It was only updated hourly and the closest observation point to the study area of Scott County, MN was the Minneapolis/St. Paul International Airport. The other option was to use WeatherBug Lab's WeatherBug Application Developers Interface, which allows access to a queriable XML or RSS (Really Simple Syndication) stream that returns real-time weather conditions for any zip code in the United States. WeatherBug has access to over 8,000 remote weather stations throughout the United States and its database is updated every two seconds. WeatherBug has two weather stations within Scott County and many close by. As a result, the proximity of weather bug stations to the study area was the primary reason this option was selected to use.

In order to obtain the zip code for the point of origin, the application had to provide some search functionality to the user based on the user clicking the map. The first step in accomplishing this task was to add a zip code shapefile layer to the base map in the ArcGIS project. After selecting the GIS enabled air dispersion modeling tool button and clicking on the map, ArcObjects is used to perform a query on the zip code layer to return the zip code that was clicked. This variable can then be entered into an XML stream request that will be read by an XML stream reader implemented with Visual Basic. The XML stream is parsed much like the .PAS file to extract the needed information. This process includes extracting the wind speed, wind direction, temperature, humidity, the name of the location the conditions are coming from, and an image, which is a visual representation of the weather conditions. Figure 8 details how the XML stream reader processes this information and extracts the needed data.

This information is entered into ALOHA along with the other parameters needed for analysis. Before this can be conducted, it is first displayed to the user on a form that appears after the user clicks the map to define the point of origin. The user form shown in Figure 7 shows the weather data obtained for the zip code that was defined by the user. The form also displays other options that are available for the user to select to define parameters for the model such as the chemical to model, ground roughness, cloud cover, inversion information, and chemical source type. There are additional user forms for each chemical source that can be modeled. These additional forms appear when selected.



Live Conditions from Shakopee, MN
Temp: 63.4 °F   Wind Speed: 3 mph
Wind Direction: SW   Humidity: 56%

Ground Roughness: [✓ Rough Ground] Urban or Forest Open Water
Cloud Cover: Clear [✓ Partly Cloudy] Overcast
Inversion: [✓ No Inversion] Inversion Height: ft.
Chemical:
Source: Direct Source Puddle Tank Gas Pipeline

Figure 7. The plume model form displaying the weather data.

Passing Variables to ALOHA

After entering the required parameters on the ALOHA Plume Model form, the user can press the Start ALOHA button. This button passes all of the variables defined in the Plume Model form along with attributes defined on any of the source forms to ALOHA and runs the model.

```
Dim sXMLFile As String                                          '** This sets up the different variables needed for this operation.
Dim oXml As Object
Dim oError As Object
Dim oDocNode As Object
Dim oCommand As Object
Dim oCommandNode As Object
Dim oSub As Object                                              '** sXMLFile is the web address to the Weatherbug  XML feed.
Dim oParam As Object                                            '** The variable pZipCode is populated by a spatial query
Dim oNode As Object                                             '** on the Zip Code layer in the map.

sXMLFile = "http://a4780801886.api.wxbug.net/getLiveWeatherRSS.aspx?ACode=A4780801886&zipCode=" & pZipCode

Set oXml = CreateObject("MSXML2.DOMDocument")                   '** Creates an XML stream reader.
oXml.async = False

oXml.Load sXMLFile                                              '** Loads the Weatherbug XML feed to the XML stream reader.
Set oError = oXml.parseError
   ' Check if there was any error open/parsing the XML file
If oError.ErrorCode <> 0 Then
   MsgBox "Error parsing " & sXMLFile & vbCrLf & _
      "Error: " & oError.reason
   Exit Sub
Else
End If

 Set oDocNode = oXml.documentElement
 Dim fs As Object
 Set fs = CreateObject("Scripting.FileSystemObject")            '** Creates text file from the XML Stream.
 Set a = fs.CreateTextFile("C:\ALOHA\RSS2.txt", True)           '** Used mostly for error checking.
 a.WriteLine (oXml.xml)

Dim pURL As String
pTempString = oDocNode.SelectNodes("//item/description").Item(0).Text  '** Temp string to hold XML so it can be parsed.

pLocation = Split(oXml.xml, "<title>Compact Live ")(1)
pLocation = Split(pLocation, " - USA")(0)                       '** Parses XML at common break point to extract the location.

pTemp = Split(pTempString, "<b>Temperature:</b> ")(1)
pTemp = Split(pTemp, "&")(0)                                    '**Parses XML at common break point to extract temperature.

pWindSpeed = Split(pTempString, "<b>Wind Speed:</b> ")(1)
pWindSpeed = Split(pWindSpeed, " mph")(0)                       '**Parses XML at common break point to extract wind speed.

pWindDir = Split(pTempString, "mph ")(1)
pWindDir = Split(pWindDir, "&")(0)                              '**Parses XML at common break point to extract wind direction.

pHumidity = Split(pTempString, "<b>Humidity:</b> ")(1)
pHumidity = Split(pHumidity, " ")(0)                            '**Parses XML at common break point to extract humidity.

pURL = Split(pTempString, "http")(1)
pURL = Split(pURL, ".gif")(0)                                   '**Parses XML at common break point to extract image URL.

Me.lblLocation = "Live " & pLocation                            '** Assigns labels in the form to their appropriate variables.
Me.lblTemp = pTemp & " °F"
Me.lblSpeed2 = pWindSpeed & " mph"
Me.lblDirection = pWindDir
Me.lblHumidity = pHumidity & "%"

'Me.Image1.Picture = LoadURLPicture("http" & pURL & ".gif")     '** Loads the image into the picture box on the form.

Set oXml = Nothing                                              '** Clears all of the variables of the stream reader.

Set oDocNode = Nothing
Set oCommand = Nothing
Set oCommandNode = Nothing
Set oError = Nothing
```

Figure 8. This code sample shows how the application downloads the real-time weather feed from an XML stream and parses out the useful information.

ALOHA was built to be able to handle this type of automation and includes a DLL that is supposed to allow inter-application communication. This functionality was not successfully utilized for this project due to lack of documentation on the subject. However, a workaround was identified and implemented.

ALOHA is fully navigable using keyboard shortcuts, tabs, and arrows. This means that even though the tool cannot communicate with ALOHA through inter-application communication, it can navigate the user forms and populate the variables in ALOHA automatically by sending the appropriate keystrokes to the program.

After pressing the Start ALOHA button, the ALOHA application starts and the focus is set using a Shell command. Then, through a series of “if then” statements, the tool sends the ALOHA application the appropriate keystrokes using the Visual Basic SendKeys statement. A small portion of the code used to perform these tasks is provided in Figure 9. This process takes a few seconds as it goes through the appropriate dialogs entering the weather data, chemical information, and chemical source data until it stops at the plume generation portion of the application. This is as far as the automation process goes; it leaves the user the option of modeling several different types of hazards, a toxic area of vapor cloud, a flammable area of vapor cloud, or a blast area of vapor explosion.

At this point in the application, the user chooses the type of hazard they would like to model and presses OK. The user switches the focus back to the ArcMap project to import the footprint and perform the analysis. It is at this point the previously discussed process of converting the .PAS footprint file to a shapefile, which has been automated, is executed by pressing the Plot Footprint button on the user form.

Obtain Basemap Data

In order to conduct the overlay analysis to determine which features are affected by the plume footprint, an underlying basemap needs to be added to the map. The data layers that were obtained for this project besides the zip code layer that was mentioned previously are as follows: parcel polygons, critical infrastructure points, census block polygons, and roads. This data was chosen because its components make up a solid basemap, which can support the analysis required for this project.

All data was obtained from the Scott County GIS department except the census block polygons, which were obtained from ESRI’s ArcData data download site - http://arcdata.esri.com /data/tiger2000/tiger_download.cfm.

Plume Footprint Analysis and Summary Report

One of the major benefits of this tool is its ability to quickly and easily create a customized summary report. The tool has the ability to select the features that are affected by, or fall within, the plume path. A report can then be created that enters the information about the selected features into a nicely formatted report. ArcGIS has two basic options for creating reports, the default ArcMap report wizard and Crystal Reports. Crystal Reports was chosen for this tool because of its superior functionality. A report template was created in Crystal Reports, which could be reused with

```
            '********************************This is a small excerpt of the code *********************************

Shell "C:\ALOHA\ALOHA.EXE", vbNormalFocus     '** This Shell function opens ALOHA and sets the focus.

Call Wait(5)                                  '** Calls the Wait function which pauses the code for 5 seconds.
                                              '** This is used a lot in this part of the project to ensure commands
                                              '** issued at the correct time. In this case it waits for ALOHA to open.

SendKeys "~"                                  '** SendKeys sends shortcut keys to ALOHA. This sends the ENTER key.

'Set Atmosphere Data
SendKeys "^A"                                 '** Opens the Atmospheric Data dialog.

If pWindSpeed <= 2 Then                       '** Checks to see if the windspeed is > then 2 (ALOHA requirement).
   SendKeys "2"                               '** If <= 2, then send a value of 2.
Else
   SendKeys pWindSpeed                        '** If >= 2, then send the windspeed value.
End If

SendKeys "{TAB}"                              '** Send the TAB key. The next 3 keys navigate to the next area of the form.

SendKeys "{RIGHT}"                            '** Send the RIGHT arrow.

SendKeys "{TAB}"                              '** Send the TAB key.

SendKeys "{TAB}"                              '** Send the TAB key.

SendKeys pWindDir                             '** Enter the wind direction into the proper area on the form.
```

Figure 9. This code sample illustrates how the application sends data to ALOHA using the SendKeys function of Visual Basic.

new data anytime the tool is run. The report consists of five sections.

(1) A population information section with population data for the census blocks that intersect with the plume footprint.

(2) A parcel information section with data on the parcel count and acreage total for parcels that intersect with the plume.

(3) A road section with the number of road segments that are within the plume footprint.

(4) A housing information section with data on the number of households, families, household units, vacant units, owned units, and rented units.

(5) Critical infrastructure section with the names, addresses, and a brief description of critical infrastructure points that are with in the plume footprint.

The data for all of these sections are mapped to the tables created in the feature selection process, which will be described in the next section. This data is updated whenever the tool is run and the new data is reflected in the report.

After the feature selections are performed and the data tables are created, a Crystal Reports report viewer is launched which contains the report template created in the previous step. After pressing the refresh button on the report viewer, the updated data populates the report and the plume hazard summary is ready to print or save as a pdf.

Plume Analysis

After the plume is imported into the map (Figure 10), the user can start the analysis by pressing the Report button on the user form shown in Figure 7. As outlined in Figures 13 and 14, several processes must occur before the data is ready to be added to the report. On line 1, the first action that takes place is to zoom to the extent of the new plume layer. This is necessary to create an image of the map so it can be placed in the final report. Line 2 calls a function to export the data view to a JPEG and saves it in a specified directory that can be accessed by the report template. The third action that takes place is the action most important to the analysis portion of this project. Line 3 of Figure 13 calls a function called SelectbyPlume, which performs an overlay analysis on the layers affected by the plume. This function selects the data layers and exports their selected records to dbf tables that are consumed by the report template. The major processes in the SelectbyPlume function are outlined in the second half of Figure 13 and Figure 14.

The first step in the SelectbyPlume function is to programmatically set the footprint layer as the only selectable layer in the map. The program then selects the footprint layer in lines 23 and 24 so it can be made into a graphic. The select by graphic method was chosen over the select by feature method because of its ease of implementation. A graphic element is then built from each of the selected features in the footprint layer in the " Do While Loop" located at lines 34 – 48.

After a graphic is created for each part of the feature, the application needs to select the features of the parcels, census blocks, points of critical infrastructure, and roads that intersect with the plume footprint graphic. The next step in this process is to set the previously mentioned layers as the only selectable layers in the map. This is executed in lines 49 – 54 of Figure 14. The next step is to select all of the selectable layers that intersect the footprint graphic. In order to streamline the code, the application uses the built in ArcMap command Query_SelectByGraphics, which will select all of the selectable layers by the graphics in the graphics container. Once the selection has taken place, another function named ExportDBF is called on each of the four layers needed for the report. This function takes the selected features of each layer and builds a dbf table from them. The next step is to clear the selected features using an ArcMap command called Query_ClearSelection in lines 67 - 69. The remaining process clears the graphics container and refreshes the map display in lines 70 - 78.

At this point in the analysis, the application has selected all of the features of the four layers of interest that intersect the plume footprint and exported their records to four separate dbf tables. The application has also created a JPEG image of the map display for use in the final report. The final step is to open the pre-configured report template inside of a Crystal Reports report viewer with lines 9 – 12. An overview of the steps taken to create the report is shown in Figure 12.

The final report appears to the user in a separate window shown in Figure 11. The Crystal Reports report viewer allows the user to zoom in and out, pan, and view the report. The user also has the option of saving the report to a pdf or printing the report.

Case Study: An Accidental Chemical Release Scenario in Scott County, MN.

Imagine a situation similar to what occurred after a Canadian Pacific Railway train derailed in Minot, North

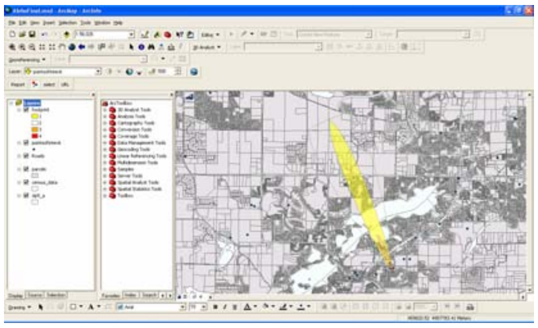


Figure 10. Screenshot of plume footprint in ArcMap.

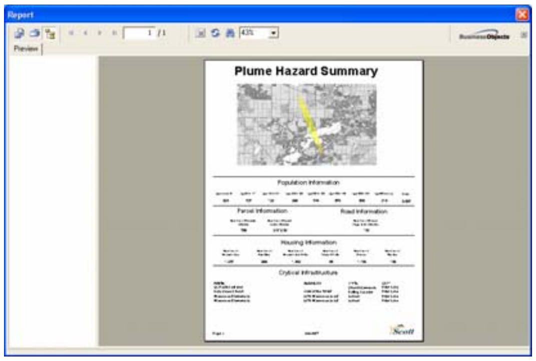


Figure 11. Report viewer showing Plume Hazard Summary.

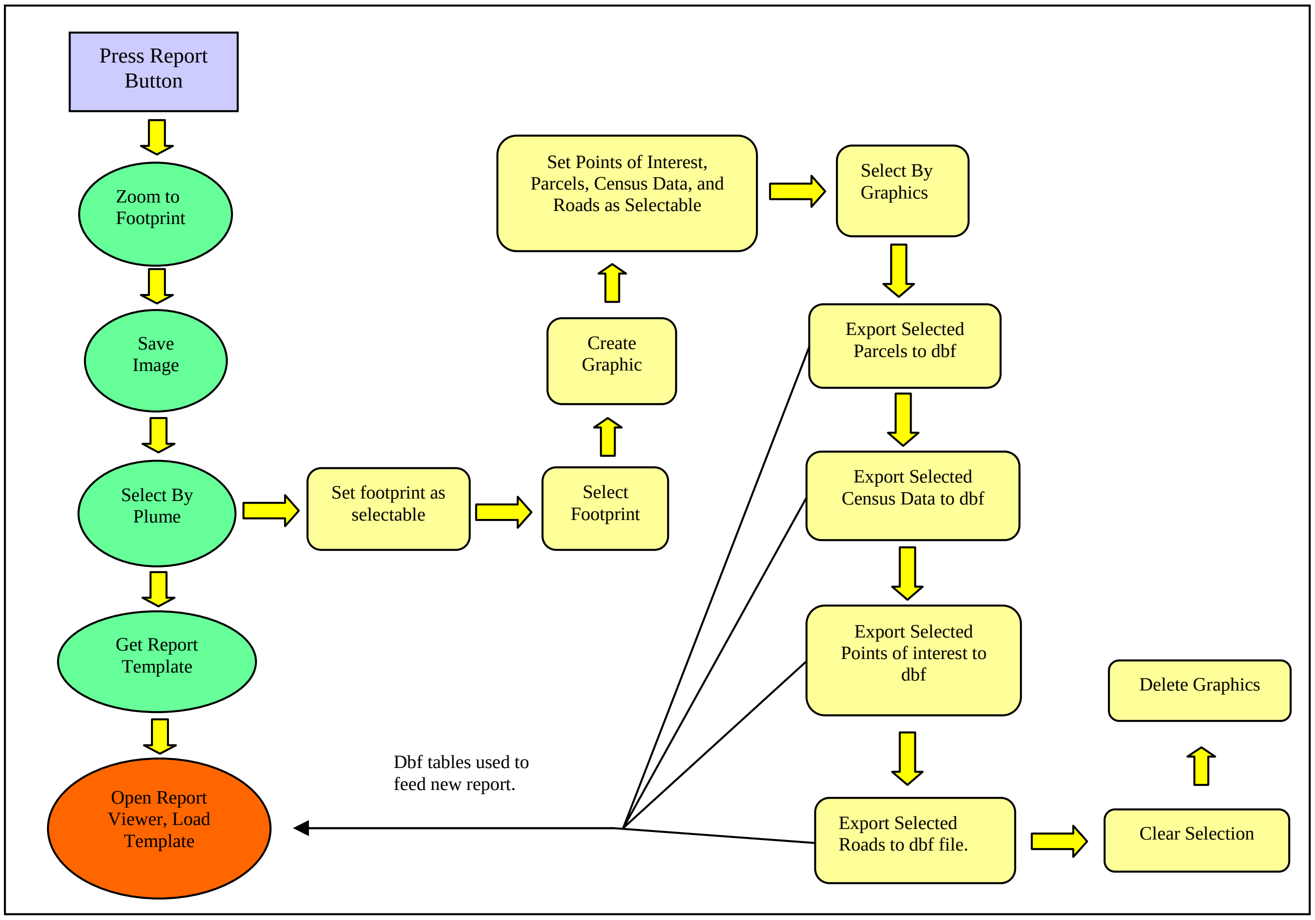

Press Report Button
Zoom to Footprint
Save Image
Select By Plume
Get Report Template
Open Report Viewer, Load Template
Set footprint as selectable
Select Footprint
Create Graphic
Set Points of Interest, Parcels, Census Data, and Roads as Selectable
Select By Graphics
Export Selected Parcels to dbf
Export Selected Census Data to dbf
Export Selected Points of interest to dbf
Export Selected Roads to dbf file.
Clear Selection
Delete Graphics
Dbf tables used to feed new report.

Figure 12. Plume footprint analysis and summary report workflow.

```
'********************************* Code Used to Create the Plume Hazard Summary Report ***********************************
Call ZoomToLayer                                                    '** Calls a function to zoom to the new plume footprint.
Call ExportActiveView                                       '** Function to create jpeg of current map view.
Call Selectbyplume                                                  '** Function to perform selection by plume.
                                                            '** The code for this function is outlined below.

Dim crReport As New CRAXDRT.Report                                  '** Creates a Crystal Reports object.
Dim crApp As New CRAXDRT.Application
Dim crTables As CRAXDRT.DatabaseTables

Set crReport = crApp.OpenReport("C:\ALOHA\DATA\Report3.rpt")        '** Defines a pre configured Crystal Report Template.
crReport.UseIndexForSpeed = True

Load frmReport
frmReport.crvMain.ReportSource = crReport
frmReport.crvMain.ViewReport
frmReport.Show 1                                            '** Open the Report.

'************************************* Code to Perform Selection by the Plume *******************************************
Call SetSelectable("footprints")                                    '** Set the footprint layer to the only selectable layer.

Dim pMxDoc As ImxDocument
Dim pMap As IMap
Dim pFeatSel As IFeatureSelection
Dim pSelSet As ISelectionSet
Dim pActiveView As IActiveView

Set pMxDoc = Application.Document
Set pActiveView = pMxDoc.FocusMap
Set pMap = pMxDoc.FocusMap
Set pFeatSel = pMap.Layer(0)                                '** Set the feature layer to the footprint layer.

pFeatSel.SelectFeatures Nothing, esriSelectionResultNew, False
pFeatSel.SelectionChanged                                   '** Select the footprint layer.

pActiveView.PartialRefresh esriViewGeoSelection, Nothing, Nothing   '** Refresh the selection.

Dim pEnumFeature As IEnumFeature
Dim pFeature As Ifeature
Dim pElement As IElement
Dim pGraphicsContainer As IGraphicsContainer

Set pGraphicsContainer = pMxDoc.FocusMap                    '** Create a graphic element based on each selected feature
Set pEnumFeature = pMxDoc.FocusMap.FeatureSelection
pEnumFeature.Reset
Set pFeature = pEnumFeature.Next

Do While Not pFeature Is Nothing
    Select Case pFeature.Shape.GeometryType                 '** Determine the geometry type and make the appropriate
        Case esriGeometryPoint                              '** symbol element.
            Set pElement = New MarkerElement
        Case esriGeometryPolyline
            Set pElement = New LineElement
        Case esriGeometryPolygon
            Set pElement = New PolygonElement
   End Select

    If Not pElement Is Nothing Then                         '** Give the feature shape to the element.
       pElement.Geometry = pFeature.Shape
       pGraphicsContainer.AddElement pElement, 0
    End If

    Set pFeature = pEnumFeature.Next                        '** Get the next feature in the selection.
Loop                                                        '** Repeat this process for all features of the footprint.
```

Figure 13. This code sample illustrates how the Plume Hazard Summary Report is created and how the data that populates the report is generated.

```
'*********************************** Code to Perform Selection by the Plume  - Continued *************************************

Call SetSelectable("pointsofinterst")                          '** Set the points of interest layer to selectable.
Call SetSelectable("census_data")                              '** Set the census data layer to selectable.
Call SetSelectable("parcels")                                          '** Set the parcels layer to selectable.
Call SetSelectable("Roads")                                            '** Set the roads layer to selectable.
Call SetUnSelectable("zip5_a")                                 '** Set the zip code layer to selectable.
Call SetUnSelectable("County_2000")                            '** Set the county layer to selectable.

Dim pGCSel As IgraphicsContainerSelect                         '** Get the graphics container and select all of the graphics in it.
Set pGCSel = pMxDoc.FocusMap
pGCSel.SelectAllElements

pMxDoc.ActiveView.Refresh
pMxDoc.UpdateContents                                          '** Update the display.

Dim pCmdItem2 As ICommandItem
Set pCmdItem2 = Application.Document.CommandBars.Find(arcid.Query_SelectByGraphics)          '**'Select by graphics.
pCmdItem2.Execute

Call ExportDBF(pMap, "parcels")                                '** Export the selected parcels to a dbf file.

Call ExportDBF(pMap, "pointsofinterst")                        '** Export the selected points of interest to a dbf file.

Call ExportDBF(pMap, "census_data")                            '** Export the selected census data to a dbf file.

Call ExportDBF(pMap, "Roads")                                  '** Export the selected roads to a dbf file.

Dim pCmdItem13 As ICommandItem
Set pCmdItem13 = Application.Document.CommandBars.Find(arcid.Query_ClearSelection)          '** Clear the selection.
pCmdItem13.Execute

Dim pAV As IActiveView
Set pAV = pMxDoc.ActiveView
Set pGraphicsContainer = pMxDoc.ActiveView

pGraphicsContainer.Reset                                       '** Reset the graphics container.

Set pElement = pGraphicsContainer.Next
    pGraphicsContainer.DeleteAllElements                       '** Deletes the graphic elements.
pGraphicsContainer.Reset

Set pElement = pGraphicsContainer.Next

pAV.Refresh                                                      '** Refresh the view.
```

Figure 14. Continuation of code from Figure 13.

Dakota in January of 2002. The train derailment caused a chemical spill that caused a cloud of toxic ammonia to disperse over a large area of town. Hundreds of people suffered injuries and one man was killed as a result of the incident.

Now imagine that emergency managers had a GIS enabled air dispersion modeling tool at their disposal to help analyze the situation. This tool could provide information on the number of people that will be affected by the spill, information on critical infrastructure such as schools, nursing homes, and businesses that would be affected. Emergency Managers could use this information to help make decisions during time critical situations.

For our example, a plume footprint will be generated based on the release of ammonia, which is a commonly transported chemical in Scott County, MN. It is assumed that the chemical release happened at a busy intersection in a densely populated area. First we open the ArcMap project that contains our tool and data. We then click on a point on the map to define the location of the spill as seen in Figure 15.

The ALOHA Plume Model Tool dialog appears with the real-time weather for the area that was clicked as shown in Figure 16. After the dialog box appears, the user can enter in the appropriate variables and press the Start ALOHA button. The parameters are entered into ALOHA and the user is presented with the choice of what type of hazard they would like to model as seen in Figure 17. In this case, the user chooses Toxic Area of Vapor Cloud and press OK.

ALOHA then creates the plume's footprint as shown in Figure 18. It is at this point a user returns to the ArcMap project and presses the Plot Footprint button. The plume is imported into ArcMap as shown in Figure 19. The footprint is converted to a shapefile and imported to the map correctly symbolized. The next step is to press the Report button to perform the analysis. A Crystal Report's report viewer appears displaying the template. The user then presses the refresh button to generate the report (Figure 20). The report can now be sent to a printer or saved as a pdf.

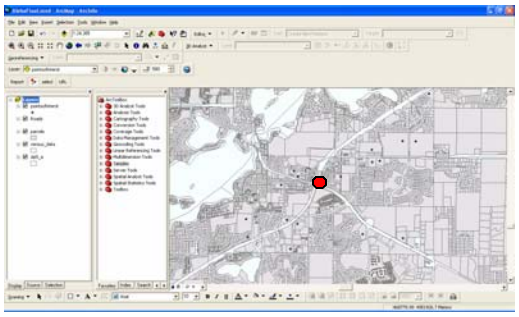
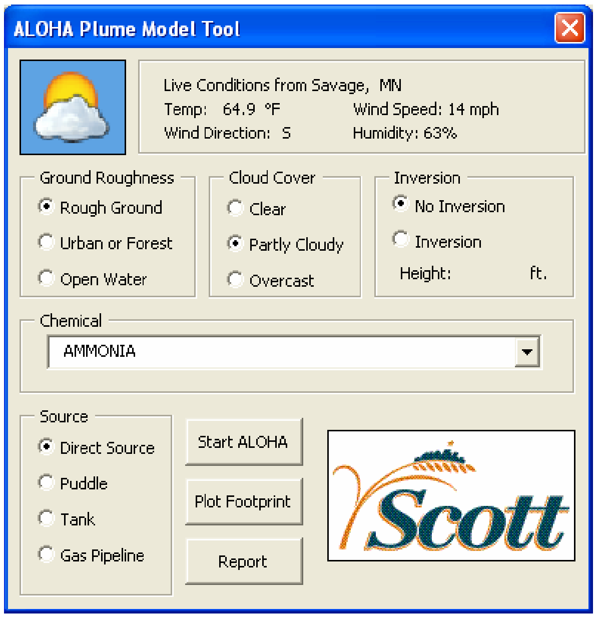


Figure 15. Location defined by clicking on the map.

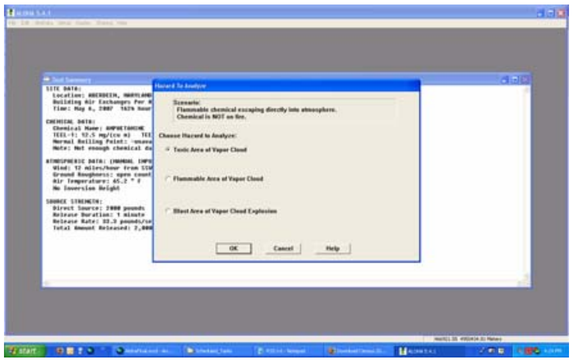


ALOHA Plume Model Tool

Live Conditions from Savage, MN
Temp: 64.9 °F   Wind Speed: 14 mph
Wind Direction: S   Humidity: 63%

Ground Roughness: [✓ Rough Ground] Urban or Forest / Open Water
Cloud Cover: Clear / [✓ Partly Cloudy] / Overcast
Inversion: [✓ No Inversion] / Inversion   Height: ft.
Chemical: AMMONIA
Source: [✓ Direct Source] / Puddle / Tank / Gas Pipeline

Figure 16. ALOHA Plume Model Tool dialog.



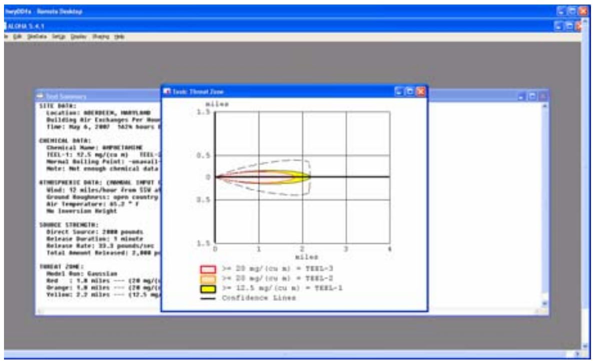
```
SITE DATA:
  Location: ABERDEEN, MARYLAND
  Building Air Exchanges Per H
  Time: May 6, 2007  1624 hour
CHEMICAL DATA:
  Chemical Name: AMPHETAMINE
  TEEL-1: 12.5 mg/(cu m)   TEE
  Normal Boiling Point: -unava
  Note: Not enough chemical da
ATMOSPHERIC DATA: (MANUAL INPU
  Wind: 12 miles/hour from SSW
  Ground Roughness: open count
  Air Temperature: 45.2 ° F
  No Inversion Height
SOURCE STRENGTH:
  Direct Source: 2000 pounds
  Release Duration: 1 minute
  Release Rate: 33.3 pounds/se
  Total Amount Released: 2,000
```

Hazard To Analyze

Scenario:
Flammable chemical escaping directly into atmosphere.
Chemical is NOT on fire.

Choose Hazard to Analyze:
[✓ Toxic Area of Vapor Cloud]
Flammable Area of Vapor Cloud
Blast Area of Vapor Cloud Explosion

Figure 17. ALOHA software after the variables have been entered.



```
SITE DATA:
  Location: ABERDEEN, MARYLAND
  Building Air Exchanges Per Hou
  Time: May 6, 2007  1624 hours
CHEMICAL DATA:
  Chemical Name: AMPHETAMINE
  TEEL-1: 12.5 mg/(cu m)   TEEL-
  Normal Boiling Point: -unavail
  Note: Not enough chemical data
ATMOSPHERIC DATA: (MANUAL INPUT
  Wind: 12 miles/hour from SSW a
  Ground Roughness: open country
  Air Temperature: 45.2 ° F
  No Inversion Height
SOURCE STRENGTH:
  Direct Source: 2000 pounds
  Release Duration: 1 minute
  Release Rate: 33.3 pounds/sec
  Total Amount Released: 2,000 p
THREAT ZONE:
  Model Run: Gaussian
  Red   : 1.8 miles --- (20 mg/(
  Orange: 1.8 miles --- (20 mg/(
  Yellow: 2.2 miles --- (12.5 mg
```

Figure 18. The plume footprint created by ALOHA.

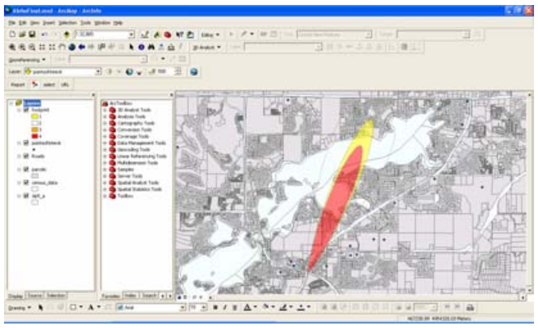


Figure 19. The plume footprint imported to ArcMap.

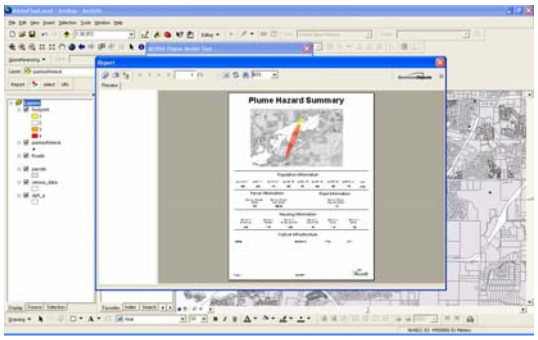


Figure 20. The final result, a hazard summary report.

Report Results

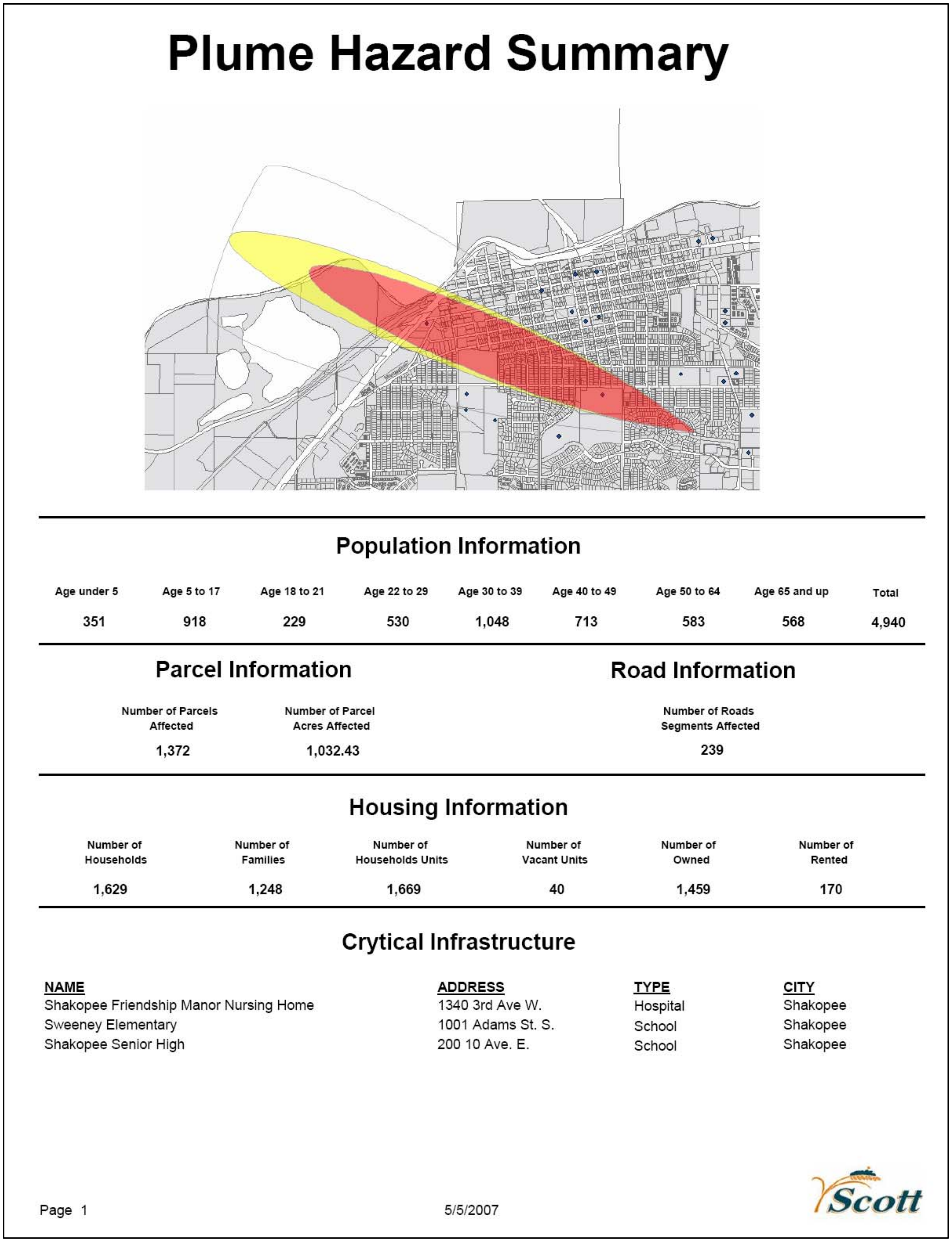
The results of this analysis show a fairly large population that is affected by the hypothetical chemical release (Figure 21). The population information section shows the total population of affected people to be 4,940. There are some additional population statistics in this report that can be useful to emergency managers, especially for identifying the number of people most vulnerable to a chemical release like the very young and the elderly. For example, because the data is broken down by age groups, an emergency manager can ascertain there are 1,151 people who were 50 years old at the time of the census and would be most likely to be at home during the day. Previously, this type of analysis required the help of a fairly experienced GIS user. Using this automated tool, an inexperienced GIS user can conduct this type of valuable analysis in a matter of minutes.

The next section of the report encompasses parcel information. This section reveals there are 1,372 parcels encompassing an area of 1,032 acres affected by the chemical plume. This data can help emergency managers gauge the extent of the potential cleanup and evacuation activities.

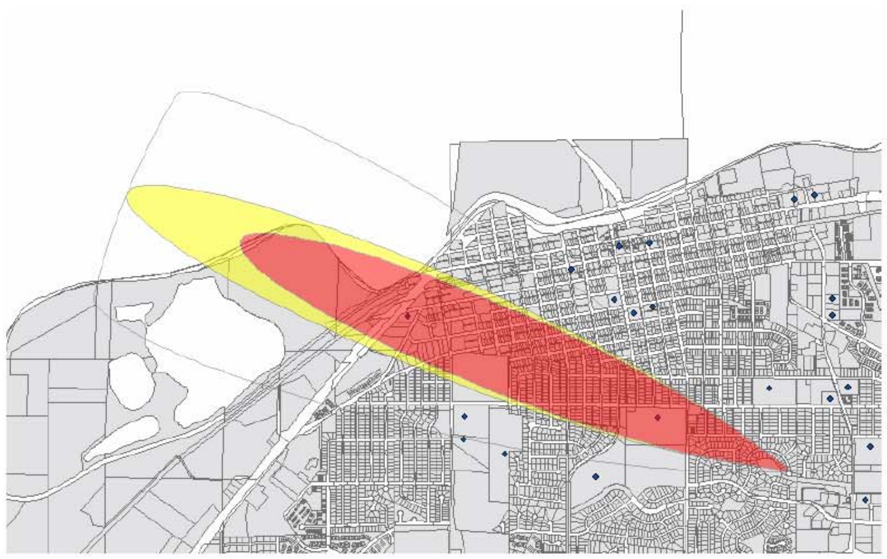
The road information section shows that there are 239 road segments that are affected by the chemical release. This data could be used to determine the number of roads that will need to be shut down to secure the area.

The housing information section of the report has several sections that that reveal how many buildings are within the plume area. One item that is important to emergency managers is the number of family units, which can reveal the residential makeup of the plume hazard area. In this case there are 1,248 family housing units in the plume area, which indicates that the area is highly residential. Another important piece of information is the number of rented units. This can help the emergency manager plan for the type of residential units that will need to be evacuated. In this case, there are 170 rental units, which indicates that they will be evacuating mostly residential homes rather then apartment buildings.

The final section of the report is the critical infrastructure section. This section shows the name and address of critical infrastructure such a schools and hospitals that are within the plume hazard area. In the case of this incident,



Plume Hazard Summary



Population Information

Age under 5	Age 5 to 17	Age 18 to 21	Age 22 to 29	Age 30 to 39	Age 40 to 49	Age 50 to 64	Age 65 and up	Total
351	918	229	530	1,048	713	583	568	4,940

Parcel Information

Number of Parcels Affected	Number of Parcel Acres Affected
1,372	1,032.43

Road Information

Number of Roads Segments Affected
239

Housing Information

Number of Households	Number of Families	Number of Households Units	Number of Vacant Units	Number of Owned	Number of Rented
1,629	1,248	1,669	40	1,459	170

Crytical Infrastructure

NAME	ADDRESS	TYPE	CITY
Shakopee Friendship Manor Nursing Home	1340 3rd Ave W.	Hospital	Shakopee
Sweeney Elementary	1001 Adams St. S.	School	Shakopee
Shakopee Senior High	200 10 Ave. E.	School	Shakopee

Page 1 5/5/2007 Scott

Figure 21. Sample report generated for case study.

there is one hospital and two schools that are affected by the chemical plume.

This information, which can be calculated in a matter of minutes, is a valuable resource for emergency managers. The data contained in each report can aid in the decision making process and add to the overall situational awareness of a chemical response. The tool provides a means for an inexperienced GIS user to conduct GIS analysis quickly without a lot of training.

Conclusion

Emergency Managers have realized the benefits of using Geographic Information Systems for visualization of information and as an important planning tool. The combination of this technology with the modeling capabilities of air dispersion modeling software has proved to be an invaluable resource for responding to chemical release scenarios. These tools can be used to respond to emergency release scenarios like natural, accidental, and intentional chemical releases and also for planning and training purposes like contingency planning and short-term site assessments.

The use of GIS enabled air dispersion modeling provides an additional level of situational awareness not available from either stand-alone GIS software or dispersion modeling software and it should be a tool available to every emergency manager.

References

Bacon, D. 2000. Real Time Modeling and Emergency Response Forecast. *Mesoscale Atmospheric Dispersion*. pp. 171-192. Retrieved February 10, 2007 from EBSCO database.

Chakraborty, J. and Armstrong, M. 1994. Estimating the Population Characteristics of Areas Affected by Hazardous Materials Accidents. *GIS-LIS.* Pp. 154-163. Retrieved February *10, 2007 from EBSCO database.*

Hunt, K., 2005. Emergency Response Planning and Integrating ALOHA Plume Models. *ArcNews.* Summer 2005. Retrieved February 12, 2007 from http://www.esri.com/news/arcnews/summer05articles/genesee-county.html.

Turpin, R. 2004. Air Plume Modeling… Planning or Diagnostic Tool. *Environmental Protection Agency.* Retrieved February 17, 2007 From http://www.ofcm.gov/atdworkshop/proceedings/session1/campagna.pdf

Tomaszewski, B. 2003. Emergency Response and Planning Application Performs Plume Modeling. *ArcUser.* December – October. Retrieved February, 17 2007 from http://www.esri.com/news/arcuser/1003/plume1of2.html

Westbrook, J. 1999. Air Dispersion Models: Tools to Assess Impacts from Pollution Sources. *Natural Resources & Environment.* Spring 1999. Retrieved January 15, 2007 2005 from, EBSCO database.