Ingesting Geospatial Data into Hazard Services’ Database for National Weather Service Flood Alerts

Nkosi Muse

Academic Affiliation, Fall 2015: Junior, University of North Carolina at Charlotte

SOARS® Summer 2015

Science Research Mentors: Kevin Manross, Joe Wakefield, Tracy Hansen
Writing and Communication Mentor: Lesley Smith
Computing Mentors: Ryan May and Sean Arms
Coach: Jeff Weber
Peer Mentor: Meghan Mitchell

Abstract

Statistically, flooding is the most devastating natural disaster not only in the US, but in the world. Of the many different severe and normal weather events, floods typically cause the most casualties and damage. Within Hazard Services of the National Oceanic and Atmospheric Administration, software is being developed and improved for use by the National Weather Service to help prevent such disasters. Because flooding is a worldwide event, the new program produced by this Hazard Services project was created with the intention of being used anywhere for the safety of the public. Time is of the essence in a flooding event and forecasters need to have all relevant information readily available. This project’s work provides a script that gives forecasters the tools to issue quick and efficient warnings by creating a catalog of flood prone areas, customized for the regions that each forecast office covers. By ingesting unique geospatial data all at once with a merged file in the form of dams, rivers, and burn scars into a relational database, forecasters can simply choose the shape and initiate a watch, warning or advisory within the Advanced Weather Interactive Processing System (AWIPS) software. Having access to pre-set outlines of dam-break outflow, burn scars, and river inundation is much more efficient than drawing the shape of an area that is expected to flood by hand. Thus, not only does the new program allow faster issuance of alerts, but it has left less room for error and more room for geospatially accurate alerts.

With this program, we were able to store and then retrieve shapefiles from a database within the AWIPS software swiftly, and much more efficiently than before.

This work was performed under the auspices of the Significant Opportunities in Atmospheric Research and Science Program. SOARS is managed by the University Corporation for Atmospheric Research and is funded by the National Science Foundation, the National Center for Atmospheric Research, the National Oceanic and Atmospheric Administration, the Woods Hole Oceanographic Institute, the Center of Excellence in Remote Sensing Education and Research at Elizabeth City State University, the University of Colorado at Boulder, and by the Center for Multiscale Modeling of Atmospheric Processes.
Introduction

Severe weather events around the U.S and even the world are significant threats to human safety, especially as the planet becomes warmer and weather patterns begin to change. Despite how far we meteorologists, researchers, and developers have come, we are still not always able to move all human lives out of harm’s way. While events like hurricanes and storm systems have longer lead times, events like flash floods due to dam breaks do not. Thus, when such an event occurs, forecasters need to be able to warn the public as soon as possible. In order to do their job as best as possible, they need to use computer software that is quick and efficient. Therefore, the purposes of a project such as this one are to give forecasters everywhere the tools to boost those qualities when it comes to warning the public of potential severe weather events, more specifically potential hydrologic disasters.

Flooding is not only the most common and widespread natural disaster related to severe weather, but it has also proved to be the most deadly, as it has killed more people annually than tornadoes, hurricanes, or lightning (National Severe Storms Lab, 2015). Accounting for forty percent of all natural disaster events worldwide, flooding has led to approximately two hundred American deaths annually as well as billions of dollars in damages (FEMA, 2015). While the bodies of water that surround us are useful in many ways during daily life, they can also become the destructive overflows that end life should those in its path not be prepared. While some areas may be at higher risk from severe events like tornadoes or hurricanes, many areas on the planet are prone to flooding. Risks can be higher in densely populated areas or even less populated areas close to rivers, streams, or creeks.

In cities, infrastructure has taken away land’s natural defense against floods, and thus water can move easily and at dangerously fast speeds over flat surfaces. Poorly dug banks and
under-maintained dams give birth to the most dangerous natural disaster of all, flash floods. Flash floods make flooding even more hazardous, as they can strike without warning from a sudden release or large addition of water over a short period of time.

Besides prevention, prediction is the best tool when it comes to the safety of people in any type of flood disaster. From the data scientists have, the past can be the best predictor of the future. Prediction involves being able to know when these severe events may strike, as well as being able to alert those in its path about it in a timely manner. While hydrometeorologists and severe weather researchers work to develop databases of flood data that can help to better understand potential future events (Gourley et al., 2013), other researches and developers work on technology and software that can provide quick and efficient alerts to the public.

The National Weather Service has always tried to make daily life safer for people when it comes to weather prediction. The technology they’ve used over the years has saved countless lives, and it’s only becoming more advanced. Some of the scientists that work to develop better and more efficient software are part of teams such as Hazard Services of the Global Systems Division in the Earth System Research Laboratory. The goal of the Hazard Services Project is to ultimately make the process of issuing severe weather alerts as simple as possible by integrating multiple programs into one efficient one. This will result in less required training, better results, quicker warnings, and safer residents (Earth System Research Laboratory, 2015). While making this program user friendly is important, it is more important that the main objective is completed as quickly and accurately as possible.

When an area is to be affected by severe weather, every second is crucial when it comes to issuing an advisory, watch, or warning, especially with flooding. The shapes that are drawn to specify these areas can be inaccurate if a forecaster does not have the ideal amount of time to
draw a shape that will encompass the exact area. For areas such as dams which can be very miniscule and unique on a map, or rivers and creeks which have irregular shapes and can change over time, drawing the affected area with prime accuracy is nearly impossible. Low resolution shapes with minimal coordinates defining the area can also lead to inaccuracy and result in over- or under-warning a location. While it is better to overdraw than under draw an area that is to be affected by severe weather, it is important to be as accurate as possible to make life as convenient and safe as possible for the residents of that area. Should a forecaster already have a pre-drawn shape for an area that can be deployed, he or she wants to be able to find this shape quickly without wasting valuable time.

The program created for this project was made to fill the holes in this process by introducing an organized database of these different high resolution, pre-drawn polygons. Should a forecaster have to issue a warning for a river bordering a town, he or she can query the database using National Weather Service (NWS) provided software and this program to quickly and efficiently deploy the appropriate shape onto the map for eventual issuing. The same can be done for a dam area or an area near a wildfire burn scar that does not have much protection from flooding. The program will be useful to any forecast office that encourages efficiency and speed with their forecasts.

Methods

Weather forecasters and researchers of the world are constantly looking for new ways and methods to make forecasting faster and more efficient. The program we have created is another step in that direction. Not only does the program make it easier for forecasters to warn the public of severe weather, it helps provide a safer place for people by giving them “actionable” information. Because it can conform to any area of the world, the program is able to
be used wherever there is a forecast office or more specifically, a computer running the AWIPS2 system.

This program relies on digital geospatial data that weather offices around the country and even around the world use to designate areas of severe weather impacts. In order to create this program, we used this geospatial data, organized in the form of what’s called a shapefile. Not only will the usefulness of this program rely on the input of shapefiles, but the testing of this program assures that it could be used by any forecaster. What makes this program so advantageous is the fact that it can take in geospatial data from any point on this planet, in the form of these shapefiles. Therefore any forecast office, such as the NWS, will be able to use this program to create archives and databases of their documented geospatial data. The program is able to manipulate this data and use whatever software that is necessary or available to make the issuance of severe weather alerts most efficient. Besides the program itself, the most important part of it is the data, provided by these forecast offices wherever they may be. Each forecast office covers a certain geographical area, and thus will have different geospatial data at its disposal.

As mentioned briefly before, this geospatial data is in the form of a shapefile. A shapefile, denoted with the filename extension .shp, is a vector data storage format for the storing the location, shape, and attribute of geographic features. A shapefile is always accompanied by two mandatory files: an index file (.shx) and a database file (.dbf). The index file is a geographical positional index of the shapefile geometry and the database file is the collection of columnar attributes for the shapes(s). The .shp, .shx and .dbf files can also be accompanied by other files such as .sbn or .sbx (spatial indices) or a .prj file (projection format). The shapes or polygons themselves are characterized by latitude and longitude coordinates around their
borders. Using software such as the programs to be later discussed, these shapes can be altered, moved, or even merged (ESRI, 2015).

The software used for this project involves two different types: one that we use to directly write out a program of commands, and one provided by a weather software company. The programming software we use is called Bash, known as a shell or command language interpreter (GNU, 2015). Using Bash, we are able to create or type scripts that can run a compilation of our commands. These commands are the ones that run this program, as well as commands that carry out our everyday, common computer tasks. The software we have utilized is the Advanced Weather Interactive Processing System (AWIPS). AWIPS II, a weather forecasting and analysis package that is currently being developed by the National Weather Service and Raytheon, has been in more recent use (Unidata/UCAR, 2015). More specifically, AWIPS is the software that we use to analyze and depict an area of severe weather, ultimately leading to the issuance of a severe weather alert. In AWIPS you can draw your own polygon or shapefile, or you can pull geospatial data from a database -- the database where our program will be importing shapefiles.

The main objective of the program is to ingest geospatial data into a database, so that the forecaster, who is AWIPS II trained and fluent, can access the shapefile or polygon, deploy it onto the software map, and from there quickly and efficiently issue a severe weather warning (Figure 1). In this process, most of our work consisted of designing the program that takes the geospatial data, merges it, and pushes it into a database, known as the PostgreSQL database. The Bash script or program that carries out these commands is broken into a few general, efficient steps:

1. Find all needed shapefiles in a certain computer directory.
2. Merge all shapefile polygons into one shapefile layer (Figure 2).
3. Import this merged shapefile into the database.

The user will have to give the script the additional information to run properly and provide the desired result. This information includes the type of shapefile they are ingesting (dam, burn scar, river) to enter its respective table, the file directory of the computer where the shapefiles are located, and if desirable or necessary, two optional pieces of information. The first optional information is a name for the merged shapefile, should the user want to save it to their computer. The second piece of optional information is the title of a column within the database’s table that may need to be renamed to “Name.” The AWIPS software will recognize only a column titled “Name” when querying the database (Figure 3).

The user will enter this information from the command line or the terminal, which the program will read in statements known as arguments. These arguments are headed with a comment known as a flag, a dash simply followed by a letter. For example, the shapefile directory argument will be headed with a “-s” so the interpreter knows what will follow the flag and what to do with what’s entered. Following the flag will be the shapefile directory. A shapefile directory argument would look like this: “-s /home/nkosi.muse/mergetheseshapefiles”.

The argument format used allows each argument entered to be organized and separated from one another to prevent confusion in the case of a potential user error.

Discussion

The general purpose of this project was to write a program that could import shapefiles into a database properly so they could be read by the AWIPS software. Fortunately, Raytheon Company had a script developed to import a single shapefile into the database. However, there were a number of things that the original script, titled “importshapefile.sh”, did not do to satisfy the original task, as well as make the program as user-friendly as possible. The main issue to be
addressed was that the database would take in shapefiles only one by one, following the deletion of the previously ingested shapefile. Thus, a merging of shapefiles was necessary in order to get all into the database. In addition, merging results in a more organized collection of geospatial data. The original Raytheon program did not merge the shapefiles (which includes naming and saving a merged file), allow specific optional arguments to be run, or rename columns within the database so that they could be read. The Raytheon program could come across as too abstract to some, which is inefficient when demonstrating how to use the script to a new user.

In moving forward with the script to make these changes, many of which were suggested along the way, a lot of talks took place regarding how to go about completing such a script. While “importshapefile.sh” was done in shell scripting or Bash, this script could be written in many different languages such Python, Perl, or others. Since Bash was the language that had to be understood in order to move this program forward, that was the language we continued the project in. Bash held advantages in convenience, being that many of the modules needed to complete it were built in. In other languages such as Python, such modules would need to be called or imported, which could become a nuisance. One of those modules included a tool from the Geospatial Data Abstraction Library (GDAL). Within GDAL is “ogr2ogr,” a tool that can perform various functions in manipulating and formatting geospatial data (GDAL, 2015). With ogr2ogr, we were able to merge the shapefiles.

Because National Weather Service offices have access to or use only certain software, what we were able to use was limited as well. This meant modules or tools such as Python’s “osgeo” or “fiona” which would have simplified the process of creating this script, were not available for use within AWIPS2; therefore python was not used. Because Bash or shell scripting is simple and always available, it had an advantage over other languages.
While the “importshapefile.sh” script had multiple commands, most optional, many of these could be disposed of to drastically simplify the amount of arguments the user would have to enter. Some of these arguments included table information for the database, which would remain constant within use here in Hazard Services. With some arguments set to constants, it made room for more important optional arguments. Optional arguments could bring about unwanted results when it comes to skipping a command for the next. For example, if one wanted to use optional argument number four, they’d have to skip optional argument number three. In a command line, one cannot simply skip over an optional argument and use the next optional argument without the program reading the user’s entry incorrectly (reading the fourth argument entry as the third argument). Thus, the way we set up the commands had to take this into account. To solve this problem, a Bash module known as “getopts” was introduced to the script. The tool “getopts” is a utility made to parse options or arguments from the command line (MKS Software, 2015). In this scenario, getopts parses arguments from the script, and it does so in the form of a designated letter preceded by a dash. This flag, as previously mentioned in the methods section, can be assigned to read an argument value that is to follow it, or the flag can be the argument itself. Most conveniently, it allows optional arguments to be skipped or disregarded with ease and without the issue of being mixed up. The flag that precedes all values will be read and designate the entered value accordingly.

The two optional arguments considered here are one: assign the merged shapefile a name to save, and two: rename a column were added for convenience of the user. If the user wants to save this merged file, he or she can include a name that follows the “-m” flag; otherwise the merged file we be deleted from the user’s directory after being ingested into the database. In
renaming the column, the user can simply enter the name of the column that needs to be renamed following a “-c” flag, rather than having to manually change the column name on their own time.

Results

After rigorous testing, including by forecasters of the National Weather Service, followed by minor changes, we were able to configure the script to achieve all desired goals set at the beginning of the summer. The program, now known as “ingestshapefiles.sh,” is able to merge a shapefile, ingest it into its respective table within the database from any directory on the user’s computer and save it if so desired. Within the database, a user can also rename the column read by AWIPS to “Name.” In AWIPS, the user can pull the shapefile from its respective recommender to deploy it on the map and issue a warning (Figure 4). There are plans to have this code implemented operationally within the next year.

In addition to the “ingestshapefile” script, there is some future work that can be done with this project. Dam and burn scar shapefiles can also come with associated data to better describe a flooding event. This data comes in the form of a text or xml file, and has information such as the river name, dam name, county or city name, as well as scenarios to describe the flooding event. These scenarios identify how much water is involved with the flooding and how fast this water may be moving. To link the shapefiles with the data, we must develop a program to parse specific data for specific situations. With this additional data, residents can be more informed when they receive any type of flood alert.
Figure 1. General overview of the script, from the directory to the command line, and then into AWIPS2.
Figure 2. Merged shapefile to be ingested into the database.
Figure 3. The user uses the recommender to AWIPS to query the database for the shapefiles.
Figure 4. The user deploys the shapefile on the map to issue a warning.
Acknowledgments

I’d like to thank the SOARS staff and the steering committee for this wonderful opportunity to do research and software development under some of the best scientists in the world. Thank you to my mentors--Kevin Manross for pushing me to innovate, Joe Wakefield, Tracy Hansen, and Susan Williams for all of your productive feedback and encouragement. Thank you to my fellow proteges for personal, professional, and scientific development/growth. A final thanks to Rebecca Haacker and Rebecca Batchelor for putting the proteges first and for helping us to be the best young scientists we can be.
References


Unidata (UCAR), cited 2015: Unidata AWIPS II. [Available online at http://www.unidata.ucar.edu/software/awips2/]

SOARS® 2015, Nkosi Muse, 16