**GIS Strategy Memo (Draft)**

**Re-Engineering SWMM/EPANET User Interface Application Software Architectures**

**RESPEC/AQUA TERRA Team**

**April 15, 2016**

**Background**

The purpose of this project is to reengineer the User Interface (UI) architecture for the EPA’s collection system and water distribution system simulation products, the Storm Water Management Model (SWMM) and software that models water distribution-piping systems (EPANET). The project focuses on the design of a modular and extensible user interface application software architecture for SWMM and EPANET. The modular and extensible architecture will enable deployment of new application features created by the EPA, third-party developers, and end users employing scripts and application “plug-ins.”

The current SWMM and EPANET user interfaces incorporate an editable graphic display of a network of gages, pumps, nodes, links, and other components. This basic display cannot directly handle any geospatial data, leading to difficulties for the user when importing and creating a network that needs to be to-scale. The statement of work for this effort calls for GIS integration, providing functionality including *data import/export from/to geodatabase*; *map rendering*; and *input/output data manipulation*.

Our team’s technical proposal describes an important choice to be made in how to provide GIS functionality within the new products. A true Geographic Information System (GIS) mapping component will greatly ease the model setup process and provide geospatial context for any project. We propose building upon an existing full-featured GIS that can handle multiple file types, reprojection, and dynamic background layers. We believe it is worth starting with an existing open-source GIS that already has all of the features users might expect.

QGIS is the leading open-source interactive GIS. QGIS has all of the GIS features needed, deep Python scripting integration, an extensive library of C++ and Python plugins, a large user base across the three desktop platforms, and is supported by an active developer community. Using QGIS as a map component within a new main program closely mirrors the existing SWMM and EPANET architecture.

An alternate approach is to use a web-based tool such as Leaflet (<http://leafletjs.com/>), an open-source JavaScript library for mobile-friendly interactive maps. In this case, the map canvas within the re-engineered EPANET and SWMM UIs would consist of a web viewer accessing files on the local computer.

**Required GIS Functionality**

To assess these two alternate approaches, we have listed below the GIS functionality required in the new EPANET/SWMM applications.

*Geodatabase Import/Export:* The UIs will allow the import of GIS data from a file geodatabase into the model data structure, and export from the model data structure into a file geodatabase. These features will require the GIS data be in an accepted pre-determined data format.

*Map Rendering:* The UIs will provide map displays with options for rendering features by color, line style, thickness, etc. for rendering model results on the map.

*Map Interaction:* The UIs will provide map interaction functionality including the ability to pan, zoom in, zoom out, draw at full extent, and set reference coordinates and distance units.

*Print Map:* The UIs will provide a means for the user to send the map display to a printer.

*Map Projection / Bounds Editing:* The UIs will provide a means for editing map projection specifications, the X and Y coordinates of the map’s bounding rectangle, as well as distance units.

*Subcatchment Polygon Editing:* The SWMM UI will provide a means for adding, removing and editing the X and Y coordinates for each vertex of the subcatchment polygons. (This requirement applies to the SWMM UI only, as subcatchments are not applicable to EPANET.)

*Node Coordinate Editing:* The UIs will provide a means for editing the X and Y coordinates of the nodes.

*Link Vertex Editing:* The UIs will provide a means for adding, removing and editing the X and Y coordinates of the interior vertices of the polyline links.

*Map Labeling:* The UIs will provide a means for adding, removing and editing the X and Y coordinates and text of map labels.

*Map Backdrop Setting:* The UIs will provide a means for adding, removing and editing the X and Y coordinates of the map backdrop’s bounding rectangle and file name of the backdrop image for display in the map pane.

*Undo/Redo Support:* With regard to map interactions, the UIs will provide the ability to undo/redo recent changes.

*Digitization Support:* UI support for an end-user digitizing additional nodes, links, or subcatchments.

*Identification Support:* UI support for an end-user interactively identifying map attributes at a given point, such as elevation values from a DEM layer, as well as automated application of a value from one layer to another (for instance DEM elevations applied to all nodes of a network).

*Scripting Support:* In addition to the core features described above, the UIs will expose the full suite of GIS libraries so that a plug-in or script developer can create additional tools for advanced GIS interaction. An example of this type of third-party produced tool would be an automatic watershed delineator.

**Use Cases**

Below are some example use cases for GIS components within EPANET/SWMM. Documented with each case are assessments of how this use case can be achieved using each of the candidate GIS approaches.

|  |  |  |
| --- | --- | --- |
| **Use Case** | **QGIS Assessment** | **Leaflet Assessment** |
| 1. The user displays the spatial layers of a file geodatabase within the SWMM/EPANET application | Display seems to work ok in GQIS as long as the layer isn’t too large; editing the spatial features in a file geodatabase requires installing a driver from ESRI | Leaflet native format is GeoJSON. Leaflet also has a plugin for file geodatabase support. |
| 1. The user displays an animated map of the study area with conduits rendered in different colors based on a time series value, such as the amount of water flowing through | Rendering is available in QGIS 2.10.1, so this feature is presumably possible; we have not yet demonstrated animation through a timeseries in QGIS | Leaflet plugin for data visualization looks promising -- <http://humangeo.github.io/leaflet-dvf/>  See also <http://leafletjs.com/examples/choropleth.html>  for another example |
| 1. The user pans and zooms the map to an area of interest within the map canvas | Available in QGIS 2.10.1 | Done in Leaflet |
| 1. The user sends the map display to a printer | Available through QGIS print composer | Leaflet.print plugin |
| 1. The user enters the map projection specifications, so that distance on the map can be determined | QGIS has projection setting and distance measurement tools | Leaflet default projection is Google Mercator. Proj4Leaflet plugin is available for using other projections |
| 1. The user adds and moves vertices of map entity, such as a subcatchment polygon boundary | Polygon vertices can be moved through QGIS editing tools, for shapefile layers | Enabled through Leaflet.draw:  <http://leaflet.github.io/Leaflet.draw/> |
| 1. The user adds, removes, or edits the X and Y coordinates and text of map labels | Labeling the points of a point layer is available in QGIS | Example of editing text:  http://plnkr.co/edit/VzUfSD?p=preview |
| 1. The user adds a map backdrop image and sets its X and Y bounding box coordinates | Image can be added as QGIS backdrop; coordinate system and scale can also be set | Tiles example shows .png images shown as backdrop |
| 1. The user changes the location of a conduit endpoint, but before saving it decides to move it back to the original location | Rollback feature in QGIS user interface allows for undoing a change to a vertex location | Leaflet.draw example shows cancel event |
| 1. The user adds an additional link to the conduit network through digitizing | QGIS allows adding a feature to any map layer interactively, when using shapefiles | Available through Leaflet.draw |
| 1. The user applies elevation values from a DEM to each node of a network | QGIS has features such as zonal statistics, that summarize the grid values within a zone; a similar function can be developed to apply a value to each node | Raster display seems focused on tiles, may require more research to find best way to handle DEM layers |
| 1. The user wishes to automatically calculate the area of a subcatchment | Computed area in QGIS using Vector Geometry Tools | Leaflet.MeasureAreaControl is available |
| 1. The user writes a script which will automatically recalculate the length of each conduit in a network | Similar functionality to computing area in QGIS using Vector Geometry Tools | Leaflet.MeasureControl is available, could use a python library to edit attributes of selected layer format |
| 1. A third party developer writes a plug-in to add automatic watershed delineator functionality | QGIS has an existing watershed delineator, so we know the functionality is available | Leaflet’s geoprocessing functions seem limited, no apparent watershed delineation functions, suggest python scripts using leaflet for display |

**Conclusions**

Either approach (QGIS or Leaflet) seems feasible. The two main differences relate to distribution and scripting language. QGIS appears to be heavier to package compared to Leaflet, which only requires the webkit browser and a small amount of Javascript. With QGIS, the native code is in C++, whereas with Leaflet Javascript is the programming language.